ઉચ્ચતર માધ્યમિક અભ્યાસક્રમ

335 - કોમ્પ્યુટર સાયન્સ

અભ્યાસક્રમ સંયોજક

સુશ્રી કૌશલ્યા બારીક

ભાષાંતર સંયોજક

ડૉ. રાજેશ કુમાર

રાષ્ટ્રીય મુક્ત વિદ્યાલયી શિક્ષણ સંસ્થાન

(MHRD *નીચે સ્વાયત્ત સંસ્થા, ભારત સરકાર*)

A-24-25, *સંસ્થાકીય વિસ્તાર, સેક્ટર* -62, *નોઈડા* -201309 (*યુ. પી.*)

વેબસાઈટ : www.nios.ac.in, *ટોલ ફ્રી નં.* . 18001809393

© રાષ્ટ્રીય મુક્ત વિદ્યાલયી શિક્ષણ સંસ્થાન

મુદ્રણ :એપ્રિલ , 2015 પ્રત

સચિવ, રાષ્ટ્રીય મુક્ત વિદ્યાલયી શિક્ષણ સંસ્થાન, એ-24-25, ઈન્સ્ટિટ્યૂશનલ એરિયા નેશનલ હાઈવે 24, સેક્ટર-62, નોઈડા-201309 દ્વારા પ્રકાશિત અને મુદ્રિત

અભ્યાસક્રમ સમિતિ

ડૉ. એમ. પી. ગોઈલ
પ્રિન્સીપાલ સિસ્ટમ્સ એનાલીસ્ટ
નેશનલ ઈન્ફોર્મેટીક્સ સેન્ટર
નવી દિલ્હી

ડૉ. એ.કે.સોની
FMS, *દિલ્હી યુનિવર્સિટી*
નવી દિલ્હી

શ્રી. શશી ભૂષણ
રીડર, કોમ્પ્યુટર સાયન્સ
IGNOU, *નવી દિલ્હી*

શ્રી. સી.ડી. શર્મા
એજયુકેશન ઓફીસર
CBSE, *નવી દિલ્હી*

ડૉ. એસ.એસ. સંગલ
ડિરેક્ટર (એકેડેમિક)
નેશનલ ઓપન સ્કૂલ
નવી દિલ્હી

શ્રી.એસ.કે.પ્રસાદ
SAP
નેશનલ ઓપન સ્કૂલ,
નવી દિલ્હી

કુ. કૌશલ્યા બારીક
ટયુટર (કોમ્પ્યુટર સાયન્સ)
નેશનલ ઓપન સ્કૂલ,
નવી દિલ્હી

લેખક

ડૉ. એમ. પી. ગોઈલ
પ્રિન્સીપાલ સિસ્ટમ્સ એનાલીસ્ટ
નેશનલ ઈન્ફોર્મેટીક્સ સેન્ટર
નવી દિલ્હી

કુ. સંગીતા પંચાલ
હેડ, કોમ્પ્યુટર સાયન્સ
હંસરાજ મોડલ સ્કૂલ, નવી દિલ્હી

સંપાદક

ડૉ. એમ. પી. ગોઈલ
પ્રિન્સીપાલ સિસ્ટમ્સ એનાલીસ્ટ
નેશનલ ઈન્ફોર્મેટીક્સ સેન્ટર
નવી દિલ્હી

કુ. કૌશલ્યા બારીક
ટયુટર (કોમ્પ્યુટર સાયન્સ)
નેશનલ ઓપન સ્કૂલ,
નવી દિલ્હી

ભાષાંતર સમિતિ

ડૉ. કુન્તલ પટેલ (સંયોજક)
સ્કૂલ ઓફ કોમ્પ્યુટર સ્ટડીઝ
અમદાવાદ યુનિવર્સિટી,
અમદાવાદ

પ્રો. દેવેન્દ્ર વસી
નિરમા યુનિવર્સિટી,
અમદાવાદ

પ્રો. અનિરુધ્ધસિંહ પરમાર
સ્કૂલ ઓફ કોમ્પ્યુટર સ્ટડીઝ
અમદાવાદ યુનિવર્સિટી,
અમદાવાદ

પ્રિય વિદ્યાર્થીઓ

કોમ્પ્યુટર સાયન્સ વિષય પસંદ કરવા બદલ અભિનંદન. આ વિષય તમને કોમ્પ્યુટરને લગતી રસપ્રદ માહિતી આપશે. તમે બધા જાણો છો તેમ, કોમ્પ્યુટર જીવનના બધા ક્ષેત્રોમાં ઉપયોગી અને એક અનિવાર્ય સાધન બની ગયું છે. તે આપણી કાર્યક્ષમતા સુધારવામાં મદદ કરે છે. એટલું જ નહી પણ તે આપણી સામે નવી તકો ખોલે છે.

આ વિષયને મુખ્ય બે ભાગમાં વહેંચવામાં આવ્યો છે. પુસ્તકના પ્રથમ ભાગમાં કોમ્પ્યુટરના મૂળભૂત તત્વોની ચર્ચા કરેલ છે. બીજા ભાગમાં C++ પ્રોગ્રામિંગની ચર્ચા કરેલ છે. આ વિષયને લગતી માહિતી એવી રીતે રજૂ કરાયેલ છે કે જેથી વિદ્યાર્થી તેને જાતે સમજી શકે અને તેને કોઈ બીજા સંદર્ભની જરૂર પડે નહીં.

પુસ્તકની શરૂઆતમાં ડિજિટલ કોમ્પ્યુટરના બંધારણ અને વિવિધ ભાગોની ચર્ચા કરેલ છે. તે મુખ્ય મુદ્દાઓને ધીમે ધીમે અને સરળ રીતે સમજાવે છે. તેને શીખવા તમારે કોમ્પ્યુટરને લગતા અગાઉના જ્ઞાનની જરૂર નથી.

આ પુસ્તકનો બીજો ભાગ C++ પ્રોગ્રામિંગ ભાષાની ચર્ચા કરે છે. આ ભાષા ઓબ્જેક્ટ ઓરીએન્ટેડ હોવાથી પ્રક્રિયાગત ભાષા કરતાં જુદી પડે છે. આના લીધે તે ખુબ જ મોટા અને જટિલ પ્રોગ્રામ માટે અનુકુળ છે. આ પુસ્તકમાં C++ ભાષાના 10 પાઠ આપવામાં આવ્યા છે. જે શરૂઆતથી અગત્યના મુદ્દાઓનો ખ્યાલ આપે છે.

પાઠ 7 અને 8 ની અંદર C++ ની પ્રસ્તાવના અને OPP ના સામાન્ય ખ્યાલ સમજાવ્યા છે. પાઠ 9 અને 10 ની અંદર તમે નિયંત્રણ નિવેદનો અને C++ માં ફંકશનનો ઉપયોગ સમજશો. તમે એરે, સ્ટ્રકચર, ટાઈપડેફ અને એન્યુમરેટેડ ડેટા વિશે પાઠ 11 અને 12 માં શીખશો. પાઠ 13, 14 અને 15 માં તમે કલાસ, ઓબ્જેક્ટ, કંસ્ટ્રકટર્સ, ડિસ્ટ્રકટર, ઈનહેરીટેન્સ અને પોઈન્ટર ભણશો. છેલ્લા પાઠમાં તમે ફાઈલ પ્રોગ્રામિંગ શીખશો.

હું આશા રાખુ કે તમને આ અભ્યાસક્રમમાંથી ઘણુ શીખવા મળશે. સાથે સાથે તમને તે વાંચવાની મજા આવશે અને તમે તેને સમજવા માટે સમક્ષ બનશો. કોઈ પણ વધુ માહિતી માટે અથવા સૂચનો આપવા, અમારો સંપર્ક કરો.

આપ સર્વેને શુભકામનાઓ,

-પાઠયક્રમ સમિતિ

અનુક્રમણિકા

પાઠ નં.	પાઠ	પાન નંબર
	કોમ્પ્યુટરના મૂળભૂત તત્વો	
1.	ડિજિટલ કોમ્પ્યુટરનું વિશ્લેષણ	1
2.	માહિતી પ્રક્રિયાના ખ્યાલ	21
3.	કોમ્પ્યુટર સોફ્ટવેર	33
4.	ઓપરેટિંગ સિસ્ટમ	47
5.	માહિતી સંચાર અને નેટવર્કીંગ	69
6.	ઈન્ટરનેટ અને જાવા પ્રોગ્રામિંગના મૂળભૂત તત્વો	84
	C++	
7.	C++ ની પ્રસ્તાવના	106
8.	OPP ના સામાન્ય ખ્યાલો	129
9.	નિયંત્રણ નિવેદનો	136
10.	કાર્યો	162
11.	એરે	187
12.	સ્ટ્રકચર, ટાઈપડેફ અને એન્યુમરેટેડ ડેટા ટાઈપ	200
13.	કલાસ અને ઓબ્જેકટો સાથે કંસ્ટ્રકટર્સ/ડિસ્ટ્રકટર	211
14.	વારસો - કલાસનું વિસ્તરણ	225
15.	નિર્દેશક	238
16.	ફાઈલો	245

1

ડિજિટલ કોમ્પ્યુટરનું વિશ્લેષણ

1.1 પ્રસ્તાવના

આપણે એ બાબત ધ્યાનમાં રાખવી જોઈએ કોમ્પ્યુટરએ એ પ્રોગ્રામ કરી શકાય તેવું મશીન - ઉપકરણ છે. કોમ્પ્યુટરનાં મુખ્ય બે લક્ષણો છે. (i) *વિશેષ સૂચોનોનો સચોટ પ્રતિભાવ આપે છે.* (ii) *તે પહેલેથી જ રેકોર્ડ કરાયેલ સૂચનો (પ્રોગ્રામ) નો અમલ કરે છે. આધુનિક કોમ્પ્યુટર ઈલેક્ટ્રોનિક અને ડિજિટલ છે. હાર્ડવેર એટલે વિજળીના તાર, ટ્રાન્ઝીસ્ટર્સ, સરકીટ (વિદ્યુત પરિપથ) વગેરે યાંત્રિક ભાગ. જયારે સૂચનો અને માહિતી (ડેટા) એટલે સોફ્ટવેર. સામાન્ય કોમ્પ્યુટરમાં નીચે મુજબના હાર્ડવેર જોવા મળે છે.*

(i) સેન્ટ્રલ પ્રોસેસિંગ યુનિટ(CPU) : *તે કોમ્પ્યુટરનું મગજ છે અને તે જ સૂચનાનો અમલ કરતો ભાગ છે.*

(ii) મેમરી : *તે કોમ્પ્યુટરનો* Data *નો સંગ્રહ કરે છે.* Data *અને* Program *ને ટૂંકા સમય માટે સંગ્રહ કરે છે.*

(iii) ઈનપુટ ડિવાઈસ : *સામાન્ય રીતે કી - બોર્ડ અને માઉસના ઉપયોગથી કોમ્પ્યુટરમાંના* Data *તથા* Programs *ને દાખલ કરી શકાય છે.*

(iv) આઉટપુટ ડિવાઈસ : *મોનીટરની સ્ક્રીન,પ્રિન્ટર વગેરેના ઉપયોગથી આપણે કોમ્પ્યુટર દ્વારા થયેલી ક્રિયાઓ કે કાર્યોને જોઈ શકીએ છીએ.*

(v) માસ સ્ટોરેજ ડિવાઈસ : *જેનાથી કોપ્યુટરમાં મોટી માત્રામાં કાયમી રીતે ડેટાનો સંગ્રહ કરી શકાય છે. ડિસ્ક ડ્રાઈવ અને ટેપ ડ્રાઈવ એ સામાન્ય માસ સ્ટોરેજ ડિવાઈસ છે.*

આ સિવાય પણ ઘણા ઘટકો છે જે કોમ્પ્યુટરને અસરકારક રીતે કામ કરવામાં મદદ કરે છે.

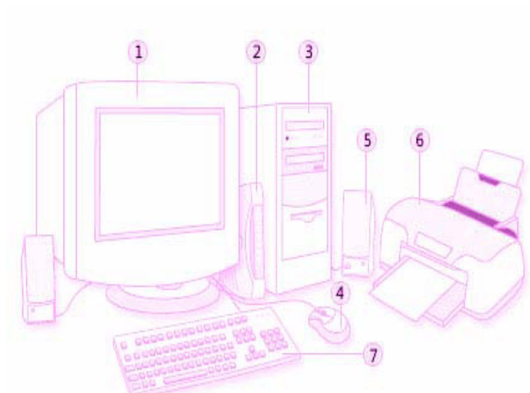
1.2 હેતુઓ

આ પ્રકરણને અંતે તમે નીચેની બાબતો માટે સક્ષમ હશો. :

- *કોમ્પ્યુટરના મૂળ ભાગ અને તેમના કાર્યને ઓળખી શકશો.*
- *કોમ્પ્યુટરના વિવિધ ભાગોની સમજૂતી આપી શકશો.*
- *કાર્ય પૂર્ણ કરવા આ બધા ભાગ કઈ રીતે કાર્ય કરે છે તે સમજી શકશો.*

1.3 કોમ્પ્યુટરના ભાગો અને કાર્યો

કોમ્પ્યૂટર દ્વારા કાર્ય પુરા પાડવા હાર્ડવેર અને સોફ્ટવેર બંને જરૂરી છે. હાર્ડવેર એટલે મિકેનિકલ અને ઈલેક્ટ્રોનિક ઉપકરણ જેને આપણે જોઈ શકીએ અને સ્પર્શ કરી શકીએ. જ્યારે સોફ્ટવેર એટલે પ્રોગ્રામ્સ, ઓપરેટિંગ સિસ્ટમ અને ડેટા જે મેમરી અને સ્ટોરેજ ડિવાઈસમાં સંગ્રહિત હોય છે.



(1) *મોનીટર* (3) *સીસ્ટમ યુનિટ* (5) *સ્પીકર* (7) *કીબોર્ડ*

(2) *મોડેમ* (4) *માઉસ* (6) *પ્રિન્ટર*

આકૃતિ- 1.1 *કોમ્પ્યૂટરના ભાગો અને કાર્યો*

કોમ્પ્યૂટર મુખ્યત્વે નીચે મુજબનાં ચાર કાર્ય કરે છે.

- **ઈનપુટ મેળવે છે** - *કી-બોર્ડ, માઉસ, સ્કેનર જેવી ઈનપુટ ડીવાઈસીસ દ્વારા ડેટા/ માહિતી મેળવે છે.*
- **માહિતીનું પૃથ્થકરણ** - *ડેટા/માહિતી પર ગાણિતીક અથવા તાર્કિક પ્રક્રિયા દ્વારા પૃથ્થકરણ કરે છે.*
- **આઉટપુટ આપે છે** - *મોનીટરની સ્ક્રીન, પ્રિન્ટર જેવી આઉટપુટ ડિવાઈસ દ્વારા ડેટા/માહિતી ને પ્રદર્શિત કરે છે.*
- **માહિતી સંગ્રહ** - *હાર્ડ ડીસ્ક, ફ્લોપી ડિસ્ક, CD વગેરે જેવી સ્ટોરેજ ડિવાઈસમાં માહિતીનો સંગ્રહ કરે છે.*

આ ચાર કોમ્પ્યૂટરનાં પાયાનાં કાર્યો છે. હાર્ડવેર આ ચારમાંથી કોઈપણ એક કાર્ય કરે છે.

કોમ્પ્યુટર હાર્ડવેરના બે પ્રકાર છે. : પ્રોસેસિંગ હાર્ડવેર, જેમાં સેન્ટ્રલ પ્રોસેસિંગ યુનિટ (CPU) અને પેરિફેરલ ડિવાઈસિસ સમાવિષ્ટ છે. CPU તેના નામ મુજબ તેમાં ડેટાનું પ્રોસેસિંગ થાય છે. જ્યારે પેરિફેરલ ડિવાઈસ વાપરીને માણસો CPU સાથે કામ લઈ શકે છે. આમ બધા સાથે મળીને કોમ્પ્યુટરનાં વિવિધ કાર્યો કરી શકે છે. આકૃતિ 1.1 માં કોમ્પ્યુટરનાં વિવિધ ભાગો દર્શાવેલ છે.

સેન્ટ્રલ પ્રોસેસિંગ યુનિટ (CPU)

કોમ્પ્યુટરનો જે ભાગ પ્રોગ્રામ સંબંધી સૂચનોનો અમલ કરે છે તેને પ્રોસેસર અથવા સેન્ટ્રલ પ્રોસેસિંગ યુનિટ (CPU) કહે છે. માઈક્રોકોમ્પ્યુટરમાં CPU એ એક ઈલેક્ટ્રોનિક ભાગનું જ બનેલું હોય છે. આ સિસ્ટમમાં સરકીટ બોર્ડ, મેમરી ચીપ, વિવિધ પોર્ટ અને અન્ય ભાગ સમાવિષ્ટ છે. માઈક્રોકોમ્પ્યુટરમાં ડીસ્ક ડ્રાઈવ, હાર્ડ ડીસ્ક વગેરે હોય છે પણ તેમને CPU થી અલગ ગણવામાં આવે છે.

CPU ના બે ભાગ છે. કન્ટ્રોલ યુનિટ (CU) અને એરિથમેટિક લોજિક યુનિટ (ALU). માઈક્રો કોમ્પ્યુટરમાં બન્ને માઈક્રોપ્રોસેસર ચીપમાં જ હોય છે.

કન્ટ્રોલ યુનિટ (CU)

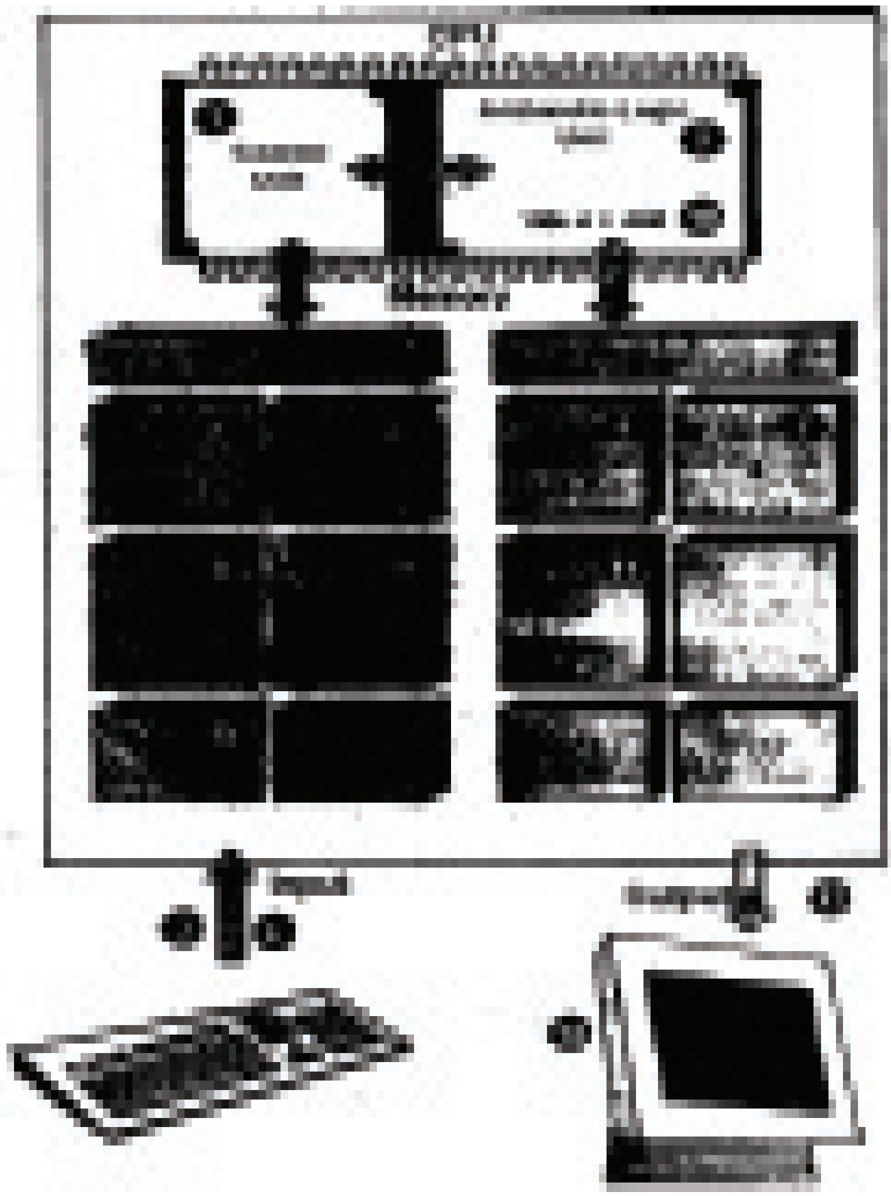
કન્ટ્રોલ યુનિટ અન્ય ભાગને પ્રોગ્રામના ઈન્સ્ટ્રક્શનને અમલમાં મૂકવા સૂચવે છે. તે ઈલેક્ટ્રોનિક સિગ્નલ (વિદ્યુત સંકેતો) ને મેમરી વચ્ચેની આપે - લે માં સૂચના આપે છે. જેમાં ટૂંક સમય માટે ડેટા, સૂચન અને વર્ગીકૃત માહિતીનો સંગ્રહ થાય છે. તથા ALU વચ્ચે પણ સંકેતોને પ્રેરિત કરે છે.

એરિથમેટિક લોજિક યુનિટ (ALU)

એરિથમેટિક લોજિક યુનિટ જે ટૂંકમાં ALU તરીકે ઓળખાય છે, તે બે રીતે કાર્ય કરે છે. ગાણિતીક અને તાર્કીક. ગાણિતીક ક્રિયાઓમાં સરવાળા, બાદબાકી, ગુણાકાર અને ભાગાકાર જેવા કાર્યો સમાવિષ્ટ છે. જ્યારે તાર્કીક ક્રિયામાં સામાન્ય રીતે તુલના જેવા કાર્યો કરવામાં આવે છે. જેમાં Data સરખો છે, નાનો છે કે મોટો છે તેવી તુલના કરવામાં આવે છે.

મેમરી (Memory)

મેમરી (Memory) જે પ્રાઈમરી અથવા મેઈન સ્ટોરેજ તરીકે ઓળખાય છે, જે માઈક્રો કોમ્પ્યુટરનો ભાગ છે, તે Data નું પ્રોસેસિંગ કરે છે, જેથી ડેટા અથવા ઈન્ફર્મેશનનું વિશ્લેષણ થઈ શકે.



આકૃતિ 1.2 CPU અને મેમરીનું કાર્ય

આ મેમરીમાં માહિતીનો સંગ્રહ થોડા સમય માટે જ થાય છે. એટલે કે માઈક્રો કોમ્પ્યુટર બંધ કરતા તે માહિતી સંગ્રહિત રહેતી નથી. જુદા જુદા પ્રકારનાં કોમ્પ્યુટર્સમાં માહિતી અને પ્રોગ્રામ સંગ્રહ કરવાની ક્ષમતા જુદી જુદી હોય છે. IBM *નું અસલ કોમ્પ્યુટર લગભગ* Data/ *ઈન્સ્ટ્રકશનનાં લગભગ* 6,40,000 *અક્ષરો જ સંગ્રહિત કરી શકે છે. જ્યારે આધુનિક કોમ્પ્યુટર્સ તો કરોડો - અબજો અક્ષરો મેમરીમાં સંગ્રહીત કરી શકે છે.*

રજીસ્ટર્સ (Registers)

કોમ્પ્યુટર્સમાં ઘણા બધા સંગ્રહસ્થાનો હોય છે જે રજીસ્ટર્સ તરીકે ઓળખાય છે. જે કન્ટ્રોલ યુનિટ અને ALU *માં હોય છે અને કાર્યને વધુ અસરકારક બનાવે છે. રજીસ્ટર્સ એ ટૂંક સમયનાં હાઈ-સ્પીડ સંગ્રહસ્થાનો છે. જે* Data *અને* Instructions *ને ટૂંક સમય માટે પ્રોસેસિંગ દરમિયાન સંગ્રહીત કરે છે. તેઓ મેમરી નથી પણ કન્ટ્રોલ યુનિટ અને* ALU *નાં ભાગરૂપે છે. તેથી જ તેઓ માહિતી મેમરી કરતા પણ વધુ ઝડપથી સાચવી શકે છે.*

એડ્રેસિસ (Addresses)

ડેટા અથવા ઈન્સ્ટ્રક્શનનાં કેરેક્ટરનું મેઈન મેમરીમાં સ્થાન નિર્ધારીત કરવા કોમ્પ્યુટર તેમને જે સ્થાને સંગ્રહિત કરે છે તેમને એડ્રેસિસ કહે છે. દરેક એડ્રેસને યુનિક નંબર ફાળવવામાં આવે છે. આ એડ્રેસિસને પોસ્ટ ઓફિસનાં સરનામાં સાથે સરખાવી શકાય. જેમાં સંખ્યા તે જ રહે છે પણ Data *બદલાતો રહે છે.*

CPU *અને* Memory *કઈ રીતે કાર્ય કરે છે ?* (How the CPU and Memory Work)

CPU *ની કાર્યપદ્ધતિ આકૃતિ* 1.2 *માં દર્શાવવામાં આવી છે. બે સંખ્યાઓનાં ગુણાકાર માટેના પગલાં આકૃતિમાં દર્શાવેલ છે અને નીચે સમજાવ્યાં છે.*

1. *કન્ટ્રોલ યુનિટને જાણ થાય છે કે મેમરીમાં પ્રોગ્રામ મૂકવામાં આવ્યો છે અને તે પ્રોગ્રામનું પગલું અમલમાં મુકે છે.*
2. *પ્રોગ્રામ વપરાશકર્તાને જણાવે છે. "પ્રથમ સંખ્યા નાંખો."*
3. *વપરાશકર્તા કી-બોર્ડ વડે* 100 *ટાઈપ કરે છે. જેના વિદ્યુત સંકેતો* CPU *ને પહોંચે છે.*
4. *કન્ટ્રોલ યુનિટ તે સંકેતોને ગ્રહણ કરી મેમરીનાં એડ્રેસને મોકલે છે. એડ્રેસ -7*
5. *ઉપર મુજબનાં સૂચનોનાં અમલ પછી વપરાશકર્તાને ફરી બીજી સંખ્યા નાખવા જણાવવામાં આવે છે.*
6. *વપરાશકર્તા કી-બોર્ડ વડે* 4 *ટાઈપ કરે છે. જેના વિદ્યુત સંકેતો* CPU *ને પહોંચે છે.*
7. *કન્ટ્રોલ યુનિટ તે સંકેતો ગ્રહણ કરી મેમરીના એડ્રેસ-* 8 *ને મોકલે છે.*
8. *પ્રોગ્રામમાં ત્યારપછીનાં સૂચનનો અમલ થાય છે. પહેલી અને બીજી સંખ્યાનો ગુણાકાર કરો.*
9. *આ ક્રિયા માટે કન્ટ્રોલ યુનિટ* ALU *ને તે બે સંખ્યાઓનો ગુણાકાર કરવાનું જણાવે છે. ત્યારબાદ કન્ટ્રોલ યુનિટ* ALU *ને એડ્રેસ* 7 (100) *અને એડ્રેસ* 8 (4) *ની કોપી મોકલે*

છે.

10. ALU ગુણાકાર કરે છે.100 x 4 = 400

11. કન્ટ્રોલ યુનિટ ગુણાકારનાં પરિણામ (400) ની કોપી મેમરીને એડ્રેસ 9 પર સંગ્રહ કરવા મોકલે છે.

12. ત્યારબાદનાં પ્રોગ્રામનો અમલ થાય છે. પરિણામ દર્શાવો.

13. આ સૂચનનાં અમલ માટે કન્ટ્રોલ યુનિટ એડ્રેસ 9 (400) ને મોનિટરની સ્કીન પર પ્રદર્શિત કરે છે.

14. મોનિટર 400 દર્શાવે છે.

15. અંતિમ સૂચનનો અમલ થાય છે. "End" પ્રોગ્રામ સમાપ્ત.

આકૃતિ 1.2 એ મૂળભૂત પ્રક્રિયાનું સરળ નિરૂપણ માત્ર છે. CPU અને Memory ની સાચી પ્રક્રિયા તો ખૂબ જ જટીલ છે. વાસ્તવમાં તો તેમાં અનેક એડ્રેસ હોય છે. હજારો થી કરોડો એડ્રેસ હોય છે. વળી એડ્રેસ વિદ્યુત તરંગોના રૂપમાં હોય છે. જે કોમ્પ્યુટર દ્વારા જ અર્થગ્રહણ કરી શકાય.

હવે આપણે કોમ્પ્યુટરનાં વિવિધ ભાગો વિશે ચર્ચા કરીશું.

1.4 ઈનપુટ ડિવાઈસ

ઈનપુટ ડિવાઈસ મશીન-રીડેબલ રૂપમાં Data મોકલે છે. જોકે મુખ્ય ડિવાઈસ તરીકે કી-બોર્ડમાં ઉપયોગ થાય છે તેમ છતા ઓપ્ટિકલ કેરેક્ટર રીકગ્નીશન (OCR), મેગ્નેટિક ઈન્ક કેરેક્ટર રીકગ્નીશન (MICR), માર્ક સેન્સ રીડર, વગેરેનો પણ ઈનપુટ ડિવાઈસ તરીકે ઉપયોગ કરી શકાય છે.

1.4.1 કી-બોર્ડ

કી-બોર્ડ એટલે અમુક વધારાની કી સાથેનું સ્ટાન્ડર્ડ ટાઈપરાઈટરનું કી-બોર્ડ. QUERTY ને પાયાનો લે-આઉટ તરીકે જાળવી રાખવામાં આવ્યો છે જેથી તાલીમ પામેલ ટાઈપિસ્ટને કેરેક્ટર ટાઈપ કરવાનું સુગમ બની રહે પ્રોગ્રામ લોડ કરવો, text ને એડીટ કરવી વગેરે જેવા વિશેષ કાર્યો કરવા વધારાની Keys આપવામાં આવી છે. તે function keys તરીકે ઓળખાય છે. આ keys અલગ અલગ સિસ્ટમમાં અલગ - અલગ હોઈ શકે છે.

1.4.2. ઓપ્ટિકલ કેરેક્ટર રીકગ્નીશન (OCR)

ટૂંકમાં OCR તરીકે ઓળખાય છે. જે પેપર પરની text ને read કરી image માં પરિવર્તિત કરે છે જેથી કોમ્પ્યુટર ગ્રહણ કરી શકે. OCR સિસ્ટમથી પુસ્તક કે સામયિકના આર્ટિકલને કોમ્પ્યુટરમાં ઈલેક્ટ્રોનીક રૂપે સાચવી શકાય છે.

1.4.3 મેગ્નેટિક ઈન્ક કેરેક્ટર રિકગ્નીશન (MICR)

MICR મેગ્નેટિક પદાર્થની બનેલી ઈન્કથી છપાયેલા અક્ષરોને ઓળખી શકે છે. આ ડિવાઈસ

મોટેભાગે બેન્કમાં જોવા મળે છે. MICR અમુક પ્રકારના અક્ષરો જ ઓળખી શકતુ હોવાથી, તે અક્ષરો યોગ્ય રૂપે બનેલા હોવા જોઈએ.

1.4.4 ઓપ્ટિકલ માર્ક રિકગ્નીશન (OMR)

ઓપ્ટિકલ માર્ક રીકગ્નીશનને માર્ક સેન્સ રીડર પણ કહેવાય છે. આ ટેકનોલોજી દ્વારા પેન્સિલનાં માર્ક (નિશાન)ને OMR ડિવાઈસ ઓળખી કાઢે છે. OMR નો કૌશલ્ય કસોટી વગેરેમાં ઉપયોગ થાય છે.

1.4.5 બાર કોડ રીડર

આ ડિવાઈસ મુખ્યત્વે સુપર માર્કેટ, બુક સ્ટોલ વગેરે જગ્યાએ જોવા મળે છે. બાર કોડ રીડર વસ્તુ કે તેના પેકેટ ઉપર છપાયેલા બારકોડ અથવા ઉભા પટ્ટાને વાંચી શકે છે. સુપર માર્કેટમાં યુનિવર્સલ પ્રોડક્ટ કોડના આધારે બાર કોડ રાખવામાં આવે છે. જેની મદદથી સુપર માર્કેટનાં કોમ્પ્યુટર જે તે વસ્તુની વિગત અને કિંમત ચેક કરે છે. કોમ્પ્યુટર ત્વરિતપણે તે વિશેની વિગત બતાવે છે.

1.4.6 ડિજીટાઈઝીંગ ટેબલેટ

આ એક એવી ડિવાઈસ છે જેના વડે આપ ચિત્રો અને રેખાંકનોને કોમ્પ્યુટરમાં મૂકી શકો છો. ડિજિટાઈઝીંગ ટેબલેટમાં ઈલેક્ટ્રોનિક ટેબલેટ અને કર્સર અથવા પેન સમાવિષ્ટ છે. આ કર્સર (અથવા Puck) જે માઉસ જેવું જ કાર્ય કરે છે પણ તેમાં બિંદુ અથવા ચોકક્સ સ્થાન માટે + નિશાની હોય છે. તેમાં લગભગ 16 જેટલા બટન હોઈ શકે છે.પેન (અથવા Stylus) જે બોલપોઈન્ટ પેન જેવી જ છે પણ તેનો પોઈન્ટ પર સહીના બદલે ઈલેક્ટ્રોનિક હેડ હોય છે. ટેબલટ પર ઈલેક્ટ્રોનિક ક્ષેત્ર રચાયેલું હોય છે, જેથી તે કર્સર અથવા પેનના હલનચલનની નોંધ કરી શકે છે. આ હલનચલન ડિજીટલરૂપે કોમ્પ્યુટરમાં ગ્રહણ થાય છે. ડિજીટાઈઝિંગ ટેબલેટને ડિજીટાઈઝ ગ્રાફીક ટેબલેટ, ટચ ટેબ્લેટ અથવા સિમ્પલી ટેબલેટ પણ કહેવામાં આવે છે.

1.4.7 સ્કેનર

સ્કેનર એ એવી ઈનપુટ ડિવાઈસ છે જે પેપર પર છપાયેલી ટેક્સ્ટ અને ચિત્રો વાંચી કોમ્પ્યુટર ઉપયોગ કરી શકે તેવા રૂપમાં પરિવર્તિત કરે છે. સ્કેનર તેને ડિજીટલ રૂપે ફેરવી નાંખે છે અને તેમને 0 અને 1 એમ બે બોક્સનાં રૂપે પરિવર્તિત કરી નાંખે છે. જે Bit ના Matrix માં રૂપાંતર પામે છે, તે Bit Map તરીકે ઓળખાય છે, જે ફાઈલમાં Store કરી શકાય છે, જેને કોમ્પ્યુટર દ્વારા મોનીટર સ્ક્રીન પર પ્રદર્શિત કરી શકાય છે. આ ઓપ્ટિકલ સ્કેનર ચિત્રોમાંની ટેક્સ્ટ ગ્રહણ કરી શકાતી નથી. આ text ને edit કરવા OCR દ્વારા પરિવર્તિત કરી ASCII કેરેકટર્સ રૂપે translate કરવી. હાલમાં મોટા ભાગના ઓપ્ટિકલ સ્કેનર OCR સાથે જ આવે છે.

1.4.8 માઉસ

માઉસ એ એવું ઉપકરણ છે જેના દ્વારા મોનીટરની સ્ક્રીન પર દેખાતાં કર્સર કે પોઈન્ટર દ્વારા નિયમન કરી શકાય છે. એ એક નાનું ઉપકરણ છે જેને હાથ દ્વારા સપાટ સપાટી પર ફેરવી શકાય છે. માઉસને જે તે દિશામાં ખસેડવાથી સ્ક્રિન પર પર પોઈન્ટર ખસે છે. એક અથવા વધુમાં વધુ ત્રણ બટન હોય છે. જેમના કાર્ય પ્રોગ્રામ મુજબ જુદા - જુદા હોય છે.

1.4.9 લાઈટ પેન

લાઈટ પેન પણ એક ઈનપુટ ડિવાઈસ છે જેનો ઉપયોગ લાઈટ સેન્સિટીવ ડિટેકટર દ્વારા સ્ક્રીન પરની Object Select કરવા થાય છે.

14.10 સ્પ્રીચ ઈનપુટ ડિવાઈસ

આ સ્પીચ/વોઈસ ઈનપુટ ડિવાઈસ અવાજને ડિજિટલ રૂપે પરિવર્તીત કરે છે. આ ઈનપુટ ડિવાઈસ કાર્ય કરતી કરવા વોઈસ રીકગ્નીશન સિસ્ટમવાળું સોફટવેર નાંખવું પડે છે. આ સિસ્ટમથી વોઈસ કમાન્ડ દ્વારા કમ્પ્યુટરનું સંચાલન કરી શકાય છે.

પાઠગત પ્રશ્નો 1.1

---

1. કોમ્પ્યુટર પર થતા ચાર મૂળભૂત કાર્યો કયા કયા છે ?
2. સ્કેનર શું છે ? ટૂંકમાં સમજાવો.
3. સાચો વિકલ્પ પસંદ કરો.

(અ) ગાણિતિક અને તાર્કિક ક્રિયાઓ મારા દ્વારા થાય છે.

(i) ALU (ii) એડિટર (iii) સંગ્રહ (iv) આઉટપુટ

(બ) ALU અને CU સંયુક્તે પણે ઓળખાય છે.

(i) RAM (ii) ROM (iii) CPU (iv) એક પણ નહીં.

(ક) જરૂરી માહિતીને પરિણામ સ્વરૂપ મેળવવાની પ્રક્રિયાને શું કહે છે ?

(i) આઉટપુટ (ii) ઈનપુટ (iii) પ્રોસેસિંગ (iv) સંગ્રાહક

1.5 મેમરી યુનિટ

કોમ્પ્યુટર મેમરીના બે પ્રકાર છે. પ્રાયમરી અને સેકન્ડરી. પ્રાયમરી મેમરી પ્રોસેસીંગ યુનિટમાં રહે છે. RAM એ પ્રાયમરી મેમરીનું ઉદાહરણ છે. કોમ્પ્યુટરને બંધ થતા પ્રાઈમરી મેમરી જતી રહે છે. સેકન્ડરી મેમરી કરતાં પ્રાઈમરી મેમરીને ઝડપથી મેળવી શકાય છે. ફલોપી ડિસ્ક, મેગ્નેટીક ડિસ્ક વગેરે જેવા સેકન્ડરી મેમરીના ઉપકરણો પ્રોસેસિંગ યુનિટથી બહાર રહેલા હોય

નોંધ

છે. સેકન્ડરી મેમરી કરતાં પ્રાઈમરી મેમરી વધુ મોંઘી પડે છે.

કોમ્પ્યુટર મેમરીનો સંગ્રહ બે વસ્તુઓ માટે કરવામાં આવે છે. 1) પ્રોગ્રામ સંબંધી સૂચનાઓનો અમલ કરવો. 2) ડેટા. જયારે કોમ્પ્યુટર કાર્ય કરતું હોય છે ત્યારે તેનો ડેટા પ્રાઈમરી મેમરીમાં સ્ટોર થાય છે. આ ડેટા કી બોર્ડ કે ફ્લોપી ડિસ્ક જેવી સેકન્ડરી સ્ટોરેજ ડિવાઈસમાંથી મેળવેલો હોય છે.

પ્રોગ્રામ અથવા સૂચનો પ્રાઈમરી મેમરીમાં રહે છે. જેથી તે સૂચનોનો કોમ્પ્યુટર ત્વરીતપણે અમલ કરી શકે. દા.ત. જેમ કે તમે કોઈ રેલવે રીઝર્વેશન કાઉન્ટર પર ટીકીટ બુક કરાવો ત્યારે કોમ્પ્યુટર અમુક નિયમિત સ્ટેપ્સને અનુસરવાનું રહે છે. બેઠકની પ્રાપ્યતા ચકાસવી, ભાડું ગણવું, નાણાં ચૂકવણી, આરક્ષણ કરવું તથા ટિકિટની પ્રિન્ટ કાઢવી. આ પ્રોગ્રામ કોમ્પ્યુટરની મેમરીમાં સ્ટોર થયેલા હોય છે અને દરેક request વખતે તેમનો અમલ કરવામાં આવે છે.

પરંતુ કોમ્પ્યુટરની આંતરિક પ્રક્રિયા સ્ક્રીન પર દેખાય તેના કરતાં જુદી જ હોય છે. કોમ્પ્યુટરની મેમરીમાં પ્રોગ્રામ અને ડેટા બન્નેનો સંગ્રહ બાઈનરી રૂપે થાય છે. આપ દશાંશ પધ્ધતિ વિશે તો માહિતગાર હશો જ, જેમાં 1 થી 9 અને 0 નો સમાવેશ થાય છે. બાઈનરી સિસ્ટમમાં માત્ર 0 અને 1 જ સમાવિષ્ટ છે તેમને બિટ તરીકે ઓળખવામાં આવે છે. આપણે દશાંશ પદ્ધતિમાં 0 થી 9 અંકોનો ઉપયોગ કરીએ છીએ. પરંતુ કોમ્પ્યુટર માત્ર બે અંકો 0 અને 1 જ ગ્રહણ કરે છે તેનું કારણ એ છે કે કોમ્પ્યુટરનાં અંદરના ઈન્ટિગ્રેટેડ સર્કીટ (IC) નાં અંકો સ્વીચ તરીકે લેવામાં આવે છે. જે ON અને OFF કરી શકાય છે. જો સ્વીચ ઓન હોય તો 1 અને બંધ હોય તો 0 (શૂન્ય) તરીકે ગણાશે. જુદી - જુદી રીતે સ્વીચની સંખ્યા 110101.10 જેવો મેસેજ આપશે. આમ કોમ્પ્યુટર માત્ર 0 અને 1 રૂપે જ ઈનપુટ લે છે અને તે જ રૂપે આઉટપુટ આપે છે. શું તમને આ વિચિત્ર નથી લાગતું કે કોમ્પ્યુટર 0 અને 1 રૂપે આઉટપુટ આપે ? પણ દરેક બાઈનરી નંબર દશાંશ પધ્ધતિમાં પરિવર્તિત થાય છે. દા.ત. 1010 એટલે 10 કોમ્પ્યુટર તમારી પાસેથી ડેસીમલમાં ડેટા લઈને બાઈનરી સિસ્ટમમાં પરિવર્તિત કરે છે.પ્રોસેસ બાદ ફરી બાઈનરીને ડેસિમલમાં રૂપાંતર કરી અને અંતે ડેસિમલમાં આઉટપુટ આપે છે.

કોમ્પ્યુટરમાં પ્રાઈમરી મેમરી IC (ઈન્ટિગ્રેટેડ મેમરી) રૂપે હોય છે. આ સર્કીટને રેન્ડમ એક્સેસ મેમરી (RAM) તરીકે ઓળખાય છે. RAM નાં દરેક લોકેશન પર 1 બાઈટ માહિતીનો સંગ્રહ થાય છે. (1 બાઈટ = 8 બિટ) બિટ એટલે બાઈનરી ડિઝીટનું સંક્ષિપ્ત રૂપ. જે બાઈનરી માહિતીનો એક ભાગ છે. જે 0 અને 1 હોઈ શકે. RAM વિશે વધુ માહિતી આગળ જોઈશું. પ્રાઈમરી અથવા આંતરિક સંગ્રહ એકમો Cell તરીકે ઓળખાતાં ઘણાં ભાગોનું બનેલું છે. આ દરેક Cell નિશ્ચિત સંખ્યામાં bit નો સંગ્રહ કરી શકે છે જેને વર્ડ લેન્થ તરીકે ઓળખાય છે.

દરેક Cell નો અલગ નંબર આપવામાં આવે છે જે તેનું address કહેવાય છે અને તેનો ઉપયોગ Cell ની ઓળખ માટે થાય છે. આ 0 થી શરૂ થઈને N-1 સુધીના રહે છે. મેમરી એટલે ઘણા બધાં Address નાં ઘણા બધા ખાનાવાળું મોટું કેબિનેટ દરેક ખાનામાં એક શબ્દ હોય છે અને તેનું Address બહાર લખેલ હોય છે.

1.5.1 પ્રાઈમરી મેમરીની ક્ષમતા

મેમરીનાં દરેક Cell માં એક અક્ષર (કેરેકટર) અથવા 1 બાઈટ ડેટા હોય છે. તેથી મેમરીની ક્ષમતા બાઈટ અથવા વર્ડમાં ગણવામાં આવે છે. આમ 64 કિલોબાઈટ એટલે 64 x 1024 = 32,768 બાઈટ (1કિલોબાઈટ = 1024 બાઈટ) નાના સિસ્ટમમાં મેમરીમાં થોડાક કિલોબાઈટ હોય છે તથા સુપર કોમ્પ્યુટરમાં હજારો કિલોબાઈટની હોય છે. આપના કોમ્પ્યુટરમાં આ મેમરી 8 MB, 16 MB, 128 MB,.......... (MB = Million Bytes (દસ લાખ બાઈટ)) હોય છે.નીચે કોમ્પ્યુટરની મેમરીને લગતા કેટલાંકની માહિતી આપી છે.

(અ)રેન્ડમ એક્સેસ મેમરી (RAM) : પ્રાઈમરી મેમરી એટલે રેન્ડમ એકસેસ મેમરી. આ મેમરી સીધી સ્ટોર કરવા અથવા પુનઃ મેળવવા રેન્ડમ પ્રોસેસથી મેળવાય છે. તેના દ્વારા પ્રથમ Addressના જેટલી જ ઝડપથી કોઈ પણ Address મેળવી શકાય છે તેને read/write મેમરી પણ કહે છે. કોમ્પ્યુટર બંધ થતા તરત જ રહેલો ડેટા જતો રહે છે, વીજળીનો પ્રવાહ બંધ થતા જે ડેટા જતો રહે છે, તેવા ડેટાને વોલેટાઈલ (Volatile) મેમરી કહે છે.તેથી આપણે કહી શકીએ કે RAM એ વોલેટાઈલ મેમરી છે.

(બ) રીડ ઓન્લી મેમરી (ROM) : મેમરીનો અન્ય એક પ્રકાર છે જે રીડ ઓન્લી મેમરી (ROM) તરીકે ઓળખાય છે. તે પણ કોમ્પ્યુટરમાંની ઈન્ટિગ્રેટેડ સરકીટ (IC) છે. (ROM) પરનો પ્રોગ્રામ અને ડેટા કાયમી છે. (ROM) પર કોમ્પ્યુટર સંચાલન માટેના અમુક પ્રોગ્રામ સ્ટોર થયેલા હોય છે. તેના પરનો data CPU વાંચી કરી શકે છે પણ તેને બદલી શકતો નથી. કોમ્પ્યુટરનો ઈનપુટ/આઉટપુટ પ્રોગ્રામ ROM પર સંગ્રહિત થયેલો હોય છે જે કોમ્પ્યુટર ચાલુ કરતાં તેની સાથે જોડેલા સાધનો/ડિવાઈસને સાંકળે છે જે મેમરી વિદ્યુતપ્રવાહ બંધ થતા પણ ઉડી જતી નથી, તેવી મેમરીને Non - Volatile મેમરી કહે છે. ROM એ Non- Volatile મેમરી છે.

(ક) PROM : તે કોમ્પ્યુટરમાંની પ્રાયમરી મેમરીનો અન્ય એક પ્રકાર છે. પ્રોગ્રામેબ્લ રીડ ઓન્લી મેમરી (PROM) ROM પર Store થયેલા પ્રોગ્રામને બદલી શકાતા નથી કે કાઢી શકાતા નથી. પરંતુ તેમને PROM પર Store કરી શકાય છે. એક વાર પ્રોગ્રામ મુક્યા પછી તેમને બદલી શકાતા નથી અને વિદ્યુત પ્રવાહ જતા પણ તેમના તેમ રહે છે. આમ PROM કે ROM પરના પ્રોગ્રામ અથવા ઈન્સ્ટ્રકશન બદલી શકાતા નથી કે કાઢી શકાતા નથી.

(ડ) EPROM : EPROM એટલે ઈરેઝેબલ પ્રોગ્રામેબલ રીડ ઓન્લી મેમરી. જેનો ઉપયોગ PROM અને ROM ના વિકલ્પરૂપે થાય છે. EPROM ચીપને પ્રોગ્રામ

કરી શકાય છે તથા તેના પરની માહિતી કાઢી શકાય છે. EPROM પરની માહિતી અલ્ટ્રાવાયલેટ (પારજાંબલી) પ્રકાશ દ્વારા કાઢી શકાય છે. આ મેમરીને વિશેષ પ્રોગ્રામિંગ દ્વારા પુનઃ પ્રોગ્રામ કરી શકાય છે. EPROM જ્યારે ચાલુ હોય ત્યારે માહિતી માત્ર વાંચી શકાય છે.

(ઈ) EAPROM: EAPROM એટલે ઈલેક્ટ્રીકલી અલ્ટરેબલ પ્રોગ્રામેબલ રીડ ઓન્લી મેમરી તે મૂળભૂત રીતે PROM છે પરંતુ EAPROM પરની માહિતી વિદ્યુત તરંગો દ્વારા બદલી શકાય છે. પૂર્ણ માહિતીને કાઢવાની જરૂરિયાત રહેતી નથી.

(ઉ) Cache મેમરી : મેઈન મેમરી કરતાં CPU ની ઝડપ અત્યાધિક હોય છે. તેથી મેઈન મેમરીના કારણે CPU ની ઝડપ ઘટી જાય છે. આ સમસ્યાનાં નિવારણ રૂપે CPU અને મેઈન મેમરી વચ્ચે એક નાની મેમરી ચીપ જોડવામાં આવે છે. જેને CPU જેટલી જ ઝડપથી access કરી શકાય છે. તેને CACHE મેમરી કહે છે. CACHE મેમરીને RAM કરતાં વધુ ઝડપથી access કરી શકાય છે. તેનો ઉપયોગ ચાલુ પ્રોગ્રામની માહિતી અથવા CPU દ્વારા મોટેભાગે ઉપયોગમાં લેવાતી માહિતીને Store કરવા થાય છે. વધુ ક્ષમતાવાળી CACHE મેમરી મોંઘી પડે છે. તેથી તેની સાઈઝ નાની રાખવામાં આવે છે.

(ઊ) રજીસ્ટર્સ (Registers) : CPUમાં ડેટા અને ઈન્સ્ટ્રકશનની પ્રોસેસ ખૂબ જ ઝડપથી થાય છે. વળી કોમ્પ્યુટરનાં વિવિધ એકમોમાં પણ ડેટાનો વિનિમય થાય છે. Data વધુ ઝડપથી મોકલવો ખૂબ જ જરૂરી છે. તેથી કોમ્પ્યુટર રજીસ્ટર તરીકે ઓળખાતા અસંખ્ય વિશેષ મેમરી યુનિટનો ઉપયોગ કરે છે. જો કે તે મેઈન મેમરીનો ભાગ નથી તે કામચલાઉ ધોરણે ડેટા કે માહિતી સ્ટોર કરે છે અને કન્ટ્રોલ યુનિટના આદેશ મુજબ આગળ પહોંચાડે છે.

પાઠગત પ્રશ્નો 1.2.

1. Bit અને Byte વચ્ચેનો તફાવત સ્પષ્ટ કરો.
2. Volatile અને Non- Volatile મેમરીની વ્યાખ્યા આપો.
3. ખરાં ખોટાની નિશાની કરો.

(અ) કોમ્પ્યુટર મેમરીના બે પ્રકાર છે. પ્રાઈમરી અને સેકન્ડરી

(બ) કોમ્પ્યુટર દશાંશ સિસ્ટમ પણ ગ્રહણ કરે છે.

(ક) RAM પરના પ્રોગ્રામ અને માહિતી કાયમી છે.

(ડ) PROM સેકન્ડરી મેમરી છે.

(ઈ) વીજપ્રવાહ જતો રહેતા પણ રહેતી મેમરીને Non- Volatile મેમરી કહે છે.

1.6 ગૌણ સંગ્રહ

આપણે સમજવા કે પ્રાથમિક (Primary) મેમરી અથવા મુખ્ય (Main) મેમરીની ઝડપ ઓપરેટિંગ CPU ની ઝડપ સાથે સામનો કરવા માટે શકય છે. આ હ ાઈસ્પીડ સંગ્રહ ઉપકરણ ખૂબ જ ખર્ચાળ છે અને તેથી સંગ્રહ બીટ દીઠ ખર્ચ પણ ખૂબ જ વધારે હોય છે. મુખ્ય મેમરીનો સંગ્રહ ક્ષમતા પણ ખૂબ જ મર્યાદિત છે. ઘણી વખતે તે પ્રક્રિયા કરવા માટે CPU ની સેંકડો માહિતી લાખો બાઈટ્સમાં સંગ્રહ માટે જરૂરી છે તેથી વધારાની મેમરી તમામ કોમ્પ્યુટર માટે જરૂરી છે. આ મેમરી સહાયક મેમરી અથવા ગૌણ સંગ્રહ (Secondary) તરીકે ઓળખાય છે.

આ પ્રકારની મેમરીમાં સંગ્રહ બીટ દીઠ ખર્ચ ઓછો થાય છે. જો કે ઓપરેટીંગની ઝડપ પ્રાથમિક મેમરી કરતાં ધીમી છે. માહિતી વિશાળ જથ્થો (વોલ્યુમ) કાયમી ધોરણે સંગ્રહિત થાય છે અને જયારે જરૂર પડે ત્યારે પ્રાથમિક સંગ્રહમાં પરિવહન થાય છે. સૌથી વધુ ઉપયોગમાં લેવાતી ગૌણ સંગ્રહ ઉપકરણો ચુંબકીય ટેપ અને ચુંબકીય ડિસ્ક (હાર્ડ ડિસ્ક) છે.

1.6.1 ચુંબકીય ટેપ

ચુંબકીય ટેપમાં માહિતી મોટી વોલ્યુમ અને લાંબા સમય માટે સ્ટોર થયેલ છે, જેનો મેઈન ફ્રેમ કોમ્પ્યુટર જેવા મોટા કોમ્પ્યુટર માટે ઉપયોગ થાય છે. તમે કેસેટ સ્વરૂપમાં ટેપ વાપરી શકો છો. ટેપ ડેટા સ્ટોરની કિંમત સસ્તી છે. ટેપ કાયમ માહિતી સંગ્રહ કરે છે. તે ચુંબકીય પદાર્થ ધરાવે છે. તે 12.5 મીમીથી 25 મી.મી. પહોળી પ્લાસ્ટીક ટેપ (ફિલ્મ ટેપ જેવી) અને 500 થી 1200 મીટર લાંબી જે ચુંબકીય પદાર્થથી આવરિત (કોટેડ) હોય છે. આ કેન્દ્રીય પ્રોસેસર સાથે જોડાયેલ છે અને માહિતી આપવામાં અથવા ટેપના પ્રોસેસર દ્વારા વાંચવામાં આવે છે. તે કેસેટ ટેપ રેકોર્ડર (ઓડીયો કેસેટસ) જેવી જ છે.

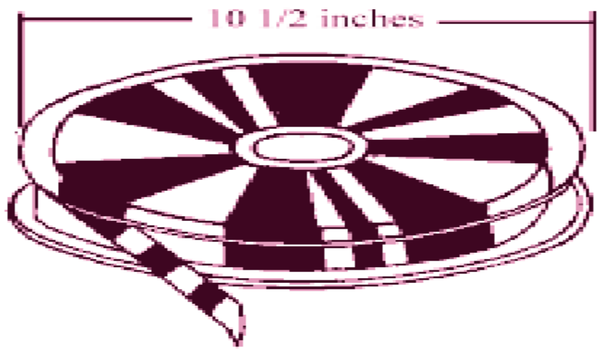
ચુબંકીય ટેપના લાભો

- **સઘન** (Compact) - ટેપ 10 ઈંચ મહત્તમ વ્યાસ, 2400 ફૂટ લાંબી છે અને તેની લંબાઈ દરેક ઈંચમાં 800, 1600 અથવા 6250 અક્ષરો એક ઈંચમાં પકડી શકે છે. ટેપની મહત્તમ ક્ષમતા 180 મિલિયન અક્ષર છે. આમ માહિતી ટેપ પર વધુ સઘન સંગ્રહાય છે.
- **આર્થિક :** બીજા સંગ્રહ ઉપકરણોની સરખામણીમાં ટેપમાં અક્ષરોને સ્ટોર કરવાની કિંમત ખૂબ જ ઓછી છે.
- **ઝડપી :** માહિતીની નકલ સરળ અને ઝડપી છે.
- **લાંબાગાળાના સ્ટોરેજની અને ફરી ઉપયોગિતા :** ચુંબકીય ટેપ લાંબા ગાળાના સંગ્રહ માટે વપરાય છે. અને એક કોઈપણ પ્રકાર માહિતીને નુકશાન વગર વારંવાર વાપરી શકાય છે.

16.2 ચુંબકીય ડિસ્ક

તમે ગ્રામો ફોનના રેકોર્ડ જોયા હશે કે જે ડિસ્ક જેવા ગોળ અને ચુંબકીય સામગ્રી સાથે આવરણયુક્ત છે. કોમ્પ્યુટરમાં ચુંબકીય ડિસ્કો એ જ સિધ્ધાંત પર બનાવવામાં આવે છે. તે કોમ્પ્યુટર ડ્રાઈવની અંદર ખૂબ જ ઊંચી ઝડપ સાથે ફરે છે. ડેટા ડિસ્કની સપાટી બંને પર

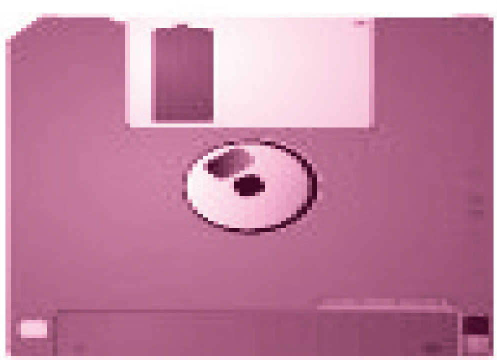
સંગ્રહ થાય. ચુંબકીય ડિસ્ક સીધી જ વાપરવા માટે સૌથી વધુ લોકપ્રિય છે. દરેક ડિસ્કને ટ્રેક હોય છે. જેમાં અદૃશ્ય કેન્દ્રિત સર્કલોનો સમાવેશ થાય છે. માહિતી નાના ચુંબકીય સ્થળોના સ્વરૂપમાં ડિસ્કની સપાટી પરના ટ્રેક પરના રેકોર્ડ થાય છે, ચુંબકીય સ્થળોની હાજરી એક બીટ (1) રજૂ કરે છે, અને તેની ગેરહાજરી (0) શૂન્ય બીટ પ્રતિનિધિત્વ કરે છે. ડિસ્ક પર સંગ્રહિત માહિતી બીજાને અસર કર્યા વિના ઘણીવાર વાંચી શકાય છે. તેથી વાંચનની કામગીરી બિન વિનાશક છે. જો તમે નવી માહિતીને લખવા માંગો છો, તો પછી વર્તમાન માહિતી ડિસ્કમાંથી ભૂંસીને અને નવી માહિતી રેકોર્ડ કરે છે.



આકૃતિ - 1.3 ચુંબકીય ટેપ

1.6.3 ફ્લોપી ડિસ્ક

તે ઉપર ચર્ચા કર્યા મુજબ ચુંબકીય ડિસ્ક જેવી જ છે તેનો વ્યાસ 3.5 ઈંચ છે.એક 3.5 ફ્લોપીની ક્ષમતા 1.44 મેગાબાઈટ્સ છે. તે કોઈ પણ બીજા સંગ્રહ ઉપકરણો કરતાં સસ્તી છે અને પોર્ટેબલ (હળવી) છે. ફ્લોપી વ્યક્તિગત કોમ્પ્યુટર પધ્ધતિ માટે ખાસ કરીને ઓછામાં ઓછા ખર્ચમાં ઉપલબ્ધ સાધન છે.



આકૃતિ - 1.4 ફ્લોપી ડિસ્ક

દરેક એપ્લીકેશન અને સોફટવેર સાથે મેમરી ક્ષમતા માટે વધુ માંગ રહે છે.તેથી ઓપ્ટિકલ સંબંધી ડિસ્ક સંગ્રહ મીડિયાનો વિકાસ થયો છે જે વિશાળ માહિતી સંગ્રહ કરવાની જરૂરિયાત છે. ઓપ્ટિકલ સંબંધી ડિસ્કને નીચેના વર્ગોમાં વિભાજિત કરી શકાય છે.

1. **સઘન ડિસ્ક/મેમરી ને ફક્ત વાંચો (CD - ROM)** - સીડી રોમની ડિસ્ક પ્રતિબિંબિત ધાતુઓથી બનાવવામાં આવે છે. સીડીરોમની હાઈ પાવર લેસર દ્વારા ઉત્પાદન પ્રક્રિયા દરમિયાન લખાયેલ છે. અહીં સંગ્રહ ઘનતા ખૂબ જ ઊંચી છે. સંગ્રહ ખર્ચ ખૂબ જ ઓછો છે અને વપરાશ સમયના પ્રમાણમાં ઝડપી છે. દરેક ડિસ્ક આશરે સાડાચાર ઈંચ વ્યાસની હોય છે અને 600 થી વધુ એમબીની માહિતી સમાવી શકાય છે. સીડીરોમને માત્ર વાંચી શકાય છે, તેમાં લખી અથવા સમાયેલા માહિતીમાં ફેરફાર કરી શકાતો નથી.

2. **એક વાર લખો,ઘણું વાંચો (Write Once Read Many - WORM) :** CD ROM પર કંઈ લખી શકાતું નથી, આ ગેરલાભને WORM વાપરવાથી દૂર કરી શકાય છે. એક WORM વપરાશકર્તા ડિસ્ક પર કાયમી ધોરણે ડેટા લખવા માટે પરવાનગી આપે છે. એક વાર ડેટા લખાઈ જાય ત્યાર પછી ભૌતિક ડિસ્ક નુકશાન વિના ડેટાને દૂર કરી શકાતો નથી. અહીં માહિતી કી- બોર્ડ, વિડિયો, સ્કેનર, OCR સાધનો અને અન્ય ઉપકરણોથી રેકોર્ડ કરી શકાય છે. WORM ફાયદો છે કે તે ગીગા બાઈટ્સમાં માહિતી સ્ટોર કરી શકે છે. WORM માંથી કોઈ પણ દસ્તાવેજનો ખૂબ જ ઝડપી ઉપયોગી કરી શકાય છે.

3. **ઈરેસેબલ ઓપ્ટિકલ ડિસ્ક :** આ પ્રકારની ડિસ્કમાં માહિતી, ભૂંસી અને ફરીથી લખી શકાય છે. આ પણ ડેટા લખવા અને ફરીથી લખવા માટે લેસર બીજાનો ઉપયોગ કરે છે.આ ડિસ્ક પરંપરાગત ડિસ્ક માટે વિકલ્પ તરીકે ઉપયોગી છે. ઈરેસેબલ ઓપ્ટિકલ ડિસ્ક (એમાં) ચુંબકીય ઓપ્ટિકલ (MO) તરીકે ઓળખાતી ટેકનોલોજી પર આધારિત છે. ઈરેસેબલ ઓપ્ટિકલ ડિસ્ક પર ડેટા બીટ લખવા માટે (MO) દ્વારા લેસર બીમને ડિસ્કની સપાટી પરના કોઈ ચોક્કસ બિંદુ પર લઈ જઈ, તેને મેગ્નેટાઈઝ કરાય છે.

1.7 અંક ગણિત તર્ક એકમ (Arithmetic Logic Unit - ALU)

આ એકમ બધા જ ગાણિતિક અને તાર્કિક કાર્યો કરે છે, જેવા કે સરવાળો, ગુણાકાર સરખામણી વગેરે. દાખલા તરીકે A અને B નો સરવાળો વિચારો. કંન્ટ્રોલ યુનિટ સંખ્યા A ને મેમરીમાંથી પસંદ કરશે અને ALU માં લાવશે. જવાબને ફરીથી મેમરીમાં સ્ટોર કરાશે અથવા AU ની અંદર વધારાની ગણતરી માટે રાખવામાં આવશે.

1.8 નિયંત્રણ એકમ (Control Unit - CU)

આ એકમ સિસ્ટમમાં બીજા બધા એકમોનું સંકલન કરે છે તેના મુખ્ય કાર્યો છે :

(i) વિવિધ એકમો વચ્ચે માહિતી અને જાણકારીનું સ્થળાંતર કરી નિયંત્રણ કરે છે.

(ii) અંકગણિત તર્ક એકમ (ALU) દ્વારા યોગ્ય કાર્ય શરૂ કરે છે.

વપરાશકારનો પ્રોગ્રામ મૂળભૂત નિયંત્રણ અને સૂચનાઓ પૂરી પાડે છે. કલ્પનાત્મક રીતે, નિયંત્રણ એકમ (CU) મેમરીને ડિકોડ કરી સૂચનાઓ મેળવે છે અને સ્પષ્ટ કાર્યો કરવા માટે વિવિધ એકમોને માર્ગદર્શન આપે છે.

1.9 આઉટપુટ ડિવાઈસ

આઉટપુટ ડિવાઈસ, માહિતીને CPU માંથી વપરાશકર્તાને જરૂરી પ્રકારે બતાવે છે. આઉટપુટ ડિવાઈસમાં ડિસપ્લે સ્ક્રીન, સ્પીકર, પ્રિન્ટર, પ્લોટર વગેરે આવે છે.

જયારે પ્રોગ્રામને લખવામાં આવે છે ત્યારે ડિસપ્લે સ્ક્રીન અક્ષરો પ્રદર્શિત કરે છે. વપરાશકર્તા વાકયો વાંચી અને સંગ્રહ અથવા પ્રિન્ટ થાય તે પહેલાં સુધારી શકે છે. સ્ક્રીન પર બાહ્ય સંગ્રહમાંથી પ્રોગ્રામને લાવીને સુધારી શકાય છે.

સ્ક્રીનનું કદ અલગ અલગ સિસ્ટમમાંથી અલગ હોય છે. તેનું પ્રમાણભૂત માપ 80 અક્ષરો અને 24 રેખાઓ છે. મોટા ભાગની સિસ્ટમમાં સ્ક્રોલીંગ (સરકાવાની) ની સગવડ હોય છે. આ સુવિધાઓ વપરાતાં સ્ક્રીન પર છુપાયેલું લખાણ લાવવામાં સ્ક્રીન પર ઊભું અથવા આડું લખાણ ખસેડે છે. આમ વપરાતી સમગ્ર ફાઈલની સમીક્ષા કરવા માટે અથવા ચોક્કસ ભાગ પસંદ કરવા માટે સ્કેન કરી શકાય છે. સ્ક્રીન પર કર્સર, કી બોર્ડ પરની કી દ્વારા નિંયત્રિત થાય છે.

19.1. ડિસ્પલે સ્ક્રીન

એક કાર્યક્મ કિડ કરવામાં આવ્યો છે. (એક ટીવી સ્ક્રીન પર સમાન છે.) દર્શાવેલ અક્ષરો પ્રદર્શિત કરે છે. વપરાશ કરતાં વાકય દ્વારા કાર્યક્મમાં વાંચી અને પ્રિન્ટર પર સંગ્રહ અથવા છપાયેલી હોય છે. પહેલા કરેકશન કરી શકાય છે. તે સ્ક્રીન પર બાહ્ય સંગ્રહ સંપાદન સંગ્રહિત કાર્યક્મ ભાગ લાવી શકાય છે.

સ્ક્રીન (દર્શાવેલ) કદ સ્થાનિક સિસ્ટમમાંથી અલગ પડે છે. તે પ્રમાણભૂત માપ 80 અક્ષરો દ્વારા 24 રેખાઓ છે. તમારી પધ્ધતિને સરકાવનાર માટે જોગવાઈ જોઈએ અને આ સુવિધાઓ વપરાતાં સ્ક્રીન છુપાયેલા લખાણ માટે લાવવામાં સ્ક્રીન પર ઊભું અથવા આડું લખાણ ખસેડે છે. આમ વપરાતી સમગ્ર ફાઈલની સમીક્ષા કરવા માટે અથવા ચોક્કસ ભાગ પસંદ કરવા માટે સ્કેન કરી શકાય છે. સ્ક્રીન પર કર્સર કી બોર્ડ પરની કી દ્વારા નિયંત્રિત થાય છે.

1.9.2 પ્રિન્ટર

પ્રિન્ટર એક ડીવાઈસ છે જે ટ્રાન્સપરન્સી અને મીડિયા પર લખાણ અથવા ચિત્રો છાપે છે. પ્રિન્ટરો ઘણા વિવિધ પ્રકારના હોય છે. ટેકનોલોજીની દષ્ટિએ પ્રિન્ટરના નીચે પ્રમાણે ભાગ પાડયાં છે.

(i) ઈન્ક જેટ પ્રિન્ટર

ઈન્કજેટ પ્રિન્ટર કાગળની શીટ પર શાહીનો છંટકાવ દ્વારા કામ છે. ઈન્કના રસ્તા પર ચુંબકીય પ્લેટ જરૂરી આકારમાં કાગળ પર શાહી પાડે છે. ઈન્કજેટ પ્રિન્ટર લેસર પ્રિન્ટરો દ્વારા ઉત્પાદિત કે અસમાન ઉચ્ચ ગુણવત્તાવાળી પ્રિન્ટ ઉત્પાદનને કરવા માટે સક્ષમ છે. એક લાક્ષણિક ઈન્ક જેટ પ્રિન્ટર ઈંચ દીઠ 300 બિંદુઓ પ્રિન્ટ કરે છે.

સામાન્ય રીતે ઈન્કજેટ પ્રિન્ટરોની કિંમત લેસર પ્રિન્ટરોની કિંમત કરતાં ઓછી હોય છે. તે પણ નોંધપાત્ર ઓછી છે. આ પ્રિન્ટરનો બીજો ગેરફાયદો તે સસ્તી નકલ બનાવનાર કાગળ

નોંધ

પર સ્મજ ચાલક છે.એક ખાસ પ્રકારની શાહી જરૂરી છે.

ઈન્કજેટ પ્રિન્ટરોમાં લેસર પ્રિન્ટરો કરતાં નાના યાંત્રિક ભાગોની જરૂર પડે છે. તેઓ પોર્ટેબલ (હળવું) પ્રિન્ટર તરીકે લોકપ્રિય છે. વધુમાં કલર ઈન્કજેટ પ્રિન્ટરોમાં રંગીન પ્રિન્ટ સસ્તી કિંમતમાં થાય છે.

આકૃતિ - 1.5 ઈન્કજેટ પ્રિન્ટર

(ii) લેસર પ્રિન્ટર

લેસર પ્રિન્ટર ડ્રમ પર ફોટો પેદા કરવા માટે લેસર બીમનો ઉપયોગ કરે છે. લેસર બીમ ડ્રમને જયાં પણ અડકે ત્યાંનો ઈલેકટ્રિકલ ચાર્જ બદલાય છે. ડ્રમને પછી ટોનરને ગરમી અને દબાણ સાથે સંયોજન દ્વારા કાગળ પરિવહન કરાય છે. આ રીતે કોપી મશીનો કામ કરે છે.

ટોનર લાગુ પડે તે પહેલાં સમગ્ર પાનું ડ્રમ પર ફેલાય છે. આ કારણે લેસર પ્રિન્ટરને કયારેક પેજ પ્રિન્ટર (Page Printer) કહેવામાં આવે છે. તેઓ બધા લેસરનો ઉપયોગ કરતાં નથી છતાં પણ લેસર પ્રિન્ટરો કહેવાય છે જેના બે પ્રકાર છે. એક ડ્રમ એલઈડી ઉપયોગ કરે છે અને અન્ય એલસીડી ઉપયોગ કરે છે. એક વાર ડ્રમ ચાર્જ થઈ જાય છે પછી બંને વાસ્તવિક લેસર પ્રિન્ટરના જેમ કાર્ય કરે છે.

લેસર પ્રિન્ટરોના મુખ્ય લક્ષણોમાંનું એક તેઓ ઠરાવ કેટલા બિંદુઓ પ્રતિ ઈંચ મૂકે તે છે, જેને રીઝોલ્યુશન કહે છે. આ રીઝોલ્યુશન 300 ડીપીઆઈ થી 1200 ડીપીઆઈમાં હોય છે. કેટલાક લેસર પ્રિન્ટરો એ રીઝોલ્યુશન તરીકે સામાન્ય રીતે ઓળખવામાં આવતી ખાસ ટેકનિક સાથે આવર્તન હાંસલ કર્યું છે.

લખાણ કરતાં લેસર પ્રિન્ટરો ગ્રાફિક્સ પ્રિન્ટીંગના નિષ્ણાંત હોય છે. તમે ઉચ્ચ રિઝોલ્યુશન ગ્રાફિકસ છાપવા વધુ મેમરી જોઈએ છે. 300 ડિપીઆઈ સાથે સંપૂર્ણ પાનાનું ગ્રાફિક છાપવા માટે ઉદાહરણ તરીકે પ્રિન્ટરમાં ઓછામાં 1 MB ની જરૂર છે. 600ડીપીઆઈમાં ગ્રાફિક માટે ઓછામાં ઓછી 4 MB RAM જરૂરી છે.

લેસર પ્રિન્ટરની ઝડપથી 4 થી 20 પાના પ્રતિ મિનિટ હોય છે. 6ppm એ એક 40 અક્ષરો પ્રતિ સેકન્ડ (CPS) બરાબર છે.

(iii) લાઈન પ્રિન્ટર

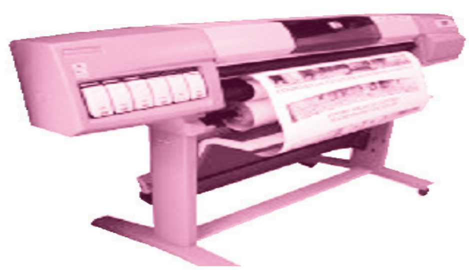
લાઈન પ્રિન્ટર એક સમયે એક સંપૂર્ણ વાકય પ્રિન્ટ કરવાની ક્ષમતા ધરાવતી ઉચ્ચ ઝડપનું પ્રિન્ટર હતું. એક ઝડપી લાઈન પ્રિન્ટર મિનિટ દીઠ 3000 લાઈન છાપી શકે છે. લાઈન પ્રિન્ટરો નો ગેરફાયદા એ છે કે તેઓ માત્ર એક જ ફોન્ટ છાપતાં હોય છે. તેઓ ગ્રાફિકસ છાપી શકતા નથી. તેમાં પ્રિન્ટની કવોલીટી (કક્ષા) સારી હોતી નથી અને તેઓ બહું અવાજ કરે છે.

(iv) થર્મલ પ્રિન્ટર

થર્મલ પ્રિન્ટરો ખાસ કરીને ગરમી પ્રત્યે સંવેદનશીલ કાગળ પર વીજળીની ગરમી દ્વારા પ્રિન્ટ બનાવતા પ્રિન્ટરો છે. થર્મલ પ્રિન્ટરો સસ્તા હોય છે અને મોટા ભાગના કેલ્કયુલેટર અને ઘણા ફેકસ મશીનોમાં વપરાય છે. તેઓ ઓછી ગુણવત્તાની પ્રિન્ટ ઉત્પન્ન કરે છે અને કાગળ થોડા અઠવાડિયા કે મહિનાઓ પછી વળી જાય અને જરી શકે છે.

1.9.3 પ્લોટર

પ્લોટર કોમ્પ્યુટરથી આદેશો પર આધારિત કાગળ પર ચિત્રો દોરે છે.પ્લોટર પેનની મદદથી લીટીઓ દોરવા બાબતે પ્રિન્ટર્સ કરતાં અલગ પડે છે. પ્રિન્ટરો માત્ર એક બિંદુઓથી નજીક અંતરે શ્રેણી છાપવા માટે લાઈન ઊભી કરી શકાય છે, જયારે પ્લોટર સતત રેખાઓ ઉત્પન્ન કરી શકે છે. વિવિધ રંગો દોરવા વિવિધ રંગની પેન વાપરે છે.



આકૃતિ - 1.6 પ્લોટર

સામાન્ય રીતે પ્લોટર પ્રિન્ટરો કરતાં નોંધપાત્ર રીતે વધુ ખર્ચાળ છે. તેઓ ઈજનેરી એપ્લીકેશનમાં ઉપયોગી થાય છે, જયાં ચોકકસાઈ ફરજિયાત છે.

1.9.4 સાઉન્ડ કાર્ડ અને સ્પીકર

આનાથી કોમ્પ્યુટર અવાજોનું સંપાદન કરી શકે છે. સાઉન્ડ કાર્ડ લગભગ તમામ સીડી રોમ માટે જરૂરી છે અને મોર્ડન વ્યકિતગત કોમ્પ્યુટર પર સામાન્ય બની ગયા છે. સાઉન્ડ કાર્ડ કોમ્પ્યુટર સાથે જોડાયેલા સ્પીકર પર અવાજ મોકલે છે, માઈક્રો ફોન વાપરીને અવાજને રેકોર્ડ કરે છે, અનજે ડિસ્ક પર સ્ટોર કરેલા અવાજોનું સંપાદન કરે છે.

લગભગ બધા સાઉન્ડ કાર્ડ મીડી ઈલેકટ્રોનિક સંગીત રજૂ કરવા માટે પ્રમાણભૂત આધાર આપે છે. વધુમાં મોટા ભાગના સાઉન્ડ કાર્ડ સાઉન્ડ બ્લાસ્ટર સાથે કામ કરી શકે તેવા હોય છે, એટલે કે તેઓ સાઉન્ડ બ્લાસ્ટર માટે લખાયેલ કમાન્ડ વાપરી શકે છે.

આકૃતિ - 1.7 સાઉન્ડ કાર્ડ અને સ્પીકર

1.9.5 3D ઓડીયો

3D ઓડીયો પરંપરાગત સ્ટીરિયો અવાજમાં વધુ ઉંડાઈઓ (સ્પષ્ટતા) ઉમેરવા માટેની એક ટેકનિક છે. ખાસ કરીને 3D સાઉન્ડ, 3D ઓડિયો માટે, સ્ટીરિયો સ્પીકર સાથે ઓરડામાં ઉપકરણ મૂકીને બનાવવામાં આવે છે. આ ઉપકરણ આવતા અવાજ વિશ્લેષણ કરે છે, તેને સુધારે છે અને એવી છાપ ઉભી કરે છે કે સ્પીકર દૂર છે.

3D ઓડિયો ખાસ કરીને એવી જગ્યાએ પ્રચલિત છે જયાં કોમ્પ્યુટરમાં સ્પીકર નાના અને નજીક ગોઠવાયેલ હોય. એક કોમ્પ્યુટર સાઉન્ડ કાર્ડ સાથે ઘણી 3D ઓડિયો ડીવાઈસ જોડી શકાય છે.

પાઠગત પ્રશ્નો 1.3

1. નીચેનાઓને વ્યાખ્યાયિત કરો.
   (અ) અંકગણિત તર્ક (ALU)
   (બ) ઈન્કજેટ પ્રિન્ટર
   (ક) પ્લોટર (Plotter)
2. ખાલી જગ્યા પૂરો.
   (અ) ....એકમ બીજી સિસ્ટમમાં બધા એકમનું સંકલન કરે છે.
   (બ) દર્શાવેલ સ્ક્રીનની પ્રમાણભૂત માપ ......રેખાઓ ......અક્ષરો છે.
   (ક) ....પ્રિન્ટરમાં ડ્રમ પર છબી પેદા કરવા માટે લેસર બીમનો ઉપયોગ થાય છે.
   (ડ) ....ઓડિયો પરંપરાગત સ્ટીરિયો અવાજ માટે વધુ ઊંડાઈ આપવા માટેની એક ટેકનિક છે.

1.10 તમે જે શીખ્યા

આ પાઠમાં તમે કોમ્પ્યુટરના વિવિધ ઘટકો વિશે વિગતવાર શીખ્યા છો. તમે એ પણ શીખ્યાં કે કેવી રીતે કોમ્પ્યુટરના વિવિધ એકમો દ્વારા આપેલ કામ પરિપૂર્ણ કરે છે.

1.11. પાઠચાંત પ્રશ્નો

1. *કોમ્પ્યુટરના મહત્વના ઘટકો કયા કયા છે ?*
2. *સીપીયુ કેવી રીતે કામ કરે છે તે ટૂંકમાં સમજાવો.*
3. *પ્લોટર એટલે શું ? તે કેવી રીતે કામ કરે છે.*
4. *પ્રિન્ટરના વિવિધ ભાગોનું વર્ણન કરો.*
5. *તફાવત લખો.*

   (અ) રેમ અને રોમ (RAM and ROM)

   (બ) અસ્થિર (Volatile) *અને બિનઅસ્થિર* (Non - Volatile) *મેમરી*

1.12. પાઠગત પ્રશ્નોના જવાબો.

પાઠગત પ્રશ્નો 1.1

1. *ચાર મૂળભૂત કાર્યો છે :*

   (અ) ઈનપુટ પ્રાપ્ત કરવું.

   (બ) માહિતીની પ્રક્રિયા કરવી

   (ક) આઉટપુટ પેદા કરવું.

   (ડ) ભવિષ્યમાં ઉપયોગ માટે માહિતીને સ્ટોર કરવી.
2. *સ્કેનર એ ઈનપુટ ડિવાઈસ છે જે અક્ષરો અથવા કાગળ પર મુદ્રિત ઉદાહરણ વાંચી શકે છે અને કોમ્પ્યુટર ઉપયોગ કરી શકે તેવા ફોર્મમાં માહિતીનું અનુવાદ કરી શકે છે.*
3. *(અ) અંકગણિત તર્ક એકમ* (ALU) *(બ) કોમ્પ્યુટરનું મુખ્ય એકમ* (CPU)

   (ક) પ્રક્રિયા (Processing)

પાઠગત પ્રશ્નો 1.2

1. *બીટની માહિતી એક બાઈનરી ભાગમાં રહે છે કે જે બાઈનરી અંક માટે ટુંકુ નામ છે. આ* 0 *અથવા* 1 *હોય છે બાઈટમાં* 8 *બાઈટ્સનો સમાવેશ થાય છે. રેમનો દરેક સંગ્રહ સ્થાન એક બાઈટની માહિતી સ્ટોર કરે છે.*
2. *વીજપુરવઠો જતો રહેતાં જે મેમરીમાંથી માહિતી ભૂંસાઈ જાય છે તેને વોલેટાઈલ મેમરી કહે છે.* RAM *વોલેટાઈલ મેમરી છે. બીજી તરફ રોમ એ નોન વોલેટાઈલ મેમરી છે, કારણ કે, વીજ પુરવઠો જતાં તેમાં રહેલી માહિતી ભૂસાંતી નથી.*
3. *(અ) સાચું (બ) ખોટું (ક) ખોટું (ડ) ખોટુ (ઈ) સાચું*

પાઠગત પ્રશ્નો 1.3

1. *(અ) અંક ગણિત તર્ક એકમ* (ALU) *સરવાળો, બાદબાકી, ગુણાકાર સરખામણી વગેરે કામગીરી કરવા માટે ઊપયોગી થાય છે.*

   (બ) ઈન્કજેટ પ્રિન્ટર એક કાગળ શીટ પર શાહીના છંટકાવ દ્વારા કામ કરે છે. આ પ્રિન્ટર લેસર પ્રિન્ટર્સ જેવી ઉચ્ચ ગુણવત્તાવાળી પ્રિન્ટ પેદા કરવાની ક્ષમતા ધરાવે છે.

   (ક) પ્લોટર કોમ્પ્યુટર આદેશોના આધારિત કાગળ પર ચિત્ર બનાવે છે.

2. *(અ) નિયંત્રણ (બ) ચોવીસ, એંશી* (24,80) *(ક) લેસર (ડ)* 3 *ડી*

2

માહિતી પ્રક્રિયાના ખ્યાલ

2.1 પ્રસ્તાવના

દરેક સંસ્થા કે બિઝનેસ, તેના વ્યવહારોના કદ અથવા હેતુને અનુલક્ષીને માહિતી પેદા કરે છે જે ઘટનાઓનો રેકોર્ડ રાખે છે. ડેટા ને ઉત્પન્ન તથા આયોજન કરવાની પ્રક્રિયા ને માહિતી આયોજન કહેવાય છે. કોમ્પ્યુટર એક મહત્વપૂર્ણ માહિતી પ્રક્રિયાના સાધન તરીકે ઉપયોગી થાય છે. આ પાઠમાં, આપણે વિવિધ મુદ્દાઓ વિશે ચર્ચા કરીશું જેમ કે માહિતી (ડાટા), માહિતી (ઈન્ફોર્મેશન), માહિતી પ્રક્રિયા અને માહિતી પ્રક્રિયા સિસ્ટમ.

2.2 હેતુઓ

આ પાઠ પસાર થયા પછી, તમે આ સ્થિતિમાં હશો

- *વ્યાખ્યાયિત કરી શકશો : માહિતી (ડેટા), માહિતી (ઈન્ફોર્મેશન), માહિતી પ્રક્રિયા*
- *માહિતી પ્રક્રિયાની વિવિધ પ્રવૃત્તિઓ સમજવી*
- *માહિતી પ્રક્રિયા ની સાયકલ (જીવનકાળ) નો ઉપયોગ.*
- *માહિતી તત્વો, રેકોર્ડ, ફાઈલો અને ડેટાબેઝ વચ્ચે નો તફાવત*

2.3 ડેટા

"માહિતી" શબ્દ ડેટમ(datum) નું બહુવચન છે જેનો મતલબ છે હકીકત, ધારણા, નિરીક્ષણ અથવા ઘટના. માહિતી ની વધુ સ્પષ્ટ રીતે લોકો, વસ્તુઓ, વિચારો અને ઘટનાઓ લગતી

હકીકતો થી રજૂઆત કરી શકીએ છીએ. માહિતી આંકડાકીય મૂળાક્ષર અથવા અન્ય ખાસ સંજ્ઞાઓ અને પ્રતીકો દ્વારા રજૂ થાય છે.

2.4 ડેટા પ્રોસેસીંગ (Data processing)

માહિતી પ્રક્રિયાનું કાર્ય માહિતી સંભાળવા અથવા અમુક રીતે માહિતી હેરફેર કરવાનું છે. માહિતી પ્રક્રિયા તેમાં સામેલ પ્રવૃતિઓને અનુલક્ષીને માહિતી નો અર્થ સોંપવા માટે પ્રયાસ કરે છે. આમ, પ્રક્રિયા નો અંતિમ ધ્યેય ડેટા ને માહિતી (ઈન્ફોર્મેશન) માં પરિવર્તન કરવાનો છે.

માહિતી પ્રક્રિયા એક પ્રક્રિયા છે જેમાં હકીકત અને આંકડાઓ એકત્રિત કરવા , અર્થ સોંપવો, અન્ય સાથે સંપર્ક કરવો અને ભવિષ્ય ના ઉપયોગ માટે જાળવી રાખવા થાય છે.

આમ ડેટા પ્રોસેસીંગ એટલે ચોક્કસ કાર્યોનો સમુહ અથવા પ્રક્રિયા જે ડેટાને ઉપયોગી માહિતીમાં ફેરવે છે. "ડેટા પ્રોસેસીંગ સિસ્ટમ" એટલે બધા જ સ્ત્રોતોનો સમૂહ જે ડેટા પ્રોસેસીંગ માટે વપરાય છે.

2.5 માહિતી (Information)

માહિતીને આ પ્રમાણે વ્યાખ્યાયિત કરી શકાય : " એવો ડેટા (માહિતી) જેને કોઈ ચોક્કસ હેતુ માટે અર્થપૂર્ણ અને ઉપયોગી સ્વરૂપમાં ફેરવેલ હોય". કોઈક વખત ડેટાને પ્રોસેસ કરવાની જરૂર પડતી નથી. જો કે, સામાન્ય રીતે જયાં સુધી ડેટાને વ્યવસ્થિત રીતે ગોઠવી, વિશ્લેષણ અને મૂલ્યાંકન ના કરવામાં આવે ત્યાં સુધી તે બહુ ઉપયોગી નથી. આ પછી જ ડેટા માહિતી બને છે.

જ્યારે માહિતી (ડૅટા)માથી માહિતી(ઈન્ફોર્મેશન) રુપંતરિત થાય તે નક્કી કરવા માટે કોઈ સખત નિયમ નથી. અક્ષરો અને સંખ્યાઓ સમૂહ એક વ્યક્તિ માટે અર્થ કરી શકે છે, પરંતુ અન્ય વ્યક્તિ માટે અર્થ ન પણ હોય શકે. માહિતીની ઓળખાણ તેના વપરાશકર્તાઓ દ્વારા થાય છે.

ઉદાહરણ તરીકે, જ્યારે તમે કોઈ કરિયાણા સ્ટોર માથી ખરીદી કરો , માહિતી વસ્તુઓની સંખ્યાબંધ માહિતી જેમકે તમારું નામ, સરનામું, તમે ખરીદી કરેલ વસ્તુઓ, વસ્તુઓ ના નંબરો, ભાવ, કર અને તમે ચૂકવણી જથ્થો વિગેરેની માહિતીમાં મૂકવામાં આવે છે. આ બધી વસ્તુઓની માહિતી અલગ હોય છે, પરંતુ તમે એક સાથે આ આઈટમ મૂકી, બિઝનેસ વ્યવહાર વિશે જાણકારી રજૂ કરી શકો છો.

2.6 માહિતી પ્રક્રિયા પ્રવૃત્તિ (Data Processing Activities)

ઉપર ચર્ચા કર્યા મુજબ, માહિતી પ્રક્રિયા માં એવી પ્રવૃત્તિઓનો સમાવેશ થાય છે. જે ડેટાને માહિતીમાં બદલે. માણસ ને માહિતી પ્રક્રિયામાં તેમને મદદ કરવા માટે અમુક સાધનો બનાવ્યા છે. આમ જેમ કે, માનવીય સાધનો જેવાકે પેંસિલ અને કાગળ, યાંત્રિક સાધનો જેવા કે ફાઈલોની ગોઠવણી, ઈલેક્ટ્રોમિકેનિકલ સાધનો જેવા કે મશીનો અને ટાઈપરાઈટર, ઈલેક્ટ્રોનિક સાધનો જેવાકે કેલ્ક્યુલેટર અને કોમ્પ્યુટર્સ વગેરે સાધનો નો સમાવેશ થાય છે. ઘણા લોકો માહિતી પ્રક્રિયા ને તરત જ કોમ્પ્યુટર્સ સાથે સાંકળે છે. ઉપર જણાવયા મુજબ માત્ર કોમ્પ્યુટર જ એવું સાધન નથી કે જે માહિતી પ્રક્રિયામાંનો ભાગ હોય, કોમ્પ્યુટર વગર પણ તે થઈ શકે છે. જો કે લોકો એ ચોક્કસ કાર્યો માટે કમ્પ્યુટર્સને પણ પાછળ રાખી દીધા છે. અમુક અન્ય ક્રિયાઓ જેમ કે માનવ કૌશલ્ય

નોંધ

અને બુદ્ધિ માટે કમ્પ્યુટરએ અવેજી છે,આકૃતિ માં બતાવ્યા પ્રમાણે ડેટા પ્રોસેસીંગ માટે જે વિવિધ કાર્યો અને પ્રવૃત્તિઓ જરૂરી છે તેઓને 5 મૂળભૂત વર્ગોમાં જૂથ કરી શકાય.

સંગ્રહ/સંચય **સ્ત્રોત** **માપવું** **રેકોર્ડિંગ** **સરખામણી**	**રૂપાંતરણ** **કોડિંગ** **વર્ગીકરણ** **ચકાસણી** **રૂપાંતરિત**	**મેનીપ્યુલેશન** **ગોઠવણી** **ગણતરી** **સારાંશ** **સરખામણી**	**સંગ્રહ** **સ્ટોરિંગ** **પુનઃ પ્રાપ્તિ**	**સંદેશા વ્યવહાર** **અને** **પુન : ઉત્પાદન**

આકૃતિ 2.1 ડેટા પ્રોસેસીંગ માટેની પ્રવૃત્તિઓ

હવે આપણે આ દરેક પ્રવૃત્તિનુ વર્ણન કરીશું

Deptt. 5 Employee Time Sheet for Week ending 21/07/2001							
			Hours Worked				
Emp No.	**Name**	**Mon**	**Tues**	**Wed**	**Thurs**	**Fri**	**Total Hours**
110	Hariom	8	8	8	8	8	40
150	Dinesh	8	8	8	8	6	38
160	Rajesh	8	4	8	8	8	36
170	Pankaj	8	6	8	5	8	35

આકૃતિ. 2.2 પેરોલ એપ્લીકેશન માટે સ્ત્રોત સાપ્તાહિક ટાઈમ શીટ છે.

(અ) સંગ્રહ

માહિતી ઘટનાઓ, વ્યવહાર અથવા અમુક અવલોકનો સ્વરૂપમાં ઉદ્દભવે છે. આ માહિતીનો પછી કેટલાક ઉપયોગી સ્વરૂપમાં રેકોર્ડ થાય છે. માહિતીને શરૂઆતમાં કાગળ સ્ત્રોત, દસ્તાવેજો પર રેકોર્ડ અને પછી પ્રક્રિયા મશીન ઉપયોગી ફોર્મ રૂપાંતરિત કરી શકાય છે. વૈકલ્પિક રીતે તેઓ એક કાગળ વિહિન, મશીન વાંચી શકે તેવા સ્વરૂપમાં પ્રત્યક્ષ ઈનપુટ ઉપકરણ દ્વારા રેકોર્ડ કરી શકાય છે. માહિતીના સંગ્રહને પણ માહિતીના સંચય તરીકે ઓળખવામાં આવે છે.

(બ) રૂપાંતરણ

એક વાર માહિતી(ડેટા) નો સંગ્રહ થયા પછી, તે માહિતી પ્રક્રિયા માટે વધુ યોગ્ય હોય તેમ મૂળભૂત દસ્તાવેજમાંથી રૂપાન્તિત કરવામા આવે છે. ડેટા પ્રથમ ઓળખ કોડ સોંપણી દ્વારા ઘડાયો છે. તેમાં અનેક કોડ નંબરો, અક્ષરો, વિશિષ્ટ અક્ષરો અથવા આમાંથી કોઈ એક સંયોજનોનો સમાવેશ થાય છે. ઉદાહરણ તરીકે, એક કર્મચારીને પોતાની શ્રેણી તરીકે 52 - 53 - 162 એમ એક કોડ ફાળવવામાં આવે છે. તે માહિતી વર્ગીકરણ અને માહિતીના કાયદાને વ્યવસ્થિત રૂપ આપવા માટે ઉપયોગી છે. વર્ગીકૃત એટલે કે, માહિતી કે જે સમાન લાક્ષણિકતાઓ અને સમાન વર્ગો વાળાં એક જૂથોમાં મૂકવામાં આવે છે. ઉદાહરણ તરીકે, એક એકાઉન્ટ ની માહિતીઓ ને તારીખ અનુસાર કરવી, સરવૈયું સરળતાથી તૈયાર કરી શકાય છે. ડેટાનું વર્ગીકરણ થયા પછી અને પ્રક્રિયા શરૂ થતા પહેલા તેને ચોક્કસ સ્વરૂપે ચકાસવામા આવે છે. હવે ચકાસણી થયા પછી

નોંધ

માહિતી(ડેટા) એક માધ્યમથી બીજા માધ્યમમાં રૂપાંતર પામે . ઉદાહરણ તરીકે માહિતી પ્રક્રિયા જો કોમ્પ્યુટરની મદદથી પૂર્ણ થઈ હોય તો માહિતી તેના મૂળભૂત સ્વરૂપથી સાધન સ્વરૂપ જેમકે ચુમ્બકિય ટેપ કે ચુમ્બકિય ડિસ્ક માં રુપાંતર થાય છે.

(ક) ઉપયોગ/કુશળ વ્યવહાર

જયારે ડેટાનો સંગ્રહ તથા રૂપાંતરણ થાય, ત્યાર બાદ રૂપાંતરણના એકમ(ફંકશન) કે જે ડેટાને માહિતી(ઈંફોર્મેશન)માં રૂપાંતરિત કરે છે. રુપાંતરણની અન્ય પ્રક્રિયઓ નીચે મુજબની છે :

ગોઠવણી

તે માહિતી વસ્તુઓની વ્યવસ્થા જરૂરી ક્રમ પ્રમાણે કરે છે. સામાન્ય રીતે તે ડેટા સાથે કામ કરવા માટે સરળ છે જો તે ડેટા લોજિકલ ક્રમમાં ગોઠવાયેલ હોય. મોટા ભાગે, ડેટા મૂળાક્ષર ક્રમમાં ગોઠવાયેલ હોય છે. ક્યારેક ગોઠવણી જાતે જ ડેટા ને માહિતી (ઈંફોર્મેશન)માં રૂપાંતરિત કરે છે. ઉદાહરણ તરીકે, ટેલિફોન ડિરેક્ટરી માં અંગ્રેજી બારાખડી પ્રમાણે નામોની સરળ યાદીની ગોઠવણ હોય છે. ડિરેક્ટરી ગોઠવણ વગર કામ લાગશે નહીં.

બિઝનેશ માહિતી પ્રક્રિયા વ્યાપક રીતે ગોઠવણીનો ઉપયોગ કરે છે. વાસ્તવમાં કેટલીક બિઝનેસ ફાઈલોમાં તમામ રેકોર્ડ તાર્કિક ક્રમ પ્રમાણે જરૂરી રખાયુ છે. સામાન્ય રીતે મૂળાક્ષર ગોઠવણી કરતાં આંકડાકીય ગોઠવણી ઝડપી છે કારણ કે તે કોમ્પ્યુટર આધારિત પ્રક્રિયા છે.

ગણતરી

ગાણિતિક સંખ્યાની ચકાસણી એટલે કે ગણતરી.કોઈપણ વસ્તુઓની માહિતી(ડેટા) એક બીજા સાથે ઉમેરી, બાદબાકી, ભાગાકાર કે ગુણાકાર કરી નવી માહિતી ((ડેટા/ ઈંફોર્મેશન)બનાવી શકાય છે. ગણતરી એ માહિતી પ્રક્રિયા નો મુળભૂત એક્મ છે. ઉદાહરણ તરીકે એક કર્મચારીનું વેતન તેણે કરેલ કામના કલાક અને કલાકના મહેનતાણાના ગુણકારથી મેળવી શકાય. તે પરથી તેનો કુલ પગાર પણ તે ગણતરીની મદદથી જાણી શકાય છે. મળવેલ કુલ રકમ, ચૂકવવા પાત્ર રકમ , કર (આવક વેરા) કે જે પગારમાંથી બાદબાકી કરી કુલ ચોખ્ખો પગાર જાણી શકાય છે.

સારાંશ

સારાંશ એટલે કે રહેલ માહિતી(ડેટા) ના જથ્થાને સંક્ષિપ્ત કરી વધુ સારી રીતે ઉપયોગમાં લઈ શકાય તેવો બનાવવો. ઉદાહરણ તરીકે તમે હાજર રહેલ તાશ (લેક્ચર) નો સારાંશ તેના મુખ્ય મુદા કે નોધ જે એક થી બે પાના (પીઈજ) સ્વરૂપે બનાવી શકો છો. જ્યારે માહિતી(ડેટા) ગાણિતીક આંકડા સ્વરૂપમાં હોય ત્યારે તેનો સારાંશ ગાણિતીક પધ્ધતિથી તે વર્ગીકરણ કરીને કરી શકાય. અન્ય ઉદાહરણ જોઈએ તો એક મુખ્ય મેનેજર જે કુલ વેચાણ થયેલ વસ્તુની માહિતી(ડેટા) તેની નીચે રહેલા વેચાણ અધિકારી પાસેથી એકત્ર કરી વેચાયેલ વસ્તુ તથા ખરીદનારની માહીતીનો એ સારાંશ છે.

સરખામણી

સરખામણી એટલે માહિતી ડેટા નું વિશ્લેષણ કે સંબંધિત માપન કરવું. ઉદાહરણ તરીકે, ધંધાના મનેજર પોતનો ધંધો કેવી પ્રગતિ કરી રહ્યો છે , તે જાણવા ચાલુ મહિનાના વેચાણને પાછલા વર્ષના વેચાણની માહિતી (ડેટા)ની સરખામણી કરે છે.

(ડ) આઉટપુટ રીઝલ્ટની જાળવણી (Managing the output Results)

એક વખત ડેટા લેવાઈ જાય પછી નીચેની પ્રક્રિયાઓ કરી શકાય છે. :

સ્ટોરીંગ (storing)

સ્ટોર એટલે ડેટાને પાછળથી વપરાશ માટે જાળવી રાખવો. સ્ટોરેજ એ ડેટાના પુનઃ ઉપયોગ અને પ્રોસેસીંગ માટેની પધ્ધતિમાં જરૂરી ભાગ છે. ડેટા સ્ટોર કરવા માટેની માનવીય પધ્ધતિમાં ફાઈલ કેબીનેટનો, અને કોમ્પ્યુટર આધારિત પધ્ધતિમાં મેગ્નેટીક ડિસ્ક અને ટેપ જેવા ઈલેક્ટ્રોનીક સાધનો વપરાય છે. સ્ટોર કરવાની પ્રક્રિયાઓમાં ડેટાને સ્ટોર કરવો, માહિતીને યોગ્ય ક્રમમાં ગોઠવવો જેથી તેને સરળતાથી પાછી મળેવી શકાય વગેરેનો સમાવેશ થાય છે. આપણે ત્યારે જ ડેટા ને સ્ટોર કરવો જોઈએ જયારે ડેટાને સ્ટોર કરવાની કિંમત તેના ભવિષ્યમાં થનાર કિંમત કરતાં ઓછી હોય.

પુનઃપ્રાપ્તિ (Retrieving)

પુનઃપ્રાપ્તિ એટ્લે જે માહિતી(ડેટા) કે માહિતી(ઈંફોર્મેશન) સંગ્રહિત છે. તેને ફરીથી શોધવી કે પાછી મેળવવી. પુનઃપ્રાપ્તિ પધ્ધતિ માહિતી(ડેટા) સંગ્રહિત એકમો (ડિવાઈસ) નો ઉપયોગ કરે છે.આવી જ રીતે માહિતી(ડેટા) કે જે ફાઈલ અને ક્બાટની અંદર હોય કે કોમ્પ્યુટરમાં હોય તેને ફરીથી શોધવામાં આવે છે. આગળની પ્રકીયા માટે પુનઃપ્રાપ્તિ અને સરખામણી જ્યારે સંગ્રહિત માહિતી(ડેટા) પર કરવામા આવે છે, ત્યારે સમજી શકાય તેવી માહિતી(ઈંફોર્મેશન) મેળવી શકાય છે.

Weekly Payroll Summary Report 07/07/2005					
Dept.	Employee Number	Name	Hours Worked	Pay Rate	Gross Wages
2	170	Pankaj	34	10.00	340.00
	175	Rekha	32	9.00	288.00
	158	Param	20	5.00	100.00
	160	Rajesh	36	8.00	288.00
	165	Sanju	45	9.00	405.00
	159	Savini	25	3.00	75.00
Department 2 Total			192		1496.00
4	110	Hari Om	33	7.75	255.75
	115	Vimla	40	9.25	370.00
Department 4 Total			73		625.75
8	150	Dinesh	38	7.00	266.00
	155	Upma	40	9.45	390.00
	156	Arunima	32	8.50	272.00
	157	Ritwik	40	8.00	320.00
Department 8 Total			150		1,248.00
Company Total			415		3,369.75

આકૃતિ. 2.3.(a) વીકલી પે રોલ માહિતી

Department Payroll Summary for Week of 07-07-2005		
Department	**Hours**	**Gross Wages**
2	192	1,496.00
4	73	625.75
8	150	1,248.00
Company Total	415	3,369.75

આકૃતિ. 2.3.(b) ડિપોઝીટ પે રોલ માહિતી

ઈ) વાર્તાલાપ/સંદેશા વ્યવહાર

માહિતી(ડેટા/ ઈંફોર્મેશન) ને વહેંચવાની પ્રક્રિયાને સંદેશાવ્યવ્હાર કે વાર્તાલાપ કહે છે. જ્યાં સુધી માહિતી(ડેટા/ ઈંફોર્મેશન) તેના જરૂરીયાત વાળા ઉપયોગકર્તા (યુઝર) ને ન મળે ત્યાં સુધી તે અર્થહીન ગણાય છે. આમ,સંદેશા વ્યવહાર એટલે માહિતી (ડેટા/ઈંફોર્મેશન) નું સ્થળાંતર,જે માહિતી પ્રક્રિયા વ્યવસ્થા, જે ઉપયોગકર્તા દ્વારા ઉપયોગ કરી શકાય.આમ પરિણામ સ્વરૂપ માહિતીનો રિપોર્ટ કે તેના કાગળ બનવવામાં આવે છે.અને ઉપયોગકર્તા સુધી પહોચાડવામાં આવે છે. વિજાણુ/વિજળી (ઈલેક્ટ્રોનિક) માહિતી પ્રક્રિયામાં નિર્ણય કે આઉટપુટ ની આપ-લે ડિસ્પ્લેય એકમ દ્વારા થાય છે.

ઉ) પુનઃઉત્પાદન(રીપ્રોડ્ક્શન)

માહિતી(ડેટા/ ઈંફોર્મેશન) ની તેના જેવીજ બીજી નકલ તૈયાર કરવી.આવી નકલ કરવાની પ્રક્રિયા માનવીય રીતે કે સાધન દ્વારા માહિતી તૈયાર કરી શકાય છે.

પાઠગત પ્રશ્નો 2.1

1. *નીચેનાનો તફાવત આપો.*
   અ. ડેટા અને માહિતી (ઈંફોર્મેશન).
   બ. માહિતી પ્રક્રિયા અને માહિતી પ્રક્રિયા વ્યવસ્થા.
2. *નીચેના પદો વ્યાખ્યાઈત કરો.*
   (અ) અનુક્રમ (sorting)
   (બ) સંશિપ્તકરણ(summarizing)

2.7 માહિતી પ્રક્રિયા ચક્ર (data processing cycle)

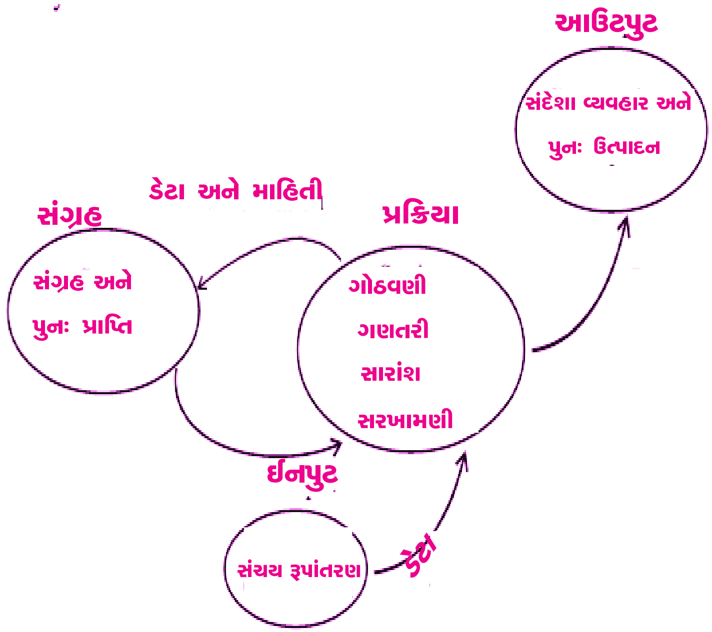
માહિતી પ્રક્રિયાની ઉપર દર્શાવેલ પ્રકીયાઓ મેન્યુઅલ અને ઈલેક્ટ્રોનિક સિસ્ટમમાં રૂપાંતરીત સામાન્ય છે.

આ પ્રક્રિયાઓ મુખ્ય ચાર જુથમાં રજૂ કરી શકાય છે. જેમ કે માહિતી(ડેટા),ઈનપુટ, માહિતી પ્રક્રિયા, ડેટા આઉટ્પૂટ અને માહિતી નો સંગ્રહ, જે એકબીજા સાથે અરસપરસ સંક્ડાયેલા હોય છે કે જેને માહિતી પ્રક્રિયા ચક્ર કહેવાય છે.

1. **ઈનપુટ**

ડેટા (માહિતી) નો ઈનપુટ એ એક પ્રક્રિયા છે જેમા ડેટા(માહિતી) નો સંગ્રહ કરે ત્યાર બાદ તેને પ્રક્રિયા કરવા માટે મોક્લે છે. ઈનપુટ દાખલ કરેલ માહિતીની તપાસ કરે છે, ચકાસે છે અને ત્યાર બાદ તેને માન્ય રાખે છે.

નોંધ



આકૃતિ 2.4 માહિતી પ્રક્રિયા ચક્ર

2. **પ્રક્રિયા**

પ્રક્રિયા નો અર્થ સૂચવે છે કે માહિતીનું વર્ગિકરણ, ગોઠવણી,ગણતરી, સારાંશ ,સરખામણી વગેરે પ્રવૃતિઓનો સમાવેશ થાય છે. જે માહિતીને અર્થપૂર્ણ માહિતીમાં રૂપાંતરીત કરે છે.

3. **આઉટપુટ**

આઉટપુટ એટલે (તેવા વ્યક્તિઓ માટે કે જેને તેની જરૂર હોય) માહિતીનું સંચાર અને વહન કરવું. ક્યારેક આઉટપુટમાં ડીકોડીંગ(લીપી ઉકેલવા) પ્રવુતિનો સમાવેશ થાય છે. જેના દ્વારા ઈલેક્ટ્રોનિક માહિતી માણસ વાંચી શકે તેવાં સ્વરૂપમાં રૂપાંતરીત કરે છે.

4. **સ્ટોરેજ (સંગ્રહ)**

આના ડેટા કે માહિતી ને ભવિષ્યમાં કામ લાગે તે માટે સ્ટોર કરવામાં આવે છે.ઉપર દર્શાવેલ ચાર મૂળભૂત કાર્ય હંમેશા તાર્કિક ક્રમ રીતે એક બીજા સાથે જોડાયેલ હોય છે.

2.8 કોમ્પ્યુટર પ્રક્રિયાની કામગીરી

કોમ્પ્યુટર નીચે દર્શાવેલ મુખ્ય ચાર કામગીરી કરે છે કે જે ઉપર દર્શાવેલ/જણાવેલ બધીજ પ્રક્રિયાઓને સમાવી લે છે.

(અ) ઇનપુટ/આઉટપુટ કામગીરી

કોમ્પ્યુટર કાચી માહિતી(ડેટા) ઈનપુટ એકમ પાસેથી સ્વીકારે છે અને અર્થપુર્ણ માહિતી(ઈંફોર્મેશન) આઉટપુટ ને આપે છે. આ કાર્ય અલગ- અલગ ધણા બધા ઈનપુટ આઉટપુટ એકમ દ્વારા થતા હોય છે.જેવાકે કિબોર્ડ,ડિસ્પ્લે સ્ક્રીન(મૉનિટર),પ્રિન્ટર. આ બધા એક્મો દ્વારા જ માણસ અને કોમ્પુટર વચ્ચે આપ લે થઈ શકે છે

(બ) કેલ્ક્યુલેશન અને લખાણ કામગીરી

કોમ્પ્યુટર અંદર રહેલી સર્કીટ નંબરો ઉપર ગણતરી કરે છે. તે લખાણમાં રહેલ નંબર તથા તથા સિમ્બોલ (નિશાની) ને લગતી કામગીરી એટલી જ કુશળતાથી કરે છે.

(ક) તર્ક/સરખામણી કામગીરી

કોમ્પ્યુટર લોજિકલ કામગીરી કરવાની ક્ષમતા પણ ધરાવે છે.ઉદાહરણ તરીકે જો આપણે બે આઈટમ્સ (વસ્તુઓ) A અને B ને સરખાવીએ તો તેની ત્રણ લોજિકલ શક્યતાઓ હોઈ શકે.

A એ B થી નાનો હોય (A<B), A અને B સમાન હોય (A = B) અથવા A એ B થી મોટો હોય (A > B). કોમ્પ્યુટર આવી સરખામણી કરી, પુર્વ નિર્ધારિત નિર્ણય પર પહોચી શકે છે.આવી સરખામણી કરવાની કળા એ કોમ્પ્યુટરની મુખ્ય કામગીરી ગણાય છે.

(ડ) સ્ટોરેજ અને પુનઃપ્રાપ્તિઃ

માહીતી તથા કાર્યક્રમ સુચનો બંને કોમ્પ્યુટરની અંદર સ્ટોરેજ (સંગ્રહ) પામે છે. એક સમયે તેઓ ઈંટરનલ મેમરીમાં સ્ટોર થયા પછી ઝડપથી વાપરી કરી શકય છે, અને તેનો વધુ ઉપયોગ કરી શકય છે.

2.9 માહિતી પ્રક્રિયાની વ્યવસ્થા(સિસ્ટમ)

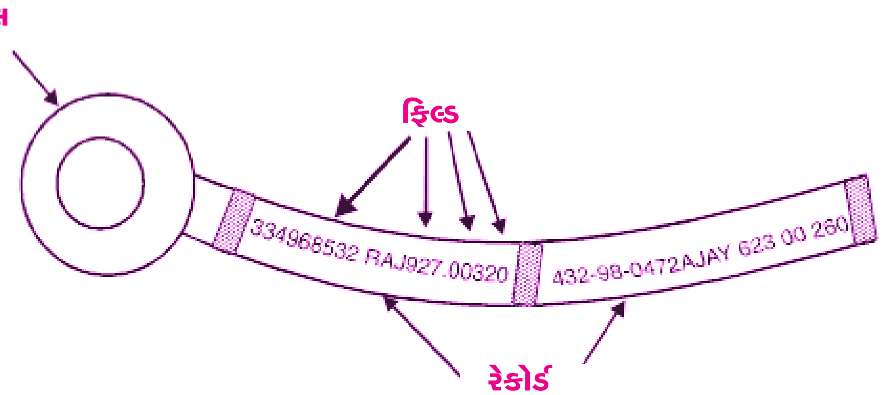
માહિતીની પ્રક્રિયા અને પ્રવુતિઓ "સિસ્ટ્મ" તરીકે ઓળખાય છે. જેમ્સઓ બ્રેનિના કહેવા મુજબ સિસ્ટ્મ ની વ્યાખ્યા નીચે મુજબ આપી શકાય."એક ગ્રપમાં એકબીજા સાથે જોડાયેલ ઘટકો કે જે એક સંગથિત પ્રક્રિયા નાં ઈનપુટ આઉપુટ સ્વીકારી એક સર્વ સામાન્ય ધ્યેય પ્રાપ્તી કરે છે."ઉદાહરણ તરીકે એક ઉત્પાદન એક્મ –જૂથ(સિસ્ટમ) રો મટીરીયલને ઈનપુટ તરીકે લઈ ઉત્પાદન તરીકે નિર્માણ થયેલ આઈટમને આઉટપુટ તરીકે આપે છે.

આવી જ રીતે ડેટા પ્રક્રિયા ગ્રુપ જે માહિતી(ડેટા) ને સ્વીકારી તેના પર યોગ્ય પ્રક્રિયા કરી સંપુર્ણ માહિતી(ઇંનફોર્મેશન) બનાવે છે.

માહિતી પ્રક્રિયા ગ્રુપના પ્રકાર ઘણા બધા છે. મેનુઅલ ડેટા પ્રક્રિયા કે જે પેન કબાટના ખાના નો ઉપયોગ કરે છે, મશીન આધારિત ડેટા પ્રોસેસ કે જે ટાઈપ રાઈટર,કેલ્ક્યુલેટર,બુક રાખવાના મશીનનો ઉપયોગ તથા છેલ્લે ઈલેક્ટ્રોનિક આધારિત ડેટા પ્રક્રિયા કે જે કોમ્પ્યુટરનો ઉપયોગ કરે છે, જે માહિતી પર જાતે જ પ્રક્રિયા કરી શકે છે.

2.10 ડેટા વ્યવસ્થા (Data Organisation)

ડેટા પ્રોસેસ ના ચક્ર ની ચર્ચા કર્યા બાદ અને તેના ઘટકોની જાણકારી લીધા બાદ હવે ,આપણે જોઈએ કે રો ડેટા (કાચી માહીતી) પર પ્રકીયા કરતા પહેલા તેનુ આયોજન કેવી રીતે કરી શકાય. ડેટાને વિવિધ રીતે ગોઠવી શકાય છે.પરંતુ સામાન્ય રીતે આયોજન એક હાઈરારર્ચિકલ(અધિકમિક ભાગ) રીતે કરવું હિતાવહ છે.



આકૃતિ. 2.5 ડેટા રોકોર્ડ

- **ડેટા/આઇટ્મ (માહિતી વસ્તુ)**

  કોમ્પ્યુટર ની ફાઈલમાં ડેટા (માહિતી) સ્ટોર કરતો નાનામાં નાનો એકમ ડેટા આઈટ્મ છે.આ એક એવું તત્વ છે જે ડેટા ના મુખ્ય ઘટકો રજુ કરે છે. જેવા કે કર્મચારીનું નામ વસ્તુ ની કિંમત વિગેરે.એક સેલેરી (પગાર) એપ્લીકેશનના સોફટવેરમાં કર્મચારીનું નામ ''પંકજ'' એ પણ ડેટા આઈટમ (માહિતી વસ્તુ) છે.

- **ફિલ્ડ(ક્ષેત્ર)**

  કોમ્પ્યુટર ફાઈલમાં ડેટા આઈટ્મ(માહીતી વસ્તુ)ભૌતિક ક્ષેત્રો માં ગોઠવાયેલ હોય છે.તેઓની લેંથ(લંબાઈ)કદાચ નિશ્ચીત અથવા અનિશ્ચીત હોઈ શકે છે.જેમકે કર્મચારીઓ નો નંબર 3 આંકડાથી વધુ ન હોવો જોઈએ. આ ત્રણ આંકડા ની કિંમત અલગ અલગ ડેટા(માહિતી) વસ્તુ ને સંગ્રહ કરે છે. આમ, તે નિશ્ચીત ક્ષેત્ર થયું. આનથી વીપરીત અલગ અને વીપરીત લંબાઈનું હોઈ શકે તેથી તેનો સંગ્રહ(સ્ટોરેજ) કરવા માટે મહત્તમ લંબાઈ આપી શકાય છે આમ, આ ક્ષેત્ર અનિશ્ચીત થયું કહેવાય.

રેકોર્ડ(નોંધ)

રેકૉર્ડ(નોંધ) એટ્લે એક્બીજા સાથે જોડાયેલ માહિતી વસ્તુ(ડેટા આઈટ્મ) દરેક રેકૉર્ડ(નોંધ) કોઈ ચોક્કસ માહીતીની જાણકારી આપતી હોય છે. ઉ.દા. અલગ અલગ ક્ષેત્રો એ એક નોંધ તૈયાર થાય છે. જે આકૃતિ 2.4 મુજબ કર્મચારીઓનાં નંબર, કર્મચારીનું નામ, એનો મુળ પગાર ભથ્થુ, અને ઘર ભાડા ભથ્થા. આ માહિતી(ડેટા આઈટ્મ) થી પગાર પત્રક બનાવાય છે. પ્રથમ રેકોર્ડ પંકજ નામના કર્મચારીની માહિતી આપે છે. ત્યાર બાદ બીજી નોધ રેખા નામની કર્મચારીની છે. દરેક અનુક્રમે આપેલ નોંધ કર્મચારી માટેની દરેક માહિતી દર્શાવે છે. આ દર્શાવે છે કે દરેક વસ્તુની માહિતી કેવી રીતે એક બીજા સાથે જોડાયી એક્જૂથ બનાવી નોંધ ને રજુ કરે છે.

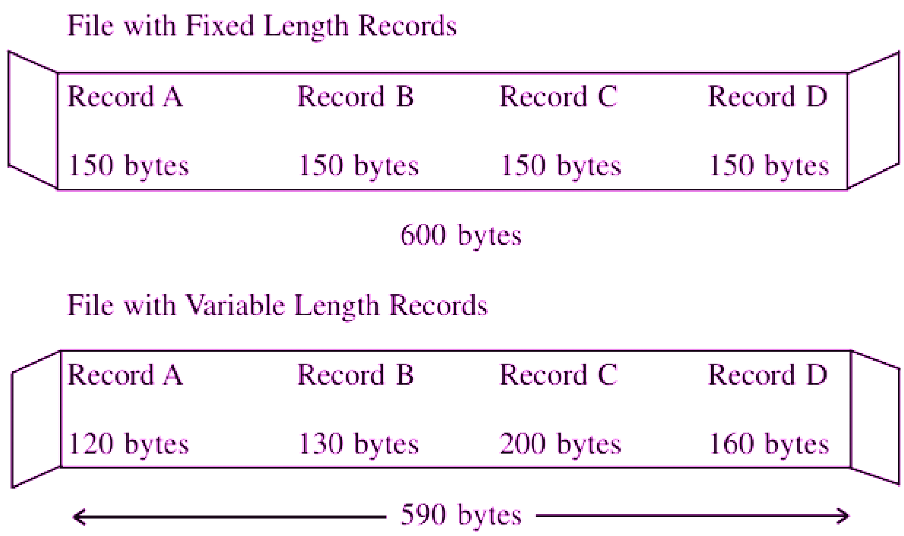
ફાઇલ

નોંધ(રેકોર્ડ) નો સમૂહ એટ્લે ફાઈલ. ફાઈલ એપ્લિકેશન માટે તમામ સંબધીત નોંધ ધરાવે છે. આવીજ રીતે એક પગાર પત્રક માં આકૃતિ 2.5 માં દર્શાવ્યા મુજબ બધી જ સંબધીત

નોંધ એકબીજા સાથે જોડાઈ પગાર પત્રકની રચના કરે છે. સામાન્ય રીતે ફાઈલનો સંગ્રહ કોઈને કોઈક માધ્યમ ઉપર થયો હોય છે. જેમકે ચુમ્બકીય ટેપ , ચુમ્બકીય ડિસ્ક કે ફલોપી ડિસ્ક.

ડેટાબેઝ

એકબીજા સાથે સંબંધિત ફાઈલનો સમુહ એટ્લે ડેટાબેઝ, ડેટાબેઝ એ ચોક્કસ એપ્લિકેશન માટે તમામ સંબંધીત ફાઈલોનો સમાવેશ કરે છે.



આકૃતિ. 2.6.

2.11 અલગ - અલગ અને ચોક્કસ લંબાઈનો રેકોર્ડ (Variable and Fixed Length Records)

રેકૉર્ડ ચોક્કસ લંબાઈ અથવા તો ચલીત લંબાઈ ધરાવતા હોઈ શકે જેમ કે આકૃતિ 2.5 માં દર્શાવેલ છે.

ચોક્કસ લંબાઈના રેકોર્ડ

અહીં આપેલી ફાઈલના રેકોર્ડ એક જ સરખા બાઈટ્સની લંબાઈ ધરાવે છે. આવી ફાઈલને ફ્લેટ ફાઈલ કહે છે. જો બધા જ રેકોર્ડ એક સરખા ડેટાનો જથ્થો આવવાની સંભાવના હોય તો ચોક્કસ લંબાઈનો રેકોર્ડ વપરાય છે.

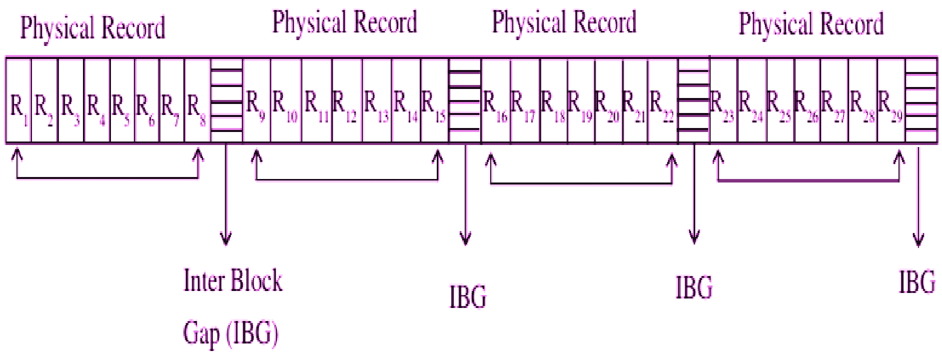
બદલાતી લંબાઈના રેકોર્ડ

આવા કિસ્સામાં રેકોર્ડની લંબાઈ અલગ - અલગ હોય છે. બદલાતની લંબાઈના રેકોર્ડનો ઉપયોગ જયારે ફાઈલના રેકોર્ડ અલગ - અલગ જથ્થા લંબાઈ વાળા હોય ત્યારે સ્ટોરેજ જગ્યા બચાવે છે.

2.12 તાર્કિક (લોજીકલ) ની સરખામણીએ ફીઝીકલ રેકોર્ડ

લોજીકલ રેકૉર્ડમાં એક એન્ટિટી સંબંધિત માહિતી હોય છે જે પગાર પત્રક (પે રોલ) માં રહેલ કર્મચારીની વિગત આપવા અથવા વિદ્યાર્થી દ્વારા પ્રેકટિકલ પરીક્ષા માં મેળવેલ માર્કસની યાદી. એક ફિઝિકલ રેકૉર્ડ જેમની અલગ-અલગ માહિતી એક બીજા માટે સ્ટોર(સંગ્રહ)કરી તેનો ઉલ્લેખ કરે છે. તે એક ઈનપુટ આઉટપુટ ડિવાઈસ(ઉપકરણ) દ્વારા એક એકમ તરીકે ગણવામાં

આવે છે. એક જ લોજીકલ રેકોર્ડની માહિતી અલગ - અલગ ફિઝીકલ રેકોર્ડમાં અથવા અનેક લોજીકલ રેકૉર્ડ એક જ ફિઝિકલ રેકૉર્ડમાં સ્ટોર હોઈ શકે ઉદાહરણ તરીકે મેગ્નેટીક ટેપમાં ઘણા લોજીકલ રેકોર્ડ એક બ્લોકમાં ઝડપ વધારવા લખાય છે. જેને ફિઝિકલ રેકોર્ડ કહેવાય છે. જે આકૃતિ. 2.6 માં બતાવ્યું છે.



આકૃતિ. : 2.7 ફીઝીકલ અને લોજીકલ રેકોર્ડ

પાઠગત પ્રશ્નો 2.2

1. *ડેટા પ્રોસેસિંગની વિવિધ પ્રક્રિયાઓ જણાવો ?*
2. *ડેટા પ્રોસેસિંગ સાયકલના વિવિધ પગલાંની વ્યાખ્યા જણાવો.*

2.13 તમે જે શીખ્યા ?

આ પ્રકરણમાં આપણે જાણ્યું કે કાચી માહિતી કેવી રીતે અર્થપૂર્ણ માહિતીમાં રૂપાંતરિત થાય છે. વિવિધ ડેટા પ્રોસેસિંગને લગતા કાર્યોમાં કોમ્પ્યુટરના ઉપયોગની ચર્ચા પણ કરી. ડેટાને લગતાં હાઈરાર્ચી (અધિશ્રેણી) ની ચર્ચા પણ પાઠમાં કરી.

2.14 પાઠચાંત પ્રશ્નો

1. *વ્યાખ્યા આપો : ડેટા, ડેટા પ્રોસેસિંગ અને ઈન્ફર્મેશન*
2. *મેનીપ્યુલેશન (ઉપયોગ) માં વપરાતી વિવિધ પ્રક્રિયાઓ જણાવો.*
3. *ટૂંકમાં લખો. : કોમ્પ્યુટરમાં થતી વિવિધ પ્રક્રિયાઓ વિશે ટૂંકમાં લખો.*
4. *પ્રક્રિયા થતાં પહેલાં ડેટા (માહિતી) કેવી રીતે ગોઠવાય છે ? ટૂંકમાં ચર્ચા કરો.*

2.15 પાઠગત પ્રશ્નોના જવાબો

પાઠગત પ્રશ્ન - 2.1

(અ) ડેટા બહુવચન શબ્દ જે ડેટામાંથી બન્યો છે. જેનો મતલબ અનુમાન કે ધારણા એવો થાય છે. અર્થપૂર્ણ માહિતીની વ્યાખ્યા એવી કહી શકાય કે - કાચી માહિતી જે ચોક્કસ ઉદેશ્ય માટે પ્રકીયા થઈ મૂળભૂત માહિતી (રો - ડેટા) માંથી રૂપાંતરિત થઈ છે.

(બ) માહિતી પ્રક્રિયા એવી પ્રક્રિયા છે જેમાં ચોકસાઈ વાળા આંકડા નું સંગ્રહણ થયેલું હોય. જે મતલબ સમજાવે, બીજા સાથે સંદેશા વ્યવહાર કરે અને જરૂર પડે ભવિષ્યમાં તેનો ઉપયોગ કરી શકે. તે ઘણી બધી પ્રતિક્રિયા અને ઓપરેશનનો સમૂહ છે. જે કાચી માહિતીને

અર્થપૂર્ણ માહિતીમાં રૂપાંતરિત કરે છે. ડેટા (માહિતી) પ્રોસેસિંગ સીસ્ટમમાં બીજા એકમ (ડીવાઈસ) પણ સાથે જોડવામાં આવે છે.

2.

(અ) તે ડેટા આઈટમ ને જરૂરિયાત મુજબ યોગ્ય ક્રમમાં ગોઠવે છે.

(બ) આનો અર્થ એવો પણ થાય કે જેમ જરૂર પડે તેમ ડેટાનો જથ્થો વધારી કે ઘટાડી શકાય છે.

પાઠગત પ્રશ્નો 2.2

1. ડેટા માહિતી પ્રોસેસિંગની પ્રક્રિયાઓ નીચે દર્શાવેલ પાંચ જૂથમાં વહેંચી શકાય.
   1. એકત્રીકરણ (કલેકશન)
   2. રૂપાંતરણ (કન્વરઝન)
   3. કુશળ વ્યવહાર (મેનીપ્યુલેશન)
   4. સંગ્રહ અને પુનઃપ્રાપ્તિ
   5. વાર્તાલાપ
2. નીચે દર્શાવેલ મુખ્ય પદ જે ડેટા (માહિતી) પ્રોસેસિંગ સાયકલના ભાગ છે.
   1. માહિતી નિવેશ (ડેટા ઈનપુટ)
   2. માહિતી પ્રક્રિયા (ડેટા પ્રોસેસ)
   3. માહિતી નિર્ગમ (ડેટા આઉટપુટ)
   4. માહિતી સંગ્રહ (ડેટા સ્ટોરેજ)

3

કોમ્પ્યુટર સોફ્ટવેર

3.1 પ્રસ્તાવના

જો તમે કોઈ માણસ પાસેથી કામ કરાવવા માગતા હોય, તો તમને તેને સમજાતી ભાષામાં શું કરવું છે તે કહેશો. તેવી જ રીતે જો તમારે કોમ્પ્યુટર પાસેથી કંઈ કામ કરાવવું હોય,ત્યારે કોમ્પ્યુટર સમજે તેવી ભાષામાં કહેવું પડે એટલે કે મશીન ભાષા. મશીન ભાષા બાયનરી અંકોથી (દ્વિઅંકી) બનેલી છે, એટલે કે 0 અને 1. મનુષ્ય માટે દ્વિઅંકી સંખ્યામાં વિચારવું એ ખૂબ જ જટલિ અને મુશ્કેલ મનાય છે. કોમ્પ્યુટર સાથે વાતચીત માટે, એવું વિચારવામાં આવ્યું હતું કે કોઈ ત્રીજી પ્રોગ્રામિંગ ભાષા વિકસાવાય જે મનુષ્ય અને કોમ્પ્યુટર બંન્ને સમજી શકે. આમ પ્રોગ્રામીંગ ભાષાએ નિયમોનો સમૂહ છે જે કોમ્પ્યુટરને ચોક્કસ કામગીરી કરવા માટે સૂચનાઓ આપે છે.

પ્રોગ્રામિંગ ભાષાઓ નીચી અથવા ઊંચી કહી શકાય, તેનો આધાર કોમ્પ્યુટર વાપરે છે તે ભાષાની કેટલી નજીક (નીચી, એટલે કે 0 અને 1) અથવા મનુષ્ય વાપરે છે તે ભાષાની (ઉચ્ચ, એટલે કે અંગ્રેજી પ્રકારની) કેટલી નજીક છે તેના પર રહેલો છે.

3.2 ઉદ્દેશો

આ પાઠ ભણ્યા પછી તમે નીચેનાઓ માટે સક્ષમ હશો :-

- *મશીન ભાષા વિવિધ સ્તરે (અથવા પેટી) ઓળખવી જેમ કે મશીન ભાષા, એસેમ્બલી ભાષા, ઉચ્ચ સ્તરીય ભાષા*
- *સોફ્ટવેરનો ખ્યાલ સમજાવવા*

- સિસ્ટમ સોફ્ટવેર અને એપ્લિકેશન સોફ્ટવેર વચ્ચેનો તફાવત સમજવા
- કમ્પાઈલર અને ઈન્ટરપ્રિટર વચ્ચેનો તફાવત
- ઓપરેટિંગ સિસ્ટમ અને તેના વિવિધ કાર્યો વ્યાખ્યાયિત કરી શકશો.

3.3 કોમ્પ્યુટરની ભાષા (Computer Language)

જે ભાષામાં પ્રોગ્રામ લખવામાં આવે છે તેને પ્રોગ્રામીંગ ભાષાઓ કહે છે. આ ભાષાઓ નીચેના વર્ગોમાં વર્ગીકૃત કરી શકાય છે.

- મશીન ભાષા
- એસેમ્બલી ભાષા
- ઉચ્ચસ્તરીય ભાષા

મશીન ભાષા (Machine Language)

આપણે વિચારીએ છીએ કે, કોમ્પ્યુટરો ખૂબ જટિલ હોય છે પરંતુ ખરેખર તેમનો આધાર ખૂબ સરળ છે. તે વિજળીના ખ્યાલ ચાલુ અથવા બંધ પર આધારિત છે. ચાલુ/બંધ, હા/ના, બે તફાવતની સિસ્ટમ વગેરે પરથી બાઈનરી સિસ્ટમના અંકેનો ઉપયોગ કરી ડેટાને રજૂ કરવાની પદ્ધતિ વિકસાવવામાં આવી છે. બાઈનરી સિસ્ટમ બે અંકો 0 અને 1 પર આધારિત છે.

તેનાથી વિપરીત, આપણે બધા દશાંશ પદ્ધતિમાં 0 થી 9 અંકોનો ઉપયોગ કરીએ છીએ. દશાંશ પદ્ધતિના અંકો 1,2 અને 3 એ બાઈનરી સિસ્ટમમાં અનુક્રમે 1,10 અને 11 દ્વારા રજૂ થાય છે. અક્ષરોને પણ બાઈનરી સિસ્ટમમાં લખવામાં આવે છે. એક પ્રણાલીમાં અક્ષર છ ને 1000001 તરીકે રજૂ કરવામાં આવે છે. અલ્પવિરામ, સેમીકોલન્સ અને અન્ય વિશિષ્ટ અક્ષરો પણ 0 અને 1 ના જુમખાઓ દ્વારા રજૂ કરવામાં આવે છે.

કોમ્પ્યુટરની શરૂઆતના દિવસોમાં મશીન જેવા કે ઈદ્રિંછઝ, જેવાં વેક્યુમ ટ્યૂબ વાપરવામાં આવતી, જેમાં વાસ્તવમાં 1/ 0 બાયનરી સ્ટેટને અનુલક્ષીને ટ્યૂબને પ્રગટેલી અથવા અપ્રકાશિત જોઈ શકાતી, સ્વીચ ચાલુ અથવા બંધ રહેતી.

વધુમાં તે શરૂઆતના દિવસોમાં સોફ્ટવેર જેવી કોઈ વસ્તુ ન હોતી. ત્યાં ફક્ત વિદ્યુત ચાલુ-બંધ કરવા માટેની સ્વીચો સાથેનું હાર્ડવેર હતું. જ્યારે પણ પ્રોગ્રામ રન કરવાની જરૂર પડતી ત્યારે બધી સ્વીચોને ગોઠવવી પડતી, કેટલીકવાર 6000 કરતાં પણ વધુ સ્વીચો ગોઠવવી પડતી. પછી આગામી પ્રોગ્રામ માટે બધી સ્વીચોને ફરીથી ગોઠવવી પડતી જેમાં અઠવાડિયા જેટલો સમય લાગી શકતો હતો.

તે દિવસોથી, મશીન સ્વીચો મશીન પ્રોગ્રામિંગથી બદલાઈ છે, પ્રોગ્રામ કે જેમાં વાક્યો 0 અને 1 ના 1 બનેલા છે. તેઓ ઈલેક્ટ્રીકથી સ્વીચો ગોઠવે છે, જેમાં 0 એ બંધ, 1 એ ચાલુ એવી રીતે રજૂ કરાય છે. આનાથી એક પ્રોગ્રામથી બીજો પ્રોગ્રામ ખૂબ સરળતાથી બદલાય છે.

મશીન ભાષામાં સીધુ પ્રોગ્રામિંગ (સૌથી નીચા સ્તરની પ્રોગ્રામિંગ ભાષા) કે જેમનાં સૂચનાઓ 0 અને 1 ના તરીકે રજૂ થાય છે, તે ખૂબ જ જટિલ અને સમય માંગી લે છે. પ્રોગ્રામરને દરેક માહિતીનું જબરદસ્ત ધ્યાન રાખવું છે. આ ઉપરાંત તેઓને કોમ્પ્યુટરની ટેકનિકલ કામગીરી સમજાવવી જોઈએ. ઉદાહરણ તરીકે, પ્રોગ્રામની એક લાઈનનો વિચાર કરો જેમાં બે નંબરનો ગુણાકાર કરેલ છે.

11110010 01110011 11010010 00010000 01110000 00101011

જોઈ શકાય છે કે આ પ્રકારના કોડની સાથે બધા કામ કરી શકે નહિ.

મશીન કોડમાં પ્રોગ્રામિંગ કરીએ તો અન્ય પ્રોગ્રામિંગ કરતાં એક ફાયદો છે. તેનો અમલ ખૂબ જ ઝડપી અને કાર્યક્ષમ છે કારણ કે કોમ્પ્યુટર એ મશીન કોડને સીધો જ સ્વીકારી લે છે. જો કે આ સ્તરે કામ કરતાં જટિલના ઉપરાંત મશીન ભાષામાં ગંભીર ગેરલાભ છે - મશીન ભાષામાં કોઈ પણ પ્રમાણભૂત નથી. અલગ અલગ મશીન માટે મશીન ભાષા અલગ અલગ હોય છે.

આ ભાષાઓ મશીન આધારિત હોય છે, અને કોઈ એક કોમ્પ્યુટર મોડેલ માટે લખાયેલો મશીન ભાષાનો પ્રોગ્રામ મોટે ભાગે બીજા કોમ્પ્યુટર મોડેલ પર કામ કરશે નહિ. ચોક્કસ કોમ્પ્યુટર ઉત્પાદકો દ્વારા ઘણા પ્રોગ્રામ અને સબરૂટીન મશીન ભાષામાં લખીને કોમ્પ્યુટરની સાથે આપવામાં આવે છે, થોડાક એપ્લીકેશન પ્રોગ્રામ વપરાશકર્તા દ્વારા મશીન ભાષામાં લખવામાં આવે છે.

એસેમ્બલી ભાષા (Assembly Langauge)

ઉપરોક્ત ચર્ચા પરથી એવું લાગે છે 0 અને 1 ની સાથે કામ કરતાં લોકો પોતે જ સાયફર બની જાય. 1950 માં, પ્રોગ્રામિંગની જટિલતા ઘટાડવા માટે અને કેટલાક સ્ટાન્ડર્ડ આપવા માટે એસેમ્બલી ભાષા વિક્સાવેલી. એસેમ્બલી ભાષાઓ સાંકેતીક ભાષાઓ તરીકે પણ ઓળખાય છે. જે સંક્ષેપ અથવા મેમોનીક કોડનો ઉપયોગ કરે છે. આ કોડ 0 અને 1 કરતાં સરળતાથી યાદ રહે છે.

11110010 01110011 11010010 00010000 01110000 00101011

આ જ વાક્ય એસેમ્બલી ભાષામાં નીચે મુજબ લખી શકાય :

PACK 210 (8,13), 02B (4,7)

ખરેખર એસેમ્બલી ભાષાએ મશીન ભાષાને બદલતી નથી. હકીકતમાં એસેમ્બલી ભાષાનો પ્રોગ્રામ ચલાવવા માટે તેને મશીન કોડમાં ફેરવવો પડે છે. એસેમ્બલી ભાષામાં લખાયેલ પ્રોગ્રામને સોર્સ પ્રોગ્રામ તરીકે, જ્યારે મશીન ભાષામાં લખાયેલ પ્રોગ્રામને ઓબ્જેક્ટ પ્રોગ્રામ તરીકે ઓળખાય છે.

એસેમ્બલી ભાષા કોડ મશીન ભાષા કોડ માટે રૂપમાં ખૂબ સમાન છે. હકીકતમાં એસેમ્બલી ભાષાના એક વાક્ય માટે એક મશીન ભાષાનું વાક્ય અનુવાદિત થાય છે. એટલે કે 15 એસેમ્બલી ભાષાના વાક્યો માટે 15 મશીન ભાષાના વાક્યો અનુવાદિત કરવામાં આવશે. આ એકથી એક અનુવાદ હજુ કપરું હતું. જો કે, એસેમ્બલી ભાષાની સૂચનાઓ (જે મેક્રો સૂચનો કહેવાય છે.)

ઘડવામાં આવી હતી જેને એક થી એક જથ્થામાં ચલાવવામાં આવતી.

એસેમ્બલી ભાષાના ઘણા ફાયદાઓ છે :

- તેઓ મશીન ભાષાઓ કરતાં વાપરવા માટે વધુ પ્રમાણિત અને સરળ છે.
- તેઓ ખૂબ જ અસરકારક રીતે કામ કરે છે, જો કે મશીન ભાષા જેટલાં કાર્યક્ષમ નથી.
- તેને સુધારવા સરળ છે.

જો કે, તેના કેટલાક ગેરફાયદા પણ છે :

- એસેમ્બલી ભાષાના પ્રોગ્રામ સામાન્ય રીતે ઘણા લાંબા હોય છે.
- એસેમ્બલી ભાષાના પ્રોગ્રામ મશીન ભાષાઓ કરતાં ઓછા એબ્સ્ટ્રેક્ટ (Less abstract) પરંતુ જટિલ હોય છે.

ઉચ્ચ સ્તરીય ભાષાઓ (High Level Langauge)

ઉચ્ચ સ્તરીય ભાષાઓ પ્રોગ્રામરોને મદદ કરે છે, તેઓ કોમ્પ્યુટર પાસેથી કેવી રીતે કામ કરવું તેના બદલે સમસ્યા હલ કરવા માટે જરૂરી તર્ક પર વધુ ધ્યાન આપી શકે છે.

નીચેના ઉદાહરણમાં કુલ પગારની ગણતરી (Culculate Gross Pay) માટેનો પ્રોગ્રામ ત્રણ અલગ ભાષામાં આપેલ છે : મશીન, એસેમ્બલી અને કોબોલ.

મશીન ભાષા (Machine Langauge)

11110010	011100011	1101	001000010000	0111	000000101011
11110010	01110011	1101	001000011000	0111	000000101111
11111100	01010010	1101	001000010010	1101	001000011101
11110000	01000101	1101	001000010011	0000	000000111110
11110011	01000011	0111	000001010000	1101	001000010100
10010110	11110000	0111	000001010100		

એસેમ્બલી ભાષા (Assembly Langauge)

PACK 210 (8,13), 02B (4,7)

PACK 218 (8,13), 02F (4,7)

MP 212 (G,13), 21 D(3,13)

SRP 213 (5,13), 03E (0),5

UNPK 050 (5,7), 214 (4,13)

IO 054 (7), X 'FO'

કોબોલ (COBOL)

MULTIPLUY HOURS-WORKED BY PAY-RATE

GIVING GROSS PAY ROUNDED

3.4 ઉચ્ચ સ્તરીય ભાષાના પ્રકારો (Type of high level langauge)

ભાષાઓને પેઢી તરીકે જોવાય છે, મશીન ભાષાને પ્રથમ પેઢીની અને એસેમ્બલી ભાષાને બીજી પેઢીની ભાષા તરીકે ઓળખાય છે. એસેમ્બલી સ્તરની ઉપરની બધી ભાષાઓને ઉચ્ચ સ્તરીય ભાષાઓ તરીકે ઓળખવામાં આવે છે. અહીં આપણે ઉચ્ચ સ્તરીય ભાષાની પેઢીના ત્રણ પેટા વિભાગ કરીશું.

- *પ્રક્રિયા લક્ષી* (Procedural-Oriented) *અથવા ત્રીજી પેઢીના*
- *સમસ્યા લક્ષી* (Problem-Oriented) *અથવા ચોથી પેઢીના*
- *નેચરલ અથવા પાંચમી પેઢીના*

પ્રક્રિયાગત લક્ષી ભાષાઓ (Procedural-Oriented Langauge)

*ઉચ્ચ સ્તરીય ભાષાઓને તેઓ સામાન્ય સમસ્યાઓ અથવા ચોક્કસ સમસ્યાઓ ઉકેલવા માટે વપરાય તે મુજબ વર્ગીકૃત કરવામાં આવે છે. સામાન્ય હેતુઓ માટેની પ્રોગ્રામિંગ ભાષાઓને પ્રક્રિયાગત (*Procedural*) ભાષાઓ અથવા ત્રીજી પેઢીની ભાષાઓ કહેવાય છે. પાસ્કલ, બેઝીક, કોબોલ અને ફોરટ્રાન એ ત્રીજી પેઢીની ભાષાઓ છે. જે સમસ્યાની તર્ક અને પ્રક્રિયાઓને ધ્યાનમાં રાખી બનાવાઈ છે. તેમની સુગમતાને લીધે પ્રક્રિયાગત ભાષાઓ વિવિધ સમસ્યાઓ ઉકેલવા માટે સમર્થ છે.*

પ્રક્રિયાગત ભાષાઓના મશીન અને એસેમ્બલી ભાષાઓ કરતાં ઘણા ફાયદા છે.:

- *પ્રોગ્રામના નિવેદનો અંગ્રેજી ભાષાને મળતાં હોવાથી કામ કરવામાં સરળતા રહે છે.*
- *તેની અંગ્રેજી પ્રકારની પ્રકૃતિના લીધે, આપેલી સમસ્યાના હલ માટેનો પ્રોગ્રામ બનાવવા ઓછો સમય જોઈએ છે.*
- *એક વખત પ્રોગ્રામ બન્યા પછી સમજવામાં અને સુધારવામાં સરળ છે.*
- *પ્રક્રિયાગત ભાષાઓ મશીન આધારિત હોય છે.*

તેમ છતાં, પ્રક્રિયાગત લક્ષી ભાષાઓના મશીન અને એસેમ્બલી ભાષાની સરખામણીમાં કેટલાંક ગેરફાયદા છે :

- ***આ પ્રોગ્રામ ચલાવવામાં થોડા ધીમા છે.***
- ***આ ભાષા કોમ્પ્યુટરના સ્રોતોને ઓછી અસરકારક રીતે વાપરે છે.***

સમસ્યા લક્ષી ભાષાઓ અને એપ્લિકેશન બનાવનાર (Problem-Oriented langauges and Application Generators)

ત્રીજી પેઢીની ભાષાઓ જેવી કે બેઝીક અથવા પાસ્કલમાં તમારે કોમ્પ્યુટરને એક પછી એક દરેક સૂચનાઓ આપવી પડતી. ચોથી પેઢીની ભાષાઓ કે જે સમસ્યા લક્ષી ભાષાઓ તરીકે ઓળખાય છે, તેઓ ઉચ્ચ સ્તરીય ભાષાઓ છે જે કોઈ ચોક્કસ સમસ્યાઓ ઉકેલવા માટે અથવા ચોક્કસ એપ્લીકેશન બનાવવા માટે બનાવાઈ છે. તમારે ત્યાં પહોંચવા માટે એક પછી એક કાર્યવાહી કરવાના બદલે તમે શું કરવા માંગો છો તેના માટે તમને સક્ષમ બનાવે છે.

ચોથી પેઢીની ભાષાઓને અનેક પ્રકારનાં એપ્લીકેશન વિકાસ સાધનોમાં* (tools) *વર્ગીકૃત કરી શકાય.

- *પર્સનલ કોમ્પ્યુટર એપ્લીકેશન સોફ્ટવેર*
- *ક્વેરી ભાષા અને રિપોર્ટ જનરેટર*
- *નિર્ણય આધારીત પધ્ધતિ અને નાણાંકીય આયોજન ભાષાઓ*
- *એપ્લીકેશન જનરેટર*

પર્સનલ કોમ્પ્યુટર એપ્લીકેશન સોફ્ટવેર (Personal Computer Applications Software)

તમે PC *માટેના વિવિધ એપ્લીકેશન સોફ્ટવેર વિશે જાણતાં હશો, પણ આપણે ખાસ કરીને ચિંતિત છીએ વર્ડ પ્રોસેસરો, સ્પ્રેડશીટ્સ, ડેટાબેઝ મેનેજમેન્ટ, ધંધાકીય ગ્રાફિક્સ અને સંકલિત પેકેજો. સ્પ્રેડશીટ્સ, ફોક્પ્રો* (Foxpro) *અથવા પાવર પોઈન્ટનું જ્ઞાન તમને તમારી એપ્લીકેશન બનાવવામાં મદદ કરે છે.*

ક્વેરી ભાષા અને રિપોર્ટ જનરેટર (Query Langauges and Report Generators) :

ક્વેરી ભાષા જેઓ પ્રોગ્રામર નથી તેવા લોકોને કેટલાક સિલેક્શન આદેશો (Commands) *નો ઉપયોગ કરીને ડેટાબેઝમાં સર્ચમાં મદદ કરે છે. ઉદાહરણ તરીકે, એરલાઈન અથવા રેલ્વે આરક્ષણ કર્મચારીઓ ટિકિટની જાણકારી માટેની ક્વેરી ભાષાનો ઉપયોગ કરે છે. રિપોર્ટ જનરેટર એ એવા લોકો માટે બનાવેલ છે જેઓને સરળતાથી અને ઝડપથી અહેવાલો (રિપોર્ટ) બનાવવા છે.* QBE, SQL, RPG *વગેરે ક્વેરી ભાષાઓ અને રિપોર્ટ જનરેટરના ઉદાહરણો છે.*

નિર્ણય આધારીત પધ્ધતિ અને નાણાંકીય આયોજન ભાષાઓ (Decision Support Systems and Financial Planning Langauges)

નિર્ણય આધારીત પધ્ધિઓ મેનેજરોને નિર્ણય લેવામાં મદદ કરવા માટે રચાયેલ સોફ્ટવેર છે. નાણાંકીય આયોજન ભાષાઓ એક પ્રકારની નિર્ણય આધારીત પધ્ધતિઓ છે જે ગાણિતિક આંકડાકીય અને આગાહી કરવા માટે કાર્યો કરે છે. બંન્ને પ્રકારની ભાષાઓ બિઝનેસ મોડેલ મેનેજમેન્ટમાં (ધંધામાં) જટિલ સમસ્યાનો હલ કરવા મદદ કરે છે.

એપ્લીકેશન જનરેટર (Application Generator)

એપ્લીકેશન જનરેટર એ સોફ્ટવેર સિસ્ટમનું બનેલું છે. જેમાં પ્રોગ્રામના ઘણા મોડ્યુલ હોય છે જેઓ અલગ અળગ કાર્યો કરવા બનાવાયેલા હોય છે. આનાથી પ્રોગ્રામર અથવા વપરાશકર્તા

ચોક્કસ એપ્લીકેશન માટે કયા ફંકશનની જરૂર છે તે જણાવે છે, અને સિસ્ટમ યોગ્ય મોડ્યુલ પસંદ કરી પ્રોગ્રામને ચલાવે છે જે વપરાશકર્તાની જરૂરિયાત પુરી કરે છે.

કોષ્ટક 3.1 ત્રીજી અને ચોથી પેઢી (3GL and 4GL) વચ્ચેના મુખ્ય તફાવતનો સારાંશ આપે છે.

કોષ્ટક 3.1

3 GL અને 4 GL વચ્ચેનો તફાવત

ત્રીજી પેઢીની ભાષાઓ **Third Generation Langauges-3GL**	**ચોથી પેઢીની ભાષાઓ** **Fourth Generation Langauges-4GL**
● વ્યવસાયિક પ્રોગ્રામરોના ઉપયોગ માટે બનાવાઈ છે. ● કામ કેવી રીતે કરવું તેની માહિતી જોઈએ. ● બધા જ વિકલ્પો સ્પષ્ટ કરવા પડે. ● ખૂબ જ મોટી સંખ્યામાં પ્રક્રિયાગત સૂચનો જોઈએ. ● કોડને વાંચો, સમજવો અને સાચવવો અઘરો છે. ● ભાષાઓ જથ્થાબંધ કાર્યો માટે બનાવાઈ છે. ● શિખવામાં અઘરી છે. ● ડીબગ કરવું અઘરું છે. ● સામાન્ય રીતે ફાઈલ લક્ષી છે.	● પ્રોગ્રામિંગ જાણતા નથી તેવા લોકો તેમજ વ્યવસાયિક પ્રોગ્રામ વાપરી શકે છે. ● શું કામ કરવું છે તેની માહિતી જોઈએ (સિસ્ટમ નક્કી કરશે કે કામ કેવી રીતે કરવું) ● મૂળભૂત વિકલ્પો આંતરિક છે, વપરાશકર્તાને તે વિકલ્પો સ્પષ્ટ કરવા પડતા નથી. ● થોડા જ સૂચનો જોઈએ. ● અંગ્રેજી પ્રકારના કમાન્ડ હોવાથી સમજવામાં અને સાચવવામાં સરળ છે. ● ભાષાઓ મુખ્યત્વે અને ઓનલાઈન વપરાશ માટે બનાવાઈ છે. ● શિખવામાં સરળ છે. ● ડીબગ કરવું સરળ છે. ● સામાન્ય રીતે ડેટાબેઝ લક્ષી છે.

કુદરતી ભાષાઓ (Natural Langauges)

કુદરતી ભાષાઓ વિકાસના સ્તર પર છે, પરંતુ તેઓ વચન આપે છે તેઓ ખૂબ જ અસરકારક રહેશે, ખાસ કરીને કૃત્રિમ બુધ્ધિ (Artificial Intelligence) અને એક્સ્પર્ટ સિસ્ટમના વિસ્તારમાં. કુદરતી ભાષાઓની બે લાક્ષણિકતાઓ છે :

- તેઓને એવી રીતે બનાવવામાં આવ્યા છે જેથી મનુષ્ય અને કોમ્પ્યુટર વચ્ચેનું જોડાણ વધુ કુદરતી બને, માણસની જેમ.
- તેઓને કોમ્પ્યુટર વધુ ચાલાક બની શકે તેવી રીતે બનાવેલ છે - ખરેખર તે જુની માહિતીનો ઉપયોગ કરીને યાદ રાખવાની અને શિખવાની પ્રક્રિયાને/સુધારવાનું અનુકરણ કરે છે.

LISP અને PROLOG એ જાણીતી કુદરતી ભાષાઓ છે.

3.5 કંમ્પાઈલર અને ઈન્ટરપ્રીટર (Compilers and Interpreters)

ઉચ્ચ સ્તરીય ભાષાને કોમ્પ્યુટરમાં ચલાવવા માટે તેને મશીન ભાષામાં બદલવી પડે છે. અનુવાદકો (transtators)બે પ્રકારના હોય છે - કંમ્પાઈલર અને ઈન્ટરપ્રીટર. ઉચ્ચ સ્તરીય ભાષાઓ કંમ્પાઈલ્ડ ભાષાઓ અથવા ઈન્ટરપ્રીટેડ ભાષાઓ કહેવાય છે.

કંમ્પાઈલ્ડ ભાષામાં ભાષાંતર પ્રોગ્રામ ચલાવાવમાં આવે છે જે ઉચ્ચ સ્તરીય ભાષાના પ્રોગ્રામ (જેને સોર્સ કોડ કહેવાય છે) ને મશીન ભાષાના કોડમાં બદલે છે. આ ભાષાંતરની પ્રક્રિયાને કંમ્પાઈલેઝશન કહેવાય છે.

મશીન ભાષાના કોડને ઓબ્જેક્ટ કોડ કહે છે અને તેને સેવ કરી તાત્કાલિક અથવા પછીથી ચલાવી શકાય છે. કેટલીક વધુ વપરાતી કંમ્પાઈલ્ડ ભાષાઓ COBOL, C, C++, FORTRAN વગેરે છે.

ઈન્ટરપ્રીટેડ ભાષામાં, ભાષાંતર પ્રોગ્રામ એ પ્રોગ્રામના દરેક વાક્યને રન (Execute) થાય તે પહેલાં જ મશીન કોડમાં ફેરવે છે. દરેક વાક્યનું એક પછી એક પહેલાં ભાષાંતર થાય છે અને પછી તરત જ રન કરવામાં આવે છે.

કંમ્પાઈલ્ડ ભાષાની જેમ કોઈ ઓબ્જેક્ટ કોડ બનતો નથી અને તેનું કંમ્પાઈલેશન પણ થતું નથી. BASIC એ સામાન્ય રીતે વપરાતી ઈન્ટરપ્રીન્ટેડ ભાષા છે.

કંપાઈલ્ડ ભાષાએ ઈન્ટરપ્રીટેડ ભાષાઓ કરતાં સારી છે કારણ કે તેઓ એક વખત ઓબ્જેક્ટ કોડ મળી જાય પછી ઝડપથી અને વધુ અસરકારક રીતે ચલાવી શકાય છે. બીજી બાજુ, ઈન્ટરપ્રીટેડ ભાષાને ઓબ્જેક્ટ કોડ બનાવવાની જરૂર નથી, તેથી સામાન્ય રીતે તેઓ બનાવવા અને ચકાસવામાં સરળ છે.

પાઠગત પ્રશ્નો 3.1

1. મશીન ભાષા શું છે ?
2. ઉચ્ચ સ્તરીય ભાષાના ફાયદાઓ ટૂંકમાં સમજાવો.
3. નીચેના વાક્યો સાચા છે કે ખોટાં તે જણાવો :
   (અ) મશીન ભાષાએ મશીન પર આધારિત છે.
   (બ) એસેમ્બલી ભાષાએ બીજી પેઢીની ભાષા છે.
   (ક) ઉચ્ચ સ્તરીય ભાષામાં લખાયેલ પ્રોગ્રામને ઓબ્જેક્ટ પ્રોગ્રામ કહે છે.
   (ડ) પોર્ટેબીલીટીએ ઉચ્ચ સ્તરીય ભાષાનો એક વિશેષતા છે.
   (ઈ) ક્વેરી ભાષાઓ જે માણસો પ્રોગ્રામર નથી તેઓને કેટલાક સીલેક્શન કમાન્ડનો ઉપયોગ કરીને ડેટાબેઝમાં શોધવામાં મદદ કરે છે.

3.6 સોફ્ટવેર એટલે શું ? (What is Software ?)

આપણે જાણીએ છીએ કે કોમ્પ્યુટર એક મશીન છે જે વપરાશકર્તાની સૂચના વગર કંઈ કરી શકતું નથી. કોમ્પ્યુટર પાસે કોઈ ચોક્કસ કાર્ય કરાવવા માટે તમારે સૂચનાઓ આપવી પડે. આ પ્રમાણે ચોક્કસ હરોળમાં ગોઠવાયેલ સૂચનાઓના સમૂહને કોમ્પ્યુટર પ્રોગ્રામ કહે છે. સોફ્ટવેરએ કોમ્પ્યુટર પ્રોગ્રામનો સમૂહ છે જે હાર્ડવેર (કોમ્પ્યુટર સિસ્ટમ) પાસે જરૂરી કાર્યો કરાવે છે. હાર્ડવેર એટલે કોમ્પ્યુટર સિસ્ટમનો ભૌતિક ઘટક. આપણે કહી શકીએ કે હાર્ડવેરને જ્યાં સુધી સૂચનો ના મળે ત્યાં સુધી તે કોઈ ગણતરી, સરખામણી અથવા વ્યવહાર કરી શક્તા નથી. આ સૂચનાઓ કોમપ્યૂટરમાં કામગીરીમાં અગત્યનો ભાગ ભજવે છે. કોમ્પ્યુટર પરની સમસ્યાઓના નિરાકરણ માટે લખાયેલા સૂચનાઓના સંપૂર્ણ જથ્થાને સોફ્ટવેર કહે છે.

3.7 સોફ્ટવેરના પ્રકારો (Types of Software)

કોમ્પ્યુટર સોફ્ટવેરને સામાન્ય રીતે બે મુખ્ય વર્ગમાં વહેંચી શકાય :-

(i) *સિસ્ટમ સોફ્ટવેર*

(ii) *એપ્લીકેશન સોફ્ટવેર*

3.7.1 સિસ્ટમ સોફ્ટવેર (System Software)

સિસ્ટમ સોફ્ટવેર માં કોમ્પ્યુટર માટે લખવામાં આવેલા સામાન્ય પ્રોગ્રામનો સમાવેશ થાય છે. તેમાં કોમ્પ્યુટરના ઉત્પાદકો દ્વારા લખવામાં આવેલા પ્રોગ્રામ અને દસ્તાવેજો હોય છે. આવા પ્રોગ્રામ કોમ્પ્યુટરમાં હંમેશના માટે હોય છે. આ સોફ્ટવેરના મુખ્ય હેતુઓ છે :

(i) *હાર્ડવેરના ઉપયોગની કાર્યક્ષમતા વધારવી, અને*

(ii) *કોમ્પ્યુટરનો ઉપયોગ સરળ બનાવવો.*

પ્રથમ હેતુનો ખ્યાલ તેના ઘટકો પરથી આવે છે જેમાંથી એક ઘટક સામાન્ય રીતે ઓપરેટિંગ સિસ્ટમ તરીકે ઓળખાય છે.

*સિસ્ટમ સોફ્ટવેર એટલે મશીન માટે સૂચનાઓનો સમૂહ જે એપ્લીકેશન સોફ્ટવેરને સમજવા અને ચલાવવા માટે વપરાય છે. ઉ.દા. તરીકે, ભાષાનું રૂપાંતર કરનાર (*Langauge Translators*) જે કંમ્પાઈલર અને ઈન્ટરપ્રીટર કહેવાય છે, ઓપરેટિંગ સિસ્ટમ અને ખાસ હેતુઓ માટેના સોફ્ટવેર.*

ભાષાનું રૂપાંતર (Langauge Translators)

ભાષાં રૂપાંતર એ એવું સોફ્ટવેર છે કે જે ઉપભોક્તા દ્વારા લખવામાં આવેલા કોમ્પ્યુટર પ્રોગ્રામને મશીન સમજી શકે તેવી ભાષામાં રૂપાંતર કરે છે.

પ્રોગ્રામીંગના નીચેના લેવલે બાયનરી નંબરો 0 *અને* 1 *વપરાય છે જે ઈલેક્ટ્રોનિક સર્કિટને સીધી રીતે સમજાય છે. જે પ્રોગ્રામીંગમાં માત્ર* 0 *અને* 1 *આંકડા આપેલા હોય તેને મશીન ભાષાના પ્રોગ્રામ કહેવાય છે. મશીન ભાષાના પ્રોગ્રામમાં* 0 *અને* 1 *હોવાથી તેવા પ્રોગ્રામને લખવા અથવા સમજવા બહુ મુશ્કેલ હોય છે.*

એસેમ્બલી ભાષાઓ મશીન ભાષાઓ પર નોંધપાત્ર સુધારો દર્શાવે છે. એસેમ્બલી ભાષાનો પ્રોગ્રામ મેમોનિક (Memonic) સંકેતો જેવા કે ADD, STORE વગેરેનો ઉપયોગ કરીને લખવામાં આવે છે. જ્યારે તેની મશીન ભાષામાં માત્ર બાયનરી અંકો દર્શાવાય છે. આથી એસેમ્બલી ભાષામાં પ્રોગ્રામીંગ સરળ છે. તેને ચલાવવા માટે તેને મશીન ભાષામાં રૂપાંતર કરવો જરૂરી છે. આ ભાષાંતર એસેમ્બલર દ્વારા થાય છે. મશીન ભાષા અને એસેમ્બલી ભાષા, બંન્નેમાં લખાયેલ પ્રોગ્રામ મશીન આધારિત છે.

ઉચ્ચ સ્તરીય ભાષાના પ્રોગ્રામને તેને ચલાવતા પહેલાં મશીન ભાષામાં રૂપાંતર કરવામાં આવે છે. આ ભાષાંતર કંમ્પાઈલર અથવા ઈન્ટરપ્રીટર દ્વારા કરાય છે. કંમ્પાઈલર એ એવો ભાષાંતર કરવાનો પ્રોગ્રામ છે જે પહેલાં ઉચ્ચ સ્તરીય ભાષામાં લખાયેલ આખો પ્રોગ્રામ વાંચીને તેને મશીન કોડમાં રૂપાંતર કરે છે.

બીજી તરફ ઈન્ટરપ્રીટર એ ભાષાંતર કરે છે જેમાં પહેલાં પ્રોગ્રામના એક વાક્યને મશીન કોડમાં ફેરવાય છે, તેને રન કરાય છે અને પછી તેવી જ રીતે બીજા વાક્યને ભાષાંતર કરી રન કરાય છે. આ પ્રક્રિયા ત્યાં સુધી ચાલે છે જ્યાં સુધી પ્રોગ્રામનો અંત ન આવે અથવા કોઈક ભૂલ પડે.

ઓપરેટિંગ સિસ્ટમ (Operating System)

ઓપરેટિંગ સિસ્ટમ ખૂબ જ મહત્ત્વનું સિસ્ટમ સોફ્ટવેર છે અને તે કોમ્પ્યુટર સિસ્ટમ ચલાવવા માટે આવશ્યક છે. ઓપરેટિંગ સિસ્ટમ એ કોમ્પ્યુટરના સ્રોતોને ખૂબ જ કાર્યક્ષમતાથી સંચાલન કરે છે, કાર્યો કરવાના સમયને નિયંત્રિત પણ કરે છે અને ડેટા અને સૂચનાઓના પ્રવાહનું ઈનપુટ/આઉટપુટ સાધનો અને મેમરી વચ્ચે સંચાલન કરે છે.

સૌ પ્રથમ ઓપરેટિંગ સિસ્ટમ જેને બેચ સંચાલિત ઓપરેટિંગ સિસ્ટમ પણ કહે છે તે બીજી પેઢીના કોમ્પ્યુટર માટે બનાવાઈ હતી. આ ઓપરેટિંગ સિસ્ટમ કાર્યોને એક પછી એક ચલાવે છે. જે કાર્યોના જથ્થા સાથે આવેલી હોય છે. સેન્ટ્રલ પ્રોસેસિંગ યુનિટને કાર્યના ચાલન સમયે જ વ્યસ્ત રખાય છે અને ઈનપુટ/આઉટપુટ કાર્યો વખતે કોઈ કામ કરતું નથી. મલ્ટી પ્રોગ્રામીંગ ઓપરેટિંગ સિસ્ટમ એક સાથે ઘણા બધા કાર્યોનું સંચાલન કરે છે, તેમાં ઈનપુટ/આઉટપુટ અને કાર્યો એક સાથે થાય છે.

બીજી ઓપરેટિંગ સિસ્ટમ આજે બહુ પ્રખ્યાત છે તે "મલ્ટી પ્રોસેસિંગ ઓપરેટિંગ સિસ્ટમ" અને "રીયલ ટાઈમ ઓપરેટિંગ સિસ્ટમ." મલ્ટી પ્રોસેસિંગ ઓપરેટિંગ સિસ્ટમ ઘણા CPU નો ઉપયોગ કરીને એક સાથે ઘણા કાર્યો કરે છે.

રીયલ ટાઈમ ઓપરેટિંગ સિસ્ટમ એ એવી ઓપરેટિંગ સિસ્ટમ છે જે સમયનાં નિયંત્રણ સાથે કામ કરે છે. ડેટાને મેળવી અને તેના પર ઝડપથી પ્રક્રિયા કરે છે જે ચાલી રહેલા કાર્ય માટે આઉટપુટ, નિયંત્રણ અને નિર્દેશ આપે છે. રેલ્વે, હોટલ, એરલાઈન વગેરેના આરક્ષણ માટે વપરાતી એપ્લિકેશન રીયલ ટાઈમની ઉદાહરણો છે.

ઓપરેટિંગ સિસ્ટમના કાર્યોને શાળાના પ્રિન્સિપાલના કાર્યો સાથે સરખાવી શકાય. પ્રિન્સિપાલ એવું વાતાવરણ પૂરું પાડે છે જેથી બીજા કામદારો અગત્યના કામ કરી શકે. પ્રિન્સિપાલ આ કાર્યો હાજર સ્રોતની ફાળવણી દ્વારા કરે છે અને જુદા જુદા વર્ગો માટે વર્ગખંડો પૂરા પાડવા વગેરે. એવી જ રીતે ઓપરેટિંગ સિસ્ટમ કોમ્પ્યુટરના કાર્યોનું અને ઈનપુટ-આઉટપુટ સાધનોના કાર્યો પર ધ્યાન

રાખે છે. ઓપરેટિંગ સિસ્ટમ પ્રોગ્રામ કોમ્પ્યુટરને વાપરનાર અને કોમ્પ્યુટરના ભાગોને એકબીજા સાથે જોડનાર માધ્યમ તરીકે કાર્ય કરે છે, અને વપરાશકર્તાના પ્રોગ્રામને કાર્યરત કરવામાં મદદરૂપ બને છે. ઓપરેટિંગ સિસ્ટમના મુખ્ય કાર્યો નીચે મુજબ છે :

1. વપરાશકર્તાની ઓળખાણ અંતે એણે ઉપયોગ કરેલા સ્ત્રોતોને સાચવી રાખે છે જેથી પરવાનગી વગરનો વપરાશકર્તા એ કોમ્પ્યુટરનો ઉપયોગ કરી શકે નહીં.

2. ઓપરેટિંગ સિસ્ટમ ઘણા બધા વપરાશકર્તાઓ વચ્ચે કોમ્પ્યુટરના પ્રોગ્રામોને એક સાથે કાર્યરત કરવા માટેની પરવાનગી કોમ્પ્યુટરના સ્ત્રોતોના વિભાજન દ્વારા આપવામાં આવે છે. આ રીતની કાર્યશૈલીને મલ્ટી પ્રોગ્રામીંગ કહે છે. એક કરતાં વધુ પ્રોગ્રામનું સંયોજન કોમ્પ્યુટરની બધી જ મેમરી રોકી રાખે છે, જેમાં બધા જ સાધનો સક્રિય રહે છે, અને કન્ટ્રોલ યુનિટ અને છએં સતત કાર્યરત રહે છે, આવી રીતે કોમ્પ્યુટર હાર્ડવરની ક્ષમતામાં વધારો થાય છે.

3. માણસની દખલગીરી કર્યા વગર એક પછી એક પ્રોગ્રામોના ભાગોને સક્રિય કરી શકાય છે.

4. વપરાશકર્તાની માહિતી અને પ્રોગ્રામોનું રક્ષણ કરી શકાય છે.

5. પ્રાથમિક મેમરી અને ગૌણ મેમરી, અને ઈનપુટ-આઉટપુટ સાધનો વચ્ચે માહિતી અને પ્રોગ્રામની હેરફેરનું ધ્યાન રાખે છે.

6. યોગ્ય ટ્રાન્સલેટરની પસંદગી માટે પ્રોગ્રામ પૂરો પાડે છે.

7. વપરાશકર્તાના પ્રોગ્રામની અંદરની ભૂલોને પકડવામાં આવે અને તેને સુધારવામાં મદદ કરે છે.

ઓપરેટિંગ સિસ્ટમ આદેશના (કમાન્ડ)ના નિશ્ચિત સમૂહને સમજે છે. આ આદેશોના સમૂહને જોબ કન્ટ્રોલ ભાષા (JCL) કહેવાય છે. JCL આદેશોના ઉપયોગ કોમ્પ્યુટર વપરાશકર્તાઓ દ્વારા ઓપરેટિંગ સિસ્ટમને તેની જરૂરીયાત જણાવવા માટે થાય છે. જે ઓપરેટિંગ સિસ્ટમ માઈક્રો કોમ્પ્યુટર સાથે વપરાય છે તેને CP/M (Control Program For Microprocessor) કહેવાય છે. યુનિક્ષ (Unix) એકબીજી ઓપરેટિંગ સિસ્ટમ છે જેની લોકપ્રિયતા વધી રહી છે જે વિવિધ પ્રકારના મશીનો માટે ઉપલબ્ધ છે. (Unix ને બેલ લેબોરેટરી દ્વારા વિકસાવવામાં આવી હતી. DOS ઓપરેટિંગ સિસ્ટમનો ઉપયોગ સામાન્યમાં થાય છે.

કોમ્પ્યુટરના ઉપયોગને સરળ બનાવવાનો બીજો ધ્યેય, જો વપરાશકર્તાને મશીન ભાષા સિવાયની ભાષામાં પ્રોગ્રામ લખી શકે, તો મેળવી શકાય. કોમ્પ્યુટર ફક્ત મશીન ભાષા સમજી શકે છે. જો કોઈ બીજી ભાષા વાપરી હોય તો તેને ચલાવતાં પહેલાં મશીન ભાષામાં ભાષાંતર કરવું પડે છે. કોમ્પ્યુટરનું મહત્ત્વનું એક પાસું છે કે આવા ભાષાંતર કોઈ બીજા પ્રોગ્રામનો ઉપયોગ કરીને જાતે કરી શકે છે. કોમ્પ્યુટર બનાવનાર અલગ અલગ ભાષાના મશીન ભાષામાં રૂપાંતર કરવા માટે ઘણા પ્રોગ્રામ આપે છે. આવા પ્રોગ્રામને ટ્રાન્સલેટર અથવા કંમ્પાઈલર કહેવાય છે અને તેઓ સિસ્ટમ સોફ્ટવેરનો વિભાગ બને છે.

આપણે એકલા કોમ્પ્યુટર હાર્ડવેરની બહુ ઓછી વાતો કરીએ છીએ. હાર્ડવેર અને સોફ્ટવેર બંન્ને સાથે મળીને કોમ્પ્યુટર સિસ્ટમ બનાવે છે. એ વાતમાં તથ્ય છે કે સિસ્ટમ સોફ્ટવેર એ તેના હાર્ડવેર

જેટલું જ અગત્યનું છે.

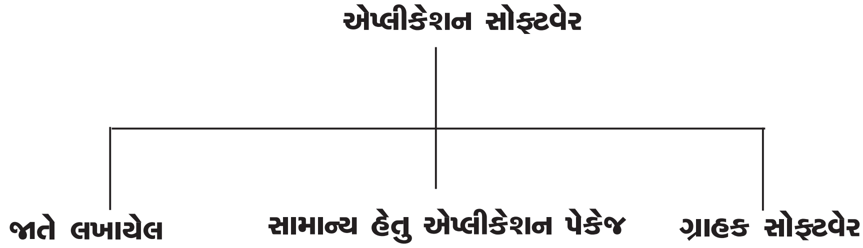
ઉપયોગી સોફ્ટવેર (Utility Software)

યુટીલીટી સોફ્ટવેર એ સિસ્ટમ સોફ્ટવેર ગણી શકાય જેનો ઉપયોગ પ્રોગ્રામ બનાવતી વખતે વારંવાર થાય છે. સોર્ટ મર્જ જેવા પ્રોગ્રામનો ઉપયોગ રેકોર્ડને સુનિશ્ચિત ક્રમમાં ગોઠવવા થાય છે. સામાન્ય રીતે આવા પ્રોગ્રામ ઉત્પાદકો દ્વારા આપવામાં આવે છે. પ્રોગ્રામર તેના પોતાના પણ યુટિલીટી સોફ્ટવેર બનાવે છે. જેને તે કોમ્પ્યુટરની સેકન્ડરી મેમરીમાં રાખે છે.

3.7.2 એપ્લીકેશન સોફ્ટવેર (Application Software)

એપ્લીકેશન સોફ્ટવેર નિશ્ચિત કાર્યો કરવા માટે લખાયેલા હોય છે જેમ કે નામું લખવા, પગાર માટેના (પેરોલ), ઈમેઈલ કરવા, રિઝલ્ટ બનાવવા વગેરે મોટા ભાગના એપ્લીકેશન સોફ્ટવેર ઉચ્ચ સ્તરીય ભાષામાં લખાયેલ છે.

ધારો કે જે કામ કરવું છે તે સાચી રીતે શોધાયેલ હોય, ધ્યાનપૂર્વક નક્કી કરેલ છે તો ભાવિ વપરાશકર્તાને નીચેનામાંથી કોઈ એપ્લીકેશન સોફ્ટવેરના સ્ત્રોતોનો વિકલ્પ મળશે.



(અ) જાતે લખાયેલ (Write it yourself): વપરાશ કર્તા દ્વારા લખાયેલ પ્રોગ્રામ સૌથી વધુ સંતોષકારક હોય છે. તે ધંધાની દરેક જરૂરિયાત પૂરી કરે છે. પ્રોગ્રામ પણ ધંધાની સાથે વિકસે છે.

(બ) સામાન્ય હેતુ એપ્લીકેશન પેકેજ (General Purpose Application Packages) : એપ્લીકેશન પેકેજ એટલે કોમ્પ્યુટર પ્રોગ્રામોનો સમૂહ જેને સામાન્ય રીતે કોઈ ચોક્કસ કામ કરવા માટે લખાયેલ હોય છે. દરેક પ્રોગ્રામ એવી રીતે લખાયેલ હોય છે જે ઘણા વપરાશકર્તાને લાગુ પડે. તેનો મુખ્ય ફાયદો છે કે તે ઘણા વપરાશકર્તા વાપરતા હોવાથી કિંમતમાં સસ્તા છે. તેનો મુખ્ય ગેરફાયદો છે કે એપ્લીકેશન પેકેજ તે ભાવી વપરાશર્તાની બધી જરૂરિયાત પૂરી પાડી શક્તા નથી.

(ક) ગ્રાહક સોફ્ટવેર (Customer Software) : વપરાશકર્તાની ચોક્કસ જરૂરિયાત પ્રમાણે બનાવેલ ખાસ કોમ્પ્યુટર પ્રોગ્રામને ગ્રાહક માટેના સોફ્ટવેર કહી શકાય. જેમ કોઈ દરજી વ્યક્તિ માટે કપડાંને ટાંકા લઈને ડ્રેસ તૈયાર કરે છે એ જ રીતે ગ્રાહક માટેના સોફ્ટવેર બનાવાય છે. આનો મુખ્ય ફાયદો છે કે તે વપરાશકર્તાની દરેક જરૂરિયાત પૂરી કરે છે. એનો મુખ્ય ગેરફાયદો છે કે તેની કિંમત સામાન્ય હેતુ એપ્લીકેશન પેકેજ કરતાં વધારે હોય છે કારણ કે તેઓ એક ખાસ ગ્રાહક માટે તૈયાર કરવામાં આવે છે.

સામાન્ય એપ્લીકેશન પેકેજ (General Application Package)

પર્સનલ કોમ્પ્યુટરના વપરાશકર્તાઓની સામાન્ય જરૂરિયાતોને શોધી અને તેને ધ્યાનમાં રાખીને સામાન્ય એપ્લીકેશન પેકેજ બનાવવામાં આવ્યા છે. જેમ કે વર્ડ પ્રોસેસર, ડેટાબેઝ પ્રોસેસિંગ, સ્પ્રેડશીટ ગણતરી, મેલ-મર્જ, પ્રેઝન્ટેશન અને ઈમેલ. આ પેકેજને એવી રીતે બનાવવામાં આવે છે કે તેઓ વાપરવામાં સરળ રહે. તેને વપરાશકર્તાને વાપરવામાં સરળ રહે તે માટે ગ્રાફિક્સવાળું વપરાશ કર્તા ઈન્ટરફેસ પૂરું પાડવામાં આવે છે. આવા પેકેજ બજારમાં સરળતાથી મળી રહે છે ને કોઈ પણ તેને ખરીદીને તેઓના કોમ્પ્યુટરમાં ઈન્સ્ટોલ (નાંખી) કરી સરળતાથી પેકેજ વાપરવાનું શરૂ કરી શકે છે.

પાઠગત પ્રશ્નો 3.2

1. સોફ્ટવેરની વ્યાખ્યા આપો.
2. સિસ્ટમ સોફ્ટવેર અને એપ્લીકેશન સોફ્ટવેર વચ્ચેનો તફાવત આપો.
3. નીચેના વાક્યો માટે સાચું અથવા ખોટું લખો.

(અ) પ્રોગ્રામ રન થાય તે પહેલાં તેને ઉચ્ચ સ્તરીય ભાષામાંથી મશીન ભાષામાં ભાષાંતર કરવું જરૂરી છે.

(બ) કોમ્પ્યુટર સિસ્ટમ ચલાવવા માટે ઓપરેટિંગ સિસ્ટમ જરૂરી છે.

(ક) JBL એટલે જોબ કોમન લેંગ્વેજ.

(ડ) એપ્લીકેશન સોફ્ટવેરને કોઈ ચોક્કસ કામ કરવા માટે લખવામાં આવે છે.

(ઈ) યુટીલીટી સોફ્ટવેરને એપ્લીકેશન સોફ્ટવેર કહી ના શકાય.

3.8 તમે શું શીખ્યા ?

આ પાઠમાં તમે વિવિધ સ્તરની ભાષાઓ અને તેના ફાયદા અને ગેરફાયદાઓ વિશે ટૂંકમાં શીખ્યા. સિસ્ટમ સોફ્ટવેર અને એપ્લીકેશન સોફ્ટવેર એમ બે પ્રકારના સોફ્ટવેરનું સમજ-સરળ ભાષામાં સમજાવ્યું છે જેથી તમે તેને સરળતાથી શીખી શકો.

3.9 પાઠ્યાંત પ્રશ્નો

1. હાર્ડવેર અને સોફ્ટવેર વચ્ચેનો તફાવત આપો.
2. મશીન ભાષાના ફાયદા અને ગેરફાયદા શું છે ?
3. સોર્સ પ્રોગ્રામ અને ઓબ્જેક્ટ પ્રોગ્રામ વચ્ચે શું તફાવત છે ?
4. એસેમ્બલી ભાષા સમજાવો. મશીન ભાષા કરતાં તેના ફાયદા શું છે ?
5. ઓપરેટિંગ સિસ્ટમની વ્યાખ્યા આપો. તેના વિવિધ કાર્યો લખો.

6. એપ્લીકેશન સોફ્ટવેર વિશે ટૂંકમાં વર્ણન કરો.

3.10 પાઠગત પ્રશ્નોના જવાબો.

પાઠગત પ્રશ્નો 3.1

1. મશીન ભાષામાં દ્વિઅંકી નંબરો (1 અને 0) ના ક્રમથી સૂચન આપવામાં આવે છે, જેનાથી કોમ્પ્યુટર સીધુ જવાબ આપી શકે. કોમ્પ્યુટર ખાલી આજ ભાષા સમજે છે.

2. ઉચ્ચ સ્તરીય ભાષાના ફાયદાઓ છે :-

   (અ) વિશ્વાસપાત્ર

   (બ) શીખવવામાં સરળ

   (ક) ભૂલો સુધારવામાં સરળ

   (ડ) સોફ્ટવેર વિકસાવવું સરળ

3. (અ) સાચું (બ) સાચું (ક) ખોટું (ડ) સાચું (ઈ) સાચું

પાઠગત પ્રશ્નો 3.2

1. સોફ્ટવેર એવા કોમ્પ્યુટર પ્રોગ્રામોનો સમૂહ છે જે હાર્ડવેરને યોગ્ય કામ કરાવે છે. સોફ્ટવેર એ સામાન્ય શબ્દ છે જે કોઈ એક પ્રોગ્રામ અથવા પ્રોગ્રામોના સમૂહને દર્શાવે છે.

2. એપ્લીકેશન સોફ્ટવેર એ પ્રોગ્રામોનો સમૂહ છે જે એપ્લીકેશનના ચોક્કસ કાર્યો કરે છે. સિસ્ટમ સોફ્ટવેર એવા પ્રોગ્રામોનો સમૂહ છે જે કોમ્પ્યુટરના ડેટાને લગતાં ઈનપુટ-આઉટપુટ કાર્યોની દેખરેખ રાખવા માટે લખાયેલ છે.

3. (અ) સાચું (બ) સાચું (ક) ખોટું (ડ) સાચું (ઈ) ખોટું

4

ઓપરેટિંગ સિસ્ટમ

4.1 પ્રસ્તાવના

તમને આજે જોવા મળતાં મોટા ભાગના કોમ્પ્યુટરો માઈક્રોસોફ્ટ વિન્ડોઝ ઓપરેટિંગ સિસ્ટમ સાથે આવે છે. તે IBM સુસંગત PCs માટે ની ઓપરેટિંગ સિસ્ટમ છે. ઓપરેટિંગ સિસ્ટમ એ પાયાનું સોફ્ટવેર છે. જે તમારૂ કોમ્પ્યુટર વાપરવાનું શક્ય બનાવે છે અને તમને જરૂરી ક્રિયાઓ કરે છે. વિન્ડોઝ 98 એ વિન્ડોઝ 95 ની સુધારેલી આવૃત્તિ છે.

4.2 હેતુઓ

આ પાઠ પછી તમે નીચેની વસ્તુ કરવાની સ્થિતિમાં હશો.

- *વિન્ડોઝ -98 ના વિવિધ લક્ષણો સમજાવી શકશો.*
- *વિવિધ ચિહ્નો (Icons) અને તેના લક્ષણો જાણી શકશો.*
- *વિન્ડોઝ -98 ના ફાઈલ મેનેજમેન્ટના લક્ષણો વ્યાખ્યાયિત કરી શકશો.*

4.3 વિન્ડોઝ - 98 ના મુખ્ય લક્ષણો

નીચેની બાબતો વિન્ડોઝ - 98 ના મુખ્ય લક્ષણો છે.

- *તે કોમ્પ્યુટર અને તમારી વચ્ચે કડી સ્થાપે છે અને તે ઈન્ટરફેશ આપે છે જેનાથી તમે કોમ્પ્યુટર સાથે વાતચીત કરી શકો છો.*
- *તે એક મુખ્ય સોફ્ટવેર તરીકે કાર્ય કરે છે જેની ઉપર વિવિધ પ્રોગ્રામ કાર્ય કરી શકે છે. દા. ત. એમ.એસ.વર્ડ, એમ. એસ.એક્સેલ, એમ.એસ.પાવર પોઈન્ટ અને એમ.એસ એકસેસ.*

- અંદરના કાર્યો જેવા કે કોમ્પ્યુટરની મેમરી, ઈનપુટ અને આઉટપુટને સંભાળે છે.
- શ્રેણીબધ્ધ ઉપયોગિતા પૂરી પાડે છે જેવા કે ખાસ કાર્યક્રમો અને સૂચનાઓ જે તમે તમારી ફાઈલો, ફોલ્ડરો અને ડિસ્કને સંભાળી શકે છે.
- GUI જે ચિત્રો, ચિહ્નો, શબ્દોના સ્ક્રિન પર ઉપયોગ કરે છે જે તમે માઉસથી નિયંત્રિત કરી શકો છો.
- બિલ્ટ ઈન કાર્યક્રમો અથવા એસેસરીઝ જેવા કે વર્ડપેડ, નોટપેડ, કેલક્યુલેટર, પેઈન્ટ પ્રોગ્રામ વગેરે.
- એક જ સમયે ઘણા ડોક્યુમેન્ટ અને સંખ્યાબંધ કાર્યક્રમો તેની વિન્ડોમાં ચલાવે છે.
- WWW એ વિન્ડોમાં ગમે ત્યાંથી સરળતાથી ઉપયોગ કરી શકાય છે. તમે ઈન્ટરનેટ ઉપર વાતચીત કરી શકો છો. વિન્ડો સાથે સમાવિષ્ટ ઈન્ટરનેટ સાધનોનો ઉપયોગ કરીને તમે ઈમેઈલ, ચેટ અને ન્યૂઝ જોઈ શકો છો.
- DVD અને ડિઝીટલ ઓડિયો સપોર્ટ કરે છે જેથી કરીને તમે ઉંચી ગુણવત્તાની ડિજિટલ ફિલ્મ અને ઓડિઓ તમારા કોમ્પ્યુટર પર વગાડી શકો. તમે ટીવી પણ જોઈ શકો.

4.3.1 માઉસ (Mouse) નો ઉપયોગ

વિન્ડોઝ - 98, માઉસની મદદથી વાપરવા માટે બનાવાવમાં આવ્યું છે. તમે કી - બોર્ડ પણ વિન્ડોઝ-98 સાથે વાપરી શકો છો પરંતુ આમાના ઘણા બધા કાર્યો માઉસની મદદથી સરળતાથી થાય છે. આખા કોર્સ દરમ્યાન તમે માઉસના ચાર પાયાના કાર્યો વાપરશો.

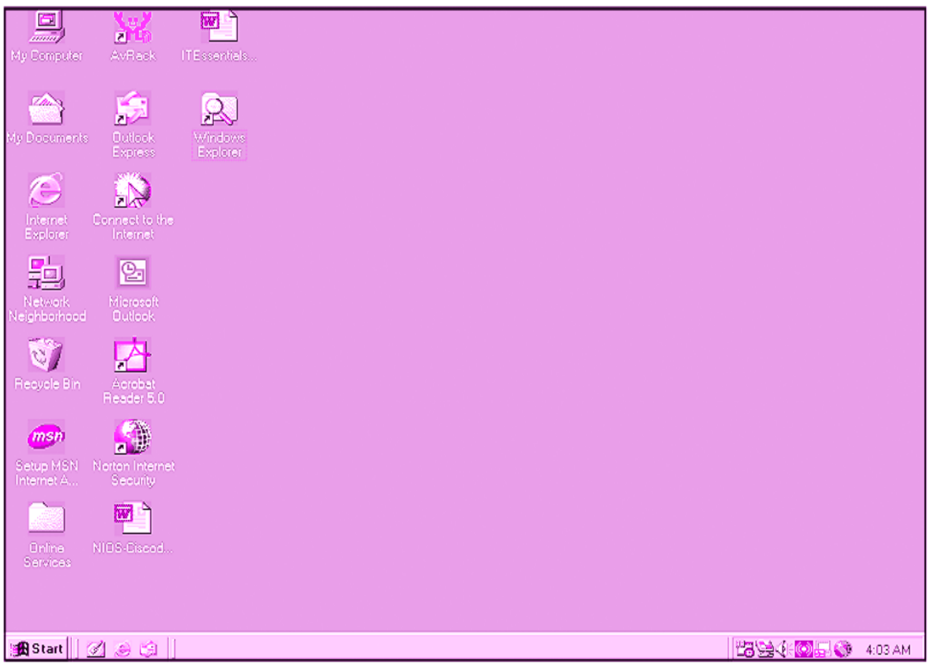
1. **પોઈન્ટિંગ (Pointing)** :- આઈટમ ઉપર માઉસ ફેરવવાને પોઈન્ટિંગ કહે છે.
2. **ક્લિકીંગ (Clicking)** :- સ્ક્રીન ઉપરની આઈટમને પોઈન્ટ કરીને, ઝડપથી માઉસની ડાબી બાજુનું બટન દબાવવાને ક્લીકીંગ કહે છે. સ્ક્રીન ઉપર ક્લિક કરીને તમે આઈટમ પસંદ કરી શકો છો. પ્રસંગોપાત તમે જમણી બાજુનું બટન દબાવીને કાર્ય કરો છો પણ તમને સૂચના ના આપવામાં આવે ત્યાં સુધી ડાબી બાજુનું માઉસ બટન દબાવવું.
3. **ડબલ-ક્લિકીંગ (Double-Clicking)** :- આઈટમને પોઈન્ટ કરીને ઝડપથી બે વાર માઉસ બટન દબાવવાને ડબલ ક્લિકીંગ કહે છે. જ્યારે તમે કયો કમાન્ડ વાપરવો તે બાબતે ચોક્કસ ના હોય ત્યારે, જે આઈટમને અસર કરવી હોય તેના પર ક્લિક કરી પ્રયત્ન કરી જોવો જો કોઈ પરિણામ ના મળે તો ડબલ ક્લિક નો પ્રયત્ન કરો. આ ઘણી વાર સંવાદ બોક્સ બતાવે છે જેમાં તમે ફેરફાર કરી શકો છો.
4. **ડ્રેગીંગ (Dragging)** :- આઈટમને પોઈન્ટ કરીને બટનને પકડી રાખીને માઉસને ફેરવવાની પ્રક્રિયાને ડ્રેગિંગ કહે છે. તમે આ ક્રિયા ડેટા પસંદ કરવા અને લખાણ અને વસ્તુઓ નકલ કરવા માટે વાપરી શકો.

4.4 મેનુ કમાન્ડસ માટેના પ્રતીકો

આખા કોર્સ દરમ્યાન અમે ખાસ સંમેલનનો ઉપયોગ મેનુ કમાન્ડસને દર્શાવવા માટે કર્યો છે. જ્યારે તમારે મેનુ બારમાંથી કમાન્ડ પસંદ કરવો હોય ત્યારે તે નીચેની પેટર્ન અનુસરે છે. દા. ત. Choose File → Open એ મેનુબારમાંથી ફાઈલ પસંદ કરીને ઓપન કમાન્ડ ફાઈલ મેનુમાંથી પસંદ કરવાનો શોર્ટ કટ છે. ઘણી વાર તમે ત્રણ કે તેથી વધુ મેનુ પસંદગીનો ક્રમ જોશો.

4.4.1 ડેસ્કટોપ (Desktop)

જ્યારે તમે તમારૂ કોમ્પ્યુટર ચાલુ કરો છો ત્યારે સ્ક્રીન ઉપર પહેલાં મૂળભૂત ઈનપુટ-આઉટપુટ દર્શાવે છે. થોડી વાર પછી વિન્ડોઝ નિયંત્રણ લે છે અને વાદળછાયુ આકાશ અને માઈક્રોસોફ્ટ વિન્ડોઝ - 98 સ્ક્રીન પર જોવા મળશે. પહેલી વસ્તુ જે હંમેશા દેખાય છે વિન્ડોઝના લોડ થયા પછી તે પાસવર્ડ ટાઈપ કરવા માટે પૂછે છે. (તે એ જ પાસવર્ડ છે જે તમે પહેલી વખત વિન્ડોઝ લોડ થઈ ત્યારે ટાઈપ કર્યો હતો.) ડેસ્કટોપ એ એવી જગ્યા છે જ્યાં તમે તમને ઉપયોગી વસ્તુઓ જે હંમેશા તમને સુલભ હોય તે મૂકી શકો છો. ઈન્સ્ટોલેશન વખતે તમે કયા ઓપ્શન પસંદ કરો છો તે પ્રમાણે એક કે વધારે વસ્તુઓ જે 4.1 આકૃતિમાં બતાવી છે તે જોઈ શકો છો.



આકૃતિ. 4.1 વિન્ડોઝનુ ડેસ્કટોપ

4.4.2 ડેસ્કટોપ ચિહ્નો (Desktop Icons)

1. **My Computer :** My Computer એ સૌથી મહત્ત્વનું ડેસ્કટોપ ચિહ્ન છે (આકૃતિ 4.1 જોવો) તે એ છે જ્યાં બધુ હાજર હોય છે, તમારી હાર્ડડ્રાઈવ, ફ્લોપી સીડી રોમ, કંટ્રોલ પેનલ. કોમ્પ્યુટરના બધા સ્રોતો My Computer માંથી ઉપયોગ કરી શકાય છે. તે તમારા કોમ્પ્યુટરનો રસ્તો છે. તે વાપરવા માટે તેના ઉપર ડબલ ક્લિક કરો.

   My Computer માં સમાવિષ્ટ છે.

(i) હાર્ડ ડિસ્ક ડ્રાઈવ (C : D :)

(ii) 3.5 ફ્લોપી ડ્રાઈવ (A:)

સીડી રોમ ડ્રાઈવ (E:)

(iii) પ્રિન્ટર

(iv) કંટ્રોલ પેનલ

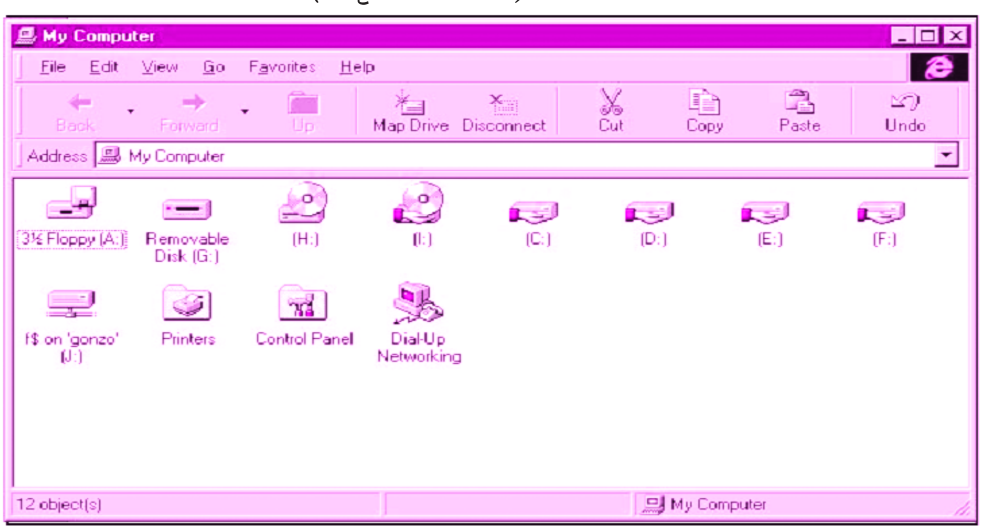
(v) ડાયલ અપ નેટવર્કિંગ (ઈન્ટરનેટ કનેક્શન માટે ખાસ)

(vi) અનુસૂચિત કાર્યો (જ્યારે તમે બહાર હોવ ત્યારે તમારા કોમ્પ્યુટરને કાર્ય આપવા)

My Computer વિન્ડો આકૃતિ 4.2 માં દર્શાવ્યા પ્રમાણે દેખાય છે તમે-થી અને મીનીમાઈઝ અને થી મેક્સિમાઈઝ અને x થી ક્લોઝ કરી શકાય છે.

My Computer માં કશું પણ ડીલીટ, કોપી, કોઈનું નામ બદલવા કરવા માટે તમે પ્રતિબંધિત છો, delete કી દબાવો તો કશુંક પણ નહીં થાય. My Computer માં આવેલ વસ્તુઓ એટલી મહત્ત્વની છે કે તેના વગર તમે તમારા કોમ્પ્યુટર ના વાપરી શકો.દા. ત. તમે હાર્ડ ડિસ્ક કાઢી નાંખો તો તમે ફાઈલો ક્યાં સંગ્રહ કરશો ? My Computer ની વસ્તુઓ કોમ્પ્યુટર સિસ્ટમને લગતાં છે તેથી તે અકબંધ રહેવા જોઈએ.

ધારો કે તમારે તમારી હાર્ડ ડિસ્કને જોવા My Computer નો ઉપયોગ કરવો છે. ડેસ્કટોપ ઉપરથી My Computer ચિહ્ન પર ડબલ ક્લિક કરો તો My Computer વિન્ડો માં અલગ અલગ ચિહ્નો દેખાશે. (આકૃતિ 4.2 જોવો)



આકૃતિ. 4.2 My Computer

હાર્ડ ડિસ્કને દર્શાવતા ચિહ્ન ઉપર ડબલ ક્લિક કરો. હાર્ડ ડિસ્ક વિન્ડો દેખાશે અને તમે હાર્ડ ડિસ્કની માહિતી જોઈ શકશો.

તેવી જ રીતે My Computer માં તમે બીજા ચિહ્નોને ડબલ ક્લિક કરી શકો.

ફ્લોપી ડિસ્ક ની માહિતી જોવા, જે સામાન્ય રીતે drive A તરીકે નિયુક્ત છે.

હાર્ડ ડિસ્કની માહિતી જોવા, જે સામાન્ય રીતે drive C : અને drive D તરીકે નિયુક્ત છે.

ROM માંની કોમ્પકટ ડિસ્કની માહિતી જોવા, જે E : તરીકે નિયુક્ત છે.

પ્રિન્ટરને સુયોજીત (Set) કરવા અને હાજર પ્રિન્ટરો ને તેના કાર્યનું સ્ટેટસ જાણવા.

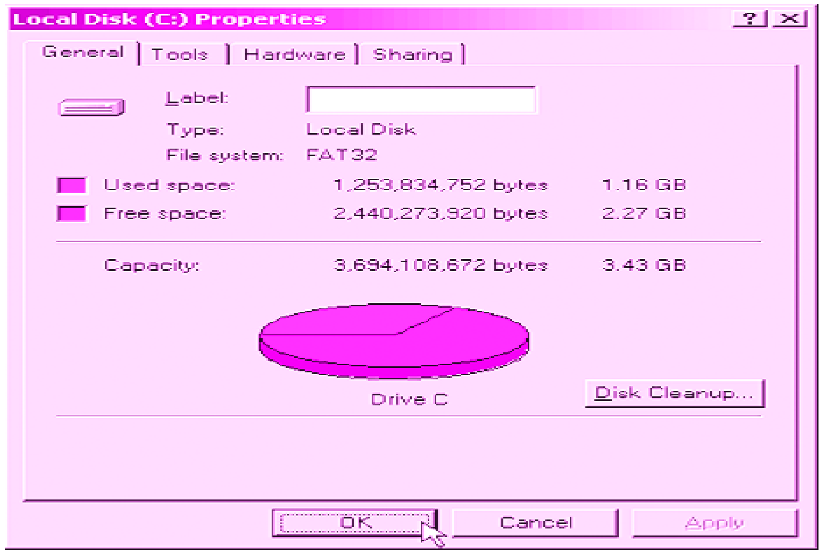
કોમ્પ્યુટરના સેટિંગ બદલવા જે સાધનો વાપરી શકો તે જોવા કોમ્પ્યુટર મેઈન્ટેનશના કાર્યો ગોઠવવા અને જોવા માટે.

વેબ ફોલ્ડરની માહિતી જોવા.

મોડમ વાપરી ને બીજા કોમ્પ્યુટર ઉપર વહેંચાયેલી માહિતી જોવા

My Computer ની ગુણધર્મ વિન્ડો (The Properties Window in My Computer)

દરેક વસ્તુ વિશે વધુ જાણવા એક ચિહ્ન પસંદ કરો, જમણી ક્લિક કરી અને મેનુ માંથી પ્રોપટી (ગુણધર્મ) પસંદ કરો. આ ગુણધર્મ વિન્ડો ચાર ટેબ માં વિભાજીત છે. સામાન્ય (General), સાધનો (Tools), વહેંચાયેલા (Sharing), કોમ્પ્રેસન (Compression). જો તમે ફ્લોપી નું ચિહ્ન પસંદ કરો, જનરલ ટેબ ફ્લોપી વિશે માહિતી આપે છે જે તમને કહે છે કે તમારી ડિસ્ક માં કેટલી જગ્યા બાકી છે અને કુલ કેટલી જગ્યા KB કે MB માં દર્શાવે છે. જો તમે 1.44 MB નો ઉપયોગ કરો તો કુલ 1,45,664 KB જગ્યા હશે. વિન્ડો માં સૌથી ઉપર ફ્લોપી ટેબલ હોય છે.Pie Chart જે મોટા ભાગે ખાલી હોય છે, તે દર્શાવે છે વપરાયેલ અને વણ વપરાયેલ જગ્યા (આકૃતિ 4.3 જુઓ).



આકૃતિ 4.3

આકૃતિ 4.3 માં Tools Tab *નીચેની ત્રણ વિન્ડોઝની ઉપયોગિતા દર્શાવે છે.*

ભૂલ-ચકાસણી સ્થિતિ (Error-Checking Status) *:- આ એવી ઉપયોગિતા છે જે તમને કોમ્પ્યુટરની ભૂલ ચકાસવામાં કરવામાં મદદ કરે છે. આકસ્મિક જો તમે તમારૂ કોમ્પ્યુટર, શટ-ડાઉન ક્લિક કર્યા સિવાય બંધ કરો તો, જ્યારે તમે કોમ્પ્યુટર ફરી વાર ચાલુ કરો ત્યારે હાર્ડ ડ્રાઈવની ચકાસણી કરો.*

બેકઅપ સ્થિતિ (Backup Status) *:- આ ઉપયોગિતા તમને તમારા મહત્ત્વની માહિતીનો બેક અપ લેવામાં મદદ કરે છે. જેથી તમારૂ કોમ્પ્યુટર બગડે તો પણ તમે ક્યારેય કશું ના ગુમાવો. કાયમ તમારે તમારા કામની બે નકલો રાખવી જોઈએ.*

ડિફ્રેગમેન્ટેશન સ્થિતિ (Defragmentation Status) *:- તમારા કોમ્પ્યુટરને ઝડપી બનાવવા આ ઉપયોગિતાનો ઉપયોગ કરો. જેટલું કોમ્પ્યુટર તમે વધારે વાપરો તેટલું તેની ઝડપ ઘટે.* Defragmentation *એને* refresh *કરે. શેરિંગ ટેબ તમને જણાવે છે કે તમારી હાર્ડડિસ્ક બીજા નેટવર્કના વપરાશકર્તા સાથે વહેંચાયેલ છે કે નહીં.*

કમ્પ્રેશન ટેબ લગભગ ક્યારેય વપરાતી નથી. ધારો કે એક દિવસ તમારી પાસે ડિસ્ક સ્પેસ ખૂટી જાય છે. ત્યારે તમે કમ્પ્રેશન વાપરીને ફાઈલ નાની કરીને જગ્યા ઉત્પન્ન કરી શકો.

My Computer *સાથે બીજા પણ થોડા ચિહ્નો ડેસ્કટોપ પર જોવા મળે છે. જે નીચે મુજબ છે.*

2. **My Documents** *:અહીં તમે તમારી વ્યક્તિગત સામગ્રી જે કોમ્પ્યુટરથી બનાવો તે સંગ્રહ કરો છો. તમે તમારી વસ્તુઓ બીજે પણ મૂકી શકો છો.* My Documents *ને ડેસ્કટોપ ડ્રોઅર તરીકે વિચારો કે જ્યાં તમે તમારા પેપરો મૂકી શકો.*

3. **Recycle Bin :-** *ડેસ્કટોપ અને ફોલ્ડર્સ માંથી કાઢી નાંખેલ ફાઈલો માટે જગ્યા પૂરી પાડે છે અને કાઢી નાંખેલ ફાઈલો પુનઃપ્રાપ્ત કરવા માટે બીજી તક પૂરી પાડે છે.*

4. **Internet Explorer :-** *આ સોફ્ટવેર છે જે બ્રાઉસર તરીકે ઓળખાય છે. જે ઈન્ટરનેટ*

વાપરવા માટે ઉપયોગી છે. Internet Explorer એવું બીજું ચિહ્ન હોઈ શકે જે તમને ઈન્ટરનેટ કનેકશન તમારૂ કોમ્પ્યુટર કોન્ફીગર કરવામાં મદદ કરશે.

5. **Network Neighbourhood *(નેટવર્ક પાડોશી)* :-** તમારા અને બીજા કોમ્પ્યુટરો જે LAN અને WAN માં જોડાયેલ છે તેના વહેંચાયેલ સ્ત્રોતોનો ઉપયોગ કરવામાં મદદ કરે છે. જો તમારૂ કોમ્પ્યુટર નેટવર્ક સાથે જોડાયેલ હોય તો તમે નેટવર્ક નેબરહૂડ નો ઉપયોગ નેટવર્ક સ્ત્રોતોનો ઉપયોગ કરવા કરી શકો જેમ તમે સ્થાનિક સ્ત્રોતો ને વાપરો છો.

   નેટવર્ક સ્ત્રોતો ને ખોલવા નીચેના પગલાં અનુસરો.

   (i) ડેસ્કટોપ ઉપર હાજર નેટવર્ક નેબરહૂડ ચિહ્ન ડબલ ક્લિક કરો. તમારા વર્ગગ્રુપના કોમ્પ્યુટરો પ્રદર્શિત થશે. જો તમારે તમારા નેટવર્કમાં હાજર બધા સ્ત્રોતો જોવા હોય તો Entire Network પર ડબલ ક્લિક કરો.

   (ii) તમને જોઈતાં સર્વર અને ફોલ્ડર પણ ડબલ ક્લિક કરો.

6. **Online સેવાઓ :-** ઓનલાઈન સેવા એ ઉમેદવારી કોમ્પ્યુટર સેવા છે તમે તેનો ઉપયોગ અલગ-અલગ પ્રકારની માહિતી વાપરવા અને સંદેશાઓની આપ-લે માટે કરી શકો. ઓનલાઈન સેવાથી તમે ન્યૂઝ, રમત ગમત, હવામાન શેરબજાર અને ઘણી બધી વસ્તુઓની માહિતી મેળવી શકો. તમને જુદી જુદી લાયબ્રેરી અને ડેટાબેઝ ને વાપરી શકો. તમે ઓનલાઈન સેવા ના બીજા વપરાશકર્તાને સંદેશો મોકલી શકો. ઓનલાઈન સેવાના ઉદાહરણો "અમેરિકન ઓનલાઈન", "Compuserve", "Prodigs" અને MSN છે.

7. **તારીખ, સમય, વોલ્યુમ :** સ્ક્રીનની નીચેની જમણી બાજુ થી તમે કહી શકો અત્યારે શું સમય છે. ઘડિયાળના ચિહ્ન ઉપર ક્લિક કરીને તમને તારીખ મળી શકે. તમે સ્પીકર ચિહ્ન પર ક્લિક કરીને વોલ્યુમ પણ કંટ્રોલ કરી શકો.

8. **સ્ટાર્ટ બટન :-** આ બટન સ્ક્રિનની ડાબી બાજુએ નીચેની સાઈડે હોય છે. આ અલગ અલગ કાર્યક્રમ અને એપ્લીકેશન ને ચાલુ કરવાનો સરળ રસ્તો આપે છે.

9. **ટાસ્કબાર :-** આ સ્ક્રીનની નીચેની બાજુએ હોય છે. આ બધી ખૂલ્લી એપ્લીકેશન અને વિન્ડો દર્શાવે છે. સ્ટાર્ટ બટન અને ટાસ્કબાર સાથે કામ કરે છે અને ખૂલ્લી એપ્લિકેશન ને સરળતાથી બંધ કરવામાં મદદ કરે છે.

તમે ડેસ્કટોપ ચિહ્ન વિશે વધુ આગળના થોડા વિભાગમાં શીખશો.

પાઠગત પ્રશ્નો 4.1

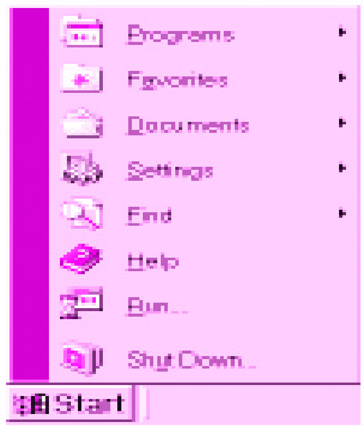
---

1. વિન્ડોઝ -98 ના કોઈ પાંચ મુખ્ય લક્ષણો લખો.
2. માઉસની એપ્લિકેશનમાં વપરાતી નીચેની વસ્તુઓ સમજાવો.
   (અ) પોઈન્ટિંગ (બ) ક્લિકિંગ (ક) ડબલ-ક્લિકિંગ (ડ) ડ્રેગીંગ
3. ડેસ્કટોપ શું છે ? કઈ મુખ્ય વસ્તુઓ તમે ડેસ્કટોપ પર જોવો છો ?

4. Computer ના મુખ્ય ભાગો કયાં છે ?

4.5 સ્ટાર્ટ બટન અને ટાસ્ક બાર

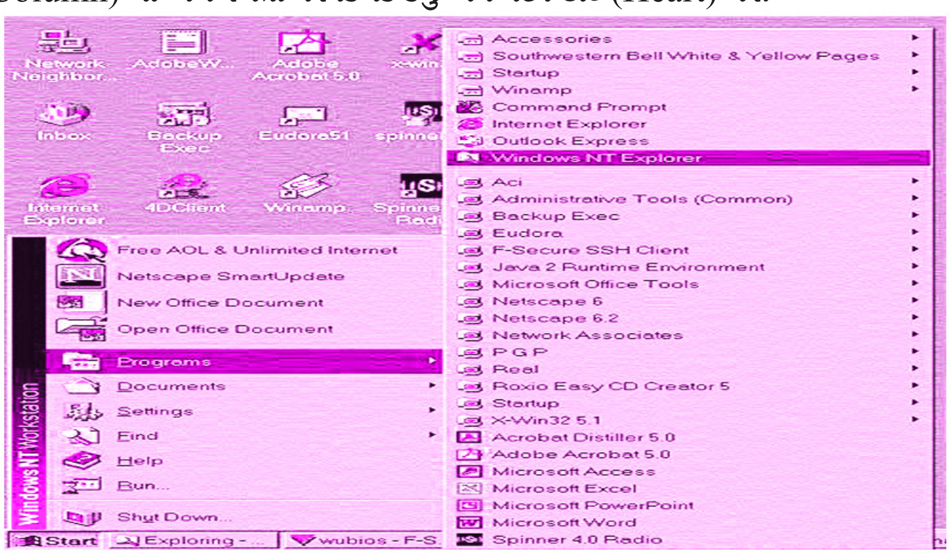
નીચે ડાબે ખૂણે Start *બટન છે. તમે સ્ટાર્ટ બટન ઉપર માઉસ મૂકો અને તમે જોશો "શરૂ કરવા અહીં ક્લિક કરો." સ્ટાર્ટ તમારા કોમ્પ્યુટરના બધા કાયોના વપરાશ માટે સરળ રસ્તો આપે છે. દરેક વખતે તમે કોમ્પ્યુટરમાં કાર્યક્રમ નાંખો છો ત્યારે તેને સ્ટાર્ટ મેનુમાં જોઈ શકશો. આકૃતિ.* 4.4 *જયારે તમે સ્ટાર્ટ પર ક્લિક કરો ત્યારે મુખ્ય ઉપલબ્ધ વસ્તુઓ બતાવે છે.*



આકૃતિ. 4.4

4.5.1 કાર્યક્રમોનું સબમેનુ

તમારા કોમ્પ્યુટર પરના દરેક કાર્યક્રમો, પ્રોગ્રામ ઉપમેનુમાંથી ઉપયોગ કરી જોઈ શકાય છે, તમે કોમ્પ્યુટ ર એસેસરીઝ જેવા કે સિસ્ટમ સાધનો, સંદેશા અને મનોરંજનના લક્ષણો પણ વાપરી શકો છો. તમારા માઉસ નું પોઈન્ટર પ્રોગ્રામ પર ફેરવો તેથી બીજું મેનુ પ્રદર્શિત થશે (Pop Up). *આકૃતિ* 4.5 *આપણે પ્રોગ્રામ પર અને પછી એસેસરીઝ ઉપર ક્લિક કર્યુ. આગામી સ્તંભ* (Column) *માં અમે રમતો પર ક્લિક કર્યુ અને પછી હાર્ટ* (Heart) *પર.*



આકૃતિ 4.5 પ્રોગ્રામનું સબમેનુ

જ્યારે તમે કાર્યક્રમો અને ફાઈલો, બીજા કાર્યક્રમ પર ખોલો છો ત્યારે Window 98 *તમારી બધી ખૂલ્લી વિન્ડોઝની ટાસ્કબારમાં યાદી બનાવે છે. ટાસ્કબારનો ફાયદો એ છે કે એક જ સમયે એક કરતાં વધારે કાર્યક્રમો ખૂલ્લા હોય ત્યારે તેની યાદી જોઈ શકો છો. તે બીજા કાર્યક્રમથી છૂપાયેલ હોય કે નહીં. કાર્યક્રમની વિન્ડોઝ ડેસ્કટોપ પર મીનીમાઈઝ કરેલ હોય તો તેને એપ્લિકેશન વિન્ડોઝ ઉપર ક્લિક કરીને વાપરી શકો છો. એપ્લિકેશન વિન્ડો ઉપર ક્લિક કરવાથી તમે તે કાર્યક્રમને બીજી વિન્ડોની આગળ લાવી શકો છો. અને જો મીનીમાઈઝ હોય તો તેને રીસ્ટોર કરી શકો છો. જો તમે વધારે વિન્ડો ખોલશો તો વધારે વસ્તુની યાદી ટાસ્કબાર ઊપર બનશે. તમે ટાસ્કબારની ઉંચાઈ વધારી શકો છો. ઉંચાઈ વધારવા માટે માઉસ પોઈન્ટર ઉપરની બોર્ડર ઉપર મૂકતાં તે (બે માથાવાળું શરીર)માં બદલાઈ જશે ત્યાર બાદ ક્લિક કરો. ડાબી બાજુનું માઉસ બટન દબાવી અને ઉપર ડ્રેગ કરો.*

4.5.2 મનપસંદ સબમેનુ (Favourite Sub Menu)

જ્યારે તમે ઈન્ટરનેટ વાપરતા હોય ત્યારે તમારા મનગમતા એડ્રેસ તમે આ ફોલ્ડરમાં સ્ટોર કરી શકો છો જેથી તમારે યાદ રાખવા ન પડે.આ લક્ષણ ઈન્ટરનેટ વપરાશકર્તા માટે બહુ લાભદાયી છે. તમે ડોક્યુમેન્ટ્સ, ફોલ્ડર અને બીજા નેટવર્ક પણ ઝડપી મેળવી શકો છો.

4.5.3 દસ્તાવેજોનું સબમેનુ (Documents Sub Menu)

વિન્ડોઝ તાજેતરમાં વાપરેલ દસ્તાવેજો પર નજર રાખે છે. સ્ટાર્ટ મેનુનો ડોક્યુમેન્ટ વિકલ્પ કોઈ પણ એપ્લીકેશનમાં ખોલેલ 15 *ડોક્યુમેન્ટ ઉપલબ્ધ બનાવે છે. આકૃતિ* 4.4 *માં ડોક્યુમેન્ટ વિકલ્પ ઉપર ક્લિક કરો અને તાજેતરમાં તમે જે ડોક્યુમેન્ટ પર કાર્ય કરો છો તેની યાદી બતાવશે. ડોક્યુમેન્ટની આલ્ફાબેટેકલી (મૂળાક્ષરો) પ્રમાણે જોવા મળશે, નહિ કે તેના વપરાશ પ્રમાણે અને તેનો યાદીમાં સમાવેશ થશે એ તમે ફોલ્ડરમાંથી ખોલશો કે એપ્લિકેશનમાંથી ખોલશો. આ ડોક્યુમેન્ટસ તમને ઉપયોગી ના લાગે તો તમે યાદીને ખાલી પણ કરી શકો છો. ટાસ્કબાર પર જમણી ક્લિક કરો નાની વિન્ડોમાં તમને પ્રોપર્ટી (ગુણધર્મો) મળશે ત્યાં ક્લિક કરવાથી બીજી વિન્ડો દેખાશે. સ્ટાર્ટ મેનુ પ્રોગ્રામ પર ક્લિક કરો ત્યાં ડોક્યુમેન્ટ મેનુમાં ક્લિઅર બટન પર ક્લિક કરો.*

4.5.4 સેટિંગ્સ

અહીથી તમે કંટ્રોલ પેનલ, પ્રિન્ટર, ટાસ્કબાર અને સ્ટાર્ટ મેનુ (જોવો આકૃતિ 4.5*) ફોલ્ડર વિકલ્પો, સક્રિય ડેસ્કટોપ અને વિન્ડોનો અપડેટ વગેરે ઉપયોગ કરી શકો છો. તમે પ્રોગ્રામ ઉમેરી શકો, કાઢી શકો, મોનીટરના સેટિંગ બદલી શકો, નેટવર્ક સાથે જોડાઈ શકો સ્ટાર્ટ ડીસ્ક બનાવી શકો, ડિસ્પ્લેના ગુણધર્મોને કસ્ટમાઈઝ કરી શકો અને ડાયલ અપ નેટવર્કિંગ સેટ કરી શકો છો.*

કંટ્રોલ પેનલના મુખ્ય લક્ષણો

1. *તમે તમારા વિન્ડોઝના વાતાવરણને અનુક્રમિત કરી શકો છો. તમે ચિત્રો, પેટર્ન અને સ્કેન ફોટોગ્રાફ તમારા કોમ્પ્યુટર ના વોલ પેપર અને પૃષ્ઠભૂમિમાં ગોઠવી શકો છો. ડિસપ્લે પ્રોપર્ટીની અલગ અલગ ટેબ વાપરીને ડેસ્કટોપ પરના આઈકોન, વિન્ડોનો કલર અને વસ્તુઓની સાઈઝ બદલી શકો છો. તમે ચાલુ ડેસ્કટોપમાં વસ્તુઓ ઉમેરી શકો છો અને સ્ક્રીન સેવર સુયોજિત કરી શકો છો.*

2. તમે કંટ્રોલ પેનલમાં આવેલ એડ - રીમૂવ ઓપ્શનનો ઉપયોગ કરીને ઝડપથી કાર્યક્રમ સ્થાપિત કરી શકો છો જેવા કે માઈક્રોસોફ્ટ વર્ડ અને ગેમ્સ.

3. કંટ્રોલ પેનલમાં આવેલ નેટવર્ક ઓપ્શનનો ઉપયોગ કરીને તમારા કોમ્પ્યુટરને નેટવર્ક સાથે જોડી શકો છો. તમારા કોમ્પ્યુટર ને નેટવર્ક સાથે જોડવા નીચેના પગલાં ભરો.

- Start ક્લિક કરો, Setting → Control Panel ને પસંદ કરો અને ડબલ ક્લિક કરો જેથી નેટવર્ક ડાયલોગ બોક્સ દેખાશે. Add ઉપર ક્લિક કરી નેટવર્ક પસંદ કરો. કમ્પોનન્ટ (Component) પ્રકારનું બોક્સ દેખાશે. ક્લાયન્ટ Add ઉપર ક્લિક કર્યા પછી ઉપર ક્લિક કરો. ક્લાયન્ટ સોફ્ટવેરની યાદી પ્રદર્શિત થશે.
- ઉત્પાદનકર્તાની યાદીમાં તમારા નેટવર્ક સોફ્ટવેરના ઉત્પાનદકર્તાના નામ પર ક્લિક કરો.
- નેટવર્ક ક્લાયન્ટની યાદીમાં તમે વાપરતા ક્લાયન્ટ સોફ્ટવેર ની પસંદગી કરો અને પછી OK પર ક્લિક કરો. હવે ક્લાયન્ટ સોફ્ટવેર તમારા કોમ્પ્યુટરમાં ઉમેરાઈ ગયું.
- કન્ફ્યુગીરેશન (રૂપરેખાંકન) ટેબ પરથી તમારો ક્લાયન્ટ પસંદ કરો અને પ્રોપર્ટી (ગુણધર્મ) પર ક્લિક કરો. તમારા નેટવર્કના કન્ફ્યુગીરેશન (રૂપરેખાંકન) નાંખો અને OK પર ક્લિક કરો. (જો તમે તમારા નેટવર્ક માટેના વિકલ્પો ના જાણતાં હોય તો તમારા નેટવર્ક સંચાલકનો સંપર્ક કરો.)
- OK પર ક્લિક કરો અને પછી ફરી વાર OK ક્લિક કરો. ક્લાયન્ટ સોફ્ટવેર સ્થાપિત થશે અને તમારૂ કોમ્પ્યુટર પુનઃ શરૂ થશે.

4. જો તમને સેટ અપમાં સમસ્યા હોય અથવા વિન્ડોઝ-98 ચાલુ કરવામાં તકલીફ હોય તો તમે સ્ટાર્ટ અપ ડિસ્કનો ઉપયોગ કરી શકો છો તમારૂ કોમ્પ્યુટર ચાલુ કરવા માટે અને સેટ અપ ચલાવવા માટે અથવા સિસ્ટમ ફાઈલનો વાપરવા માટે. જો તમને તમારી હાર્ડ ડિસ્ક સાથે પ્રોબ્લેમ હોય. દા. ત. તમે તમારા કોમ્પ્યુટર ને ચાલુ કરવા સ્ટાર્ટ અપ ડિસ્ક વાપરી શકો છે અને હાર્ડ ડિસ્કની તકલીફનું નિવારણ લાવી શકો.

વિન્ડોઝ-98 માંથી સ્ટાર્ટ અપ ડિસ્ક બનાવવા નીચેના પગલાં અનુસરો

- સ્ટાર્ટ પર ક્લિક કરી, Setting પસંદ કરો અને કંટ્રોલ પેનલ પર ક્લિક કરો અને ત્યાર બાદ Add Remove કાર્યક્રમ પર ડબલ ક્લિક કરો. Add Remove પ્રોગ્રામના ગુણધર્મોનો ડાયલોગ બોક્સ પ્રદર્સિત થશે.
- Start Up Disc ટેબ પર ક્લિક કરો અને ત્યારબાદ Create Disc પર ક્લિક કરો.
- Windows 98 સ્ટાર્ટ અપ ડિસ્ક એવું ફ્લોપી ડિસ્કને લેબલ આપો. તેને Floppy Disc Drive માં દાખલ કરો અને OK પર ક્લિક કરો.સ્ટાર્ટ અપ ડિસ્કનો ઉપયોગ કરી. તમારૂ કોમ્પ્યુટર ચાલુ કરવા નીચેના પગલાં અનુસરો.

(i) સ્ટાર્ટ અપ ડિસ્કને ફ્લોપી ડિસ્ક ડ્રાઈવમાં દાખલ કરો.

(ii) તમારૂ કોમ્પ્યુટર પુનઃ ચાલુ કરો. Window 98 નું સ્ટાર્ટ અપ મેનુ પ્રદર્શિત થશે. યોગ્ય નંબર, CD ROM માટે ટાઈપ કરો અને એન્ટર દબાવો. સ્ક્રીન પરની સૂચનાઓને અનુસરો. શ્રેણીબધ્ધ સ્કેન પછી MS DOS પ્રોમ્પટ દેખાશે. આ પ્રોમ્પટ થી તમે સિસ્ટમ ફાઈલને સ્ટાર્ટ અપ ડિસ્કનો ઉપયોગ કરી શકો છો.

5. તમે ઝડપથી પ્રિન્ટને સેટ કરવા, મૂળભૂત પ્રિન્ટર પસંદ કરવા અને પ્રિન્ટરના સેટીંગ્સ બદલવા પ્રિન્ટર વિઝાર્ડને ઉમેરી શકો છો. વિઝાર્ડ તમને પ્રિન્ટર સેટ અપ માટે દરેક પગલાંની માહિતી આપશે.

4.5.5 શોધ (FIND)

જ્યારે તમે ચોક્કસ ફોલ્ડર કે ફાઈલ શોધતાં હોય ત્યારે તમે Find કમાન્ડ વાપરી શકો છો, ઘણા બધા ફોલ્ડર ખોલવાના બદલામાં Find કમાન્ડ ચોક્કસ ડ્રાઈવ કે સમગ્ર કોમ્પ્યુટર કે નેટવર્ક ઉપરના કોમ્પ્યુટરને શોધવામાં મદદ કરે છે.

4.5.6 મદદ (Help)

જ્યારે તમે અટવાઈ જાવ ત્યારે વિન્ડોઝ તમને સહાય કરી શકે છે. ત્યારે START - HELP અથવા F1 કી દબાવો.

4.5.7 ચલાવવું (RUN)

પ્રોગ્રામ ચાલુ કરવા માટે, કાર્યક્રમ તમારા કોમ્પ્યુટરમાં સ્થાપિત (Install) કરવા માટે, ફોલ્ડર ખોલવા માટે કે નેટવર્ક સાથે જોડાવા માટેનો સૌથી ઝડપી રસ્તો છે. રનથી કોઈ પણ વસ્તુનો તમને તાત્કાલિક ઉપયોગ કરી શકો છો. પરંતુ એ વધારે અનુભવી વ્યક્તિ માટે નો ઓપ્શન છે. Run ને પસંદ કરવાથી તે વિન્ડો લઈ આવશે જ્યાં તમે ફાઈલ કે ફોલ્ડરનું નામ ટાઈપ કરી શકો છો જે તમે ખોલવા માંગો છો. જો તમને ફાઈલનો આખો રસ્તો ખબર ન હોય તો, તમારે જે જોઈએ છે તે મેળવવા માટે તમે બ્રાઉઝ બટન ઉપર ક્લિક કરી શકો. તેમ છતાં સામાન્ય રીતે ડોક્યુમેન્ટ ખોલવા કે કાર્યક્રમ રન (ચલાવવા) માટેનો સૌથી ઝડપી રસ્તો નથી. ઘણી વાર ક્રમમાં ફોલ્ડરો ખોલ્યાં કરતાં આ વધુ સીધી લાગે છે

4.5.8 શટ ડાઉન (Shut Down)

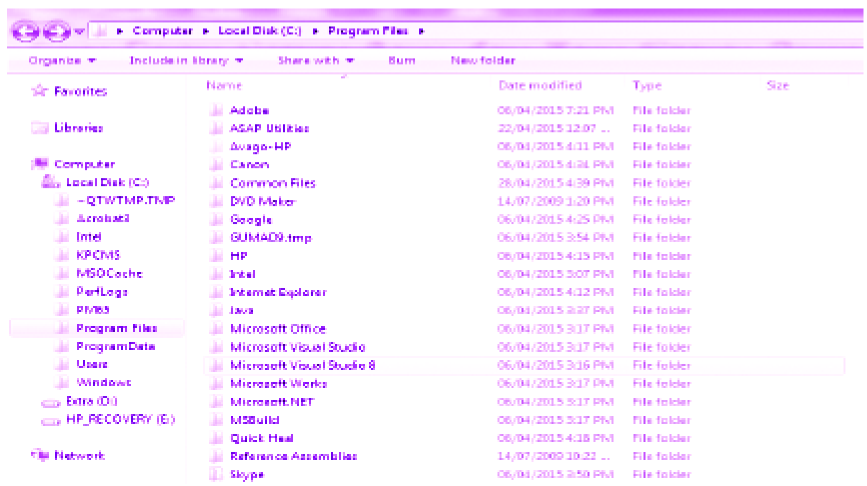
સેશનના અંતે તમારો વિન્ડોઝને શટ ડાઉન કરવું જોઈએ, ક્યારેય પાવર બંધ ના કરશો, બધી ચાલુ એપ્લિકેશન બંધ કરો, સ્ટાર્ટ બટન પર ક્લિક કરો, શટ ડાઉનનો વિકલ્પ પસંદ કરી OK પર ક્લિક કરો. ઘણાં નવા કોમ્પ્યુટરો આપમેળે Power બંધ કરી દે છે જ્યારે તમે શટ - ડાઉન કરો. જો તમારા કોમ્પ્યુટરમાં આ સુવિધા ના હોય તો પાવરને બંધ કરી દો.

4.5.9 લોગ ઓફ (Log Off)

આ લક્ષણ જો તમે તમારૂ કોમ્પ્યુટર બીજા કોઈની સાથે વાપરતાં હોય (Share કરતાં હોય) તો જ જરૂરી છે. જો તમે એકલા જ કોમ્પ્યુટર વાપરતાં હોય તો આ લક્ષણ પસંદ કરવાની જરૂર નથી.

4.6 વિન્ડોઝ એક્સપ્લોરર (Windows Explorer)

વિન્ડોઝ એક્સપ્લોરર એક લક્ષણ છે જેનો ઉપયોગ કરીને તમે તમારા કોમ્પ્યુટરમાં અને નેટવર્ક ડ્રાઈવમાં રહેલી માહિતને જોઈ શકો છો. તમારા ડ્રાઈવ અને ફોલ્ડરને અલગ વિન્ડોઝમાં ખોલવાના બદલે તમે તેને એક જ વિન્ડોઝમાં જોઈ શકો છો. વિન્ડોઝ એક્સપ્લોરરના વિન્ડોઝની ડાબી બાજુએ તમારા ડ્રાઈવ અને ફોલ્ડર્સની યાદી સમાવે છે, અને જમણી બાજુ પસંદ કરેલા ફોલ્ડરની સામગ્રી બતાવે છે. વિન્ડોઝ એક્સપ્લોરર વાપરવા માટે સ્ટાર્ટ બટન પર ક્લિક કરો પછી પ્રોગ્રામ્સ → વિન્ડોઝ એક્સપ્લોરર પર ક્લિક કરો (આકૃતિ 4.6 જુઓ)



આકૃતિ 4.6 વિન્ડોઝ એક્સપ્લોરર

વિન્ડોઝ એક્સપ્લોરર નીચેના લક્ષણો ધરાવે છે.

- *તમે ફાઈલ અથવા ફોલ્ડરમાં રહેલી ઘણી બધી ફાઈલોને એક ફોલ્ડરમાંથી બીજા ફોલ્ડર અથવા એક ડ્રાઈવમાંથી બીજા ડ્રાઈવમાં કોપી કરી શકો છો. આ માટે કર્સરને સોર્સ ફાઈલ અથવા ફોલ્ડર પર મુકી અને ખેંચીને ડેસ્ટીનેશન ફોલ્ડર અથવા ડ્રાઈવ પર મૂકો. તમે આ પ્રક્રિયા* copy *અને* paste *દ્વારા પણ કરી શકો છો.*

- *તમે ફાઈલ અથવા ફોલ્ડરને ડિલીટ (કાઢી) કરી શકો છો, તેના માટે ફાઈલ અથવા ફોલ્ડરને પસંદ કરી, ટુલબાર પરના ડિલીટ* (Delete) *બટન પર ક્લિક કરો. વિન્ડોઝ તેને રિસાયકલ બિનમાં (કચરાપેટી) મોકલશે.*

- *તમે* cut *અને* paste *નો ઉપયોગ કરીને ફાઈલ અથવા ફોલ્ડરને એક ફોલ્ડર અથવા ડ્રાઈવમાંથી બીજા ફોલ્ડર અથવા ડ્રાઈવમાં ખસેડી શકો છો.*

- *જો ભૂલથી તમે કોઈ ફાઈલ અથવા ફોલ્ડર ડિલીટ અથવા ખસેડી અથવા કોપી કરી હોય તો તેને મુળ જગ્યા પર પરત લાવી શકો છો, આના માટે કર્સરને* Undo *ઉપર ખસેડી ક્લિક કરો.*

- *તમે ફાઈલ અથવા તેના પ્રોગ્રામને બે વાર ક્લિક કરીને ખોલી શકો છો.*

4.7 ફાઈલો, ફોલ્ડરો અને વિન્ડોઝની વ્યવસ્થા (Managing Files, Folders and windows)

વિન્ડોઝ વાપરવા માટે તમને બે વસ્તુની ખબર હોવી જોઈએ : ફાઈલો અને ફોલ્ડરો. નીચેના પ્રથમ ત્રણ આઈકન (ચિન્હો) અલગ અલગ ફોલ્ડરો અને પછીના ત્રણ ચિન્હો અલગ અલગ પ્રકારની ફાઈલો બતાવે છે. (આકૃતિ 4.7)

આકૃતિ 4.7 ફાઈલો અને ફોલ્ડરો

ફાઈલો હંમેશા ફોલ્ડરની અંદર હોય છે. winows 98 માં તમે તમારા કાગળો અને પ્રોગ્રામો તમારી અનુકૂળતા પ્રમાણે ગોઠવી શકો છો. તમે ફાઈલોને ફોલ્ડરમાં રાખી શકો છો, અને તમે તેને ખસેડી, કોપી, નામ બદલવું અને ફાઈલ અને ફોલ્ડરોને શોધી શકો છો.

windows 98 માં ફાઈલનું નામ 255 અક્ષરોનું હોઈ શકે છે, જેમાં સ્પેશ (જગ્યા) પણ ગણેલી છે. જો કે ફાઈલના નામમાં નીચેના અક્ષરો વાપરી શકાતા નથી. \ / < > /.

4.7.1 ફોલ્ડર બનાવવું (creating Floders)

તમે જયારે કોઈ પ્રોગ્રામનો ઉપયોગ કરો અને તમારૂં કામ સેવ કરો, અથવા તમે પ્રોગ્રામને **ઈન્સ્ટોલ (install)** કરો ત્યારે તમે ફાઈલો બનાવો છો. તમે ફાઈલોને ઘણી જગ્યાએ રાખી શકો છો- જેમ કે હાર્ડ ડિસ્ક, નેટવર્ક ડ્રાઈવ, ફલોપી ડિસ્ક વગેરે તમે ફાઈલોને સારી રીતે ગોઠવવા માટે તેને ફોલ્ડરોમાં રાખી શકો છો.

ફોલ્ડર બનાવવા માટે નીચેના પગલાં અનુસરો :

1. ડેસ્કટોપ ઉપર **My Computer** ઉપર બે વાર કિલક કરો. **My Computer** ની વિન્ડો ખુલશે. **Programms** પર - **windows Explorerer** પસંદ કરો.
2. જેમાં તમારે ફોલ્ડર બનાવવું છે તે ડિસ્ક ડ્રાઈવ અથવા ફોલ્ડર પર બે વાર કિલક કરો. ડ્રાઈવ અથવા ફોલ્ડર વિન્ડો ખુલશે.
3. **File** મેનુમાં, **New** પર જાઓ અને **Folder** પર કિલક કરો. ફોલ્ડરનું નામ લખો અને Enter દબાવો. તમે પસંદ કરેલી જગ્યા પર નવું ફોલ્ડર દેખાશે.

ફાઈલ અથવા ફોલ્ડરને શોધવા નીચેના પગલાં અનુસરો :

1. **Start** બટન પર કિલક કરો, **Find** પર જાઓ, અને **Files or Folders** પર કિલક કરો. **Find** ડાયલોગ બોક્ષ દેખાશે.
2. હવે તમારે જે ફાઈલ અથવા ફોલ્ડર શોધવું હોય તેનું નામ લખો. Look in down તીર પર કિલક કરો અથવા કયાં શોધવું છે તે જણાવવા **Browse** પર કિલક કરો. હવે **Find** કિલક કરો. થોડીકવાર પછી સર્ચ (શોધ) નું પરિણામ જોવા મળશે.

ફાઈલ અથવા ફોલ્ડરને ખોલવા માટે નીચેના પગલાં અનુસરો :

1. ડેસ્કટોપ ઉપર **My Computer** પર બે વાર ક્લિક કરો. **My Computer** વિન્ડો ખુલશે અથવા તમે **start** પર ક્લિક કરી અને **Programs → windows Explorer** પસંદ કરો.
2. તમારે જે ફાઈલ અથવા ફોલ્ડર ખોલવું છે તે જે ડ્રાઈવમાં હોય તેના પર બે વાર ક્લિક કરો.
3. તમારે જે ફાઈલ અથવા ફોલ્ડર ખોલવું છે તેના પર બે વાર ક્લિક કરો.

ફાઈલ અથવા ફોલ્ડરનું નામ બદલવા માટે નીચેના પગલાં અનુસરો :

1. ડેસ્કટોપ ઉપર **My Computer** પર બે વાર ક્લિક કરો. **My Computer** વિન્ડો ખુલશે અથવા તમે **Start** પર ક્લિક કરી અને **Programs →Windows Explorer** પસંદ કરો.
2. તમારે જે ફાઈલ અથવા ફોલ્ડરનું નામ બદલવું છે તે જે ડ્રાઈવમાં હોય તેના પર બે વાર ક્લિક કરો. જેનું નામ બદલવું છે તે ફાઈલ અથવા ફોલ્ડર પસંદ કરો.
3. **File** મેનુમાં **Rename** પર ક્લિક કરો. નામ લખો અને **Enter** દબાવો.

ફાઈલ અથવા ફોલ્ડરને એક જગ્યાએથી બીજી જગ્યાએ કોપી અથવા ખસેડવા માટે નીચેના પગલાં અનુસરો :

1. ડેસ્કટોપ ઉપર **My Computer** પર બે વાર ક્લિક કરો. **My Computer** વિન્ડો ખુલશે અથવા તમે **Start** પર ક્લિક કરી અને **Programs→Windows Explorer** પસંદ કરો.
2. **Edit** મેનુમાં, ફાઈલને કોપી કરવા **copy** પર ક્લિક કરો, અથવા ફાઈલને ખસેડવા **cut** પર ક્લિક કરો. તમારે ફાઈલ અથવા ફોલ્ડર જ્યાં મુકવા છે તે ફોલ્ડર પર બે વાર ક્લિક કરો. **Edit** મેનુમાં **Paste** પર ક્લિક કરો. ફાઈલ તેની નવી જગ્યા પર જોવા મળશે.

ફાઈલ અથવા ફોલ્ડરને કાઢી નાખવા નીચેના પગલાં અનુસરો :

1. ડેસ્કટોપ ઉપર My Computer પર બે વાર ક્લિક કરો. My Computer પર ક્લિક કરી અને Programs → **Windows Explorer** પસંદ કરો.
2. તમારે જે ફાઈલ અથવા ફોલ્ડર કાઢી નાખવા (delete) છે તે જે ડ્રાઈવમાં છે તેના પર બે વાર ક્લિક કરો. જે ફાઈલ અથવા ફોલ્ડર કાઢી નાખવી છે તેને પસંદ કરો.
3. File મેનુમાં Delete પર ક્લિક કરો. 'Confirm File Delete' ડાયલોગ બોક્ષ દેખાશે. yes પર ક્લિક કરો. ફાઈલ ડીલીટ થઈને Recycle Bin માં ખસી જશે.

ફાઈલ અથવા ફોલ્ડરને હંમેશા માટે કાઢી નાખવા નીચેના પગલાં અનુસરો :

1. ડેસ્કટોપ ઉપર **Recycle Bin** પર બે વાર ક્લિક કરો. **Recycle Bin** વિન્ડો

ખુલશે. તમારે જે ફાઈલ અથવા ફોલ્ડર હંમેશા કાઢી નાખવા છે તેને પસંદ કરો.

2. ટુલબારમાં **Delete** પર ક્લિક કરો. **Confirm File Delete** ડાયલોગ બોક્ષ દેખાશે. **Yes** પર ક્લિક કરો. **File** હંમેશા માટે ડીલીટ થઈ જશે. (જો તમારે બધી જ ફાઈલો અને ફોલ્ડરોને **Recyle Bin** માંથી કાઢી નાખવા હોય તો, **File** મેનુમાં **Empty Recyle Bin** પર ક્લિક કરો. **Confirm File Delete** ડાયલોગ બોક્ષ દેખાશે. **Yes** પર ક્લિક કરો. **Recyle Bin** માંથી બધી ફાઈલો ડીલીટ થઈ જશે.)

4.7.2. શોર્ટકટ (Shortcuts)

તમે જે ફાઈલને વારંવાર ઉપયોગ કરતા હોય તેનો સરળતાથી વાપરી શકવા માટે તમે તેનો શોર્ટકટ બનાવી શકો. શોર્ટકટ ફાઈલની જગ્યા બદલતુ નથી- તે ફકત એક પેઈન્ટર છે જેનાથી ફાઈલ ઝડપથી ખોલી શકાય. તમે શોર્ટકટ ડીલીટ કરો તો મૂળ ફાઈલ ડીલીટ થતી નથી ફાઈલના શોર્ટકર્ટ બનાવવાના ઘણા રસ્તા છે. સૌથી સીધો રસ્તો નીચે મુજબ છે.

1. ડેસ્કટોપ ઉપર **My Computer** પર બે વાર ક્લિક કરો. **My Computer** વિન્ડો ખુલશે. અથવા **Start** પર ક્લિક કરો અને **Programs → Windows Explorer** પસંદ કરો.
2. જે ફાઈલ અથવા ફોલ્ડરનું શોર્ટકર્ટ બનાવવું હોય તે ડ્રાઈવ પર બે વાર ક્લિક કરો.
3. જે ફાઈલ અથવા ફોલ્ડરનું શોર્ટકર્ટ બનાવવું હોય તેને પસંદ કરો. ફાઈલ અથવા ફોલ્ડર પર જમણી ક્લિક કરો. જે મેનુ દેખાય તેમાં **Create Shortcut** પર ક્લિક કરો. પસંદ કરેલ ડ્રાઈવ અથવા ફોલ્ડરનો શોર્ટકર્ટ દેખાશે.(જો તમારે શોર્ટકર્ટ ને ડેસ્કટોપ પર મૂકવો હોય તો ફાઈલ અથવા ફોલ્ડર પર જમણી ક્લિક કરો. જે મેનુ દેખાય તેમાં **Send to - Desktop** પર ક્લિક કરો. શોર્ટકટ ડેસ્કટોપ પર દેખાશે.)

બનાવેલ શોર્ટકર્ટને કાઢવા માટે, તે શોર્ટકટને ખેંચીને **Recyle Bin** ઉપર લઈ જાઓ, અથવા જે શોર્ટકર્ટને કાઢવો છે તેના પર માઉસનું જમણું બટન ક્લિક કરી **Delete** પર ક્લિક કરો. શોર્ટકર્ટનું નામ બદલવા સૌ પ્રથમ બધી ખુલેલી વિન્ડો બંધ કરી તે શોર્ટકટ ઉપર જમણી ક્લિક કરી આવેલ મેનુમાંથી **Rename** પસંદ કરો. શોર્ટકર્ટનું નવું નામ આપી **Enter** દબાવો.

4.7.3 વિન્ડોઝની સૌથી સામાન્ય પ્રવૃત્તિઓ (Windows Most Common Tasks)

વિન્ડોઝના દરેક ફોલ્ડરની આકૃતિ 4.6 માં બતાવ્યા પ્રમાણે રચના હોય છે. આ રચનાને વિન્ડો કહેવાય છે. મેનુબાર, ટાઈટલ બાર, ટાસ્કબાર, સ્ક્રોલબાર અને ત્રણ ચોરસ (-) જે જમણી બાજુ ઉપર આવેલા છે તે બધી જ વિન્ડોમાં હોય છે, ભલે તમારી ફાઈલ અથવા ફોલ્ડર ચાલુ કે બંધ હોય- કોઈ એપ્લીકેશન ચાલુ કરવા અથવા ફાઈલ/ફોલ્ડરનો ઉપયોગ કરવા open કરો, જયારે કામ થઈ જાય ત્યારે **Close** કરો. તમે ટાસ્કબાર વિશે ભણેલા છો તેથી આપણે વિન્ડોઝના બીજા ભાગોની માહિતી લઈશું.

1. મિનીમાઈઝ, મેક્સીમાઈઝ/ રિસ્ટોર અને બંધ (Minimize, Maximize/Restore, and Close)

દરેક વિન્ડોની ઉપરની જમણી બાજુ ત્રણ ચોરસ હોય છે. - x બાદબાકીની નિશાનીએ મિનીમાઈઝની છે. મિનીમાઈઝ કરેલી વિન્ડો ચાલુ રહે છે, એક વિન્ડો બટન ટાસ્કબાર પર

દેખાય છે. બે-ચોરસની નિશાની ☐ મેકસીમાઈઝની/રિસ્ટોરની છે તમે તેનાથી ફાઈલને આખી વિન્ડોમાં દેખાય તેમ ગોઠવી શકો છો. તેના પર ફરીથી ક્લિક કરવાથી તે જુના માપની બને છે (તમે પાછળ ડેસ્કટોપ જોઈ શકો છો) જો ઘણી બધી ફાઈલો ખુલ્લી હોય તો મેકસીમાઈઝથી સર્વોચ્ચ પદ મળે છે અને બાકી બધી વિન્ડો ઢંકાઈ જાય છે. રિસ્ટોર કરવાથી દરેક ફાઈલ ડેસ્કટોપ ઉપર દેખાશે. તમારે જેમાં કામ કરવું છે તેને પસંદ કરી લેવી. કોઈપણને બંધ કરવા ફકત X *દબાવો.*

2. ટાઇટલ બાર (Title Bar)

ટાઈટલ બારનો સ્કીનની ઉપરની બાજુ સમાવેશ થાય છે. જે પ્રોગ્રામમાં કામ કરતા હોઈએ તેનું નામ ત્યાં બતાવે છે. ટાઈટલ બારની ઉપર બે વાર ક્લિક કરવાથી વિન્ડો વિસ્તાર નાનું-મોટું કરી શકાય છે. જો ઘણીબધી વિન્ડો ખુલ્લી હોય તો ટાઈટલ બાર ઉપર નજર કરવાથી વર્તમાન વિન્ડો વિશે ખબર પડી શકે છે. તેનો ફાયદો એ છે કે તેને કાગળના ટુકડાની જેમ ડેસ્કટોપ પર ફેરવી શકાય છે. ટાઈટલ બારની ઉપર માઉસથી ક્લિક કરી એને ખેંચો (વિન્ડો મેકસીમાઈઝ ના હોવી જોઈએ, જે તે સ્કીનના માપની હશે તો ફરી નહિ શકે) કોઈ વસ્તુને ખસેડવા, ટાઈટલ બાર પર ક્લિક કરવું પડશે, માઉસ પકડી રાખી પછી ખસેડો. આ સંપૂર્ણ પ્રક્રિયાને Drag and Drop *(ડ્રેગ એન્ડ ડ્રોપ) કહે છે. જો વિન્ડો મેકસીમાઈઝ ના હોય તો ટાઈટલ બાર પર જમણી ક્લિક કરી મેનુમાંથી* Move *પસંદ કરીને વિન્ડોને ફેરવી શકાય. માઉસનો ઉપયોગ કરીને તમે વિન્ડોની સાઈઝ (માપ) પણ બદલી શકો : માઉસને વિન્ડોની ધારી ઉપર લઈ જતાં બે - ધારીવાળું પોઈન્ટર આવશે ↕થી ઊંચાઈ, ↔થી પહોળાઈ અને ↑ થી ઊંચાઈ અને પહોળાઈનું માપ બદલી શકાય છે.*

3. સ્કોલ બાર (Scroll Bar)

જ્યારે વિન્ડોનું માપ વધે ત્યારે ઊભું લંબચોરસ-સ્કોલબાર જમણી બાજુ આવે છે. જ્યારે વિન્ડોની પહોળાઈ વધી જાય ત્યારે આડુ સ્કોલબાર આવે છે. લખાણની ઉપર નજર ફેરવવા માટે સ્કોલબારને સ્થળ સુધી ખેંચી જવું, તમે સ્કોલબારના છેડે આવેલ એરોથી પણ ખેંચી શકો છો. વિન્ડોનું માપ સ્કીનના માપ કરતા વધી શકે નહીં, તેથી સ્કોલબારનો ઉપયોગ કરીને સ્કીન કરતા મોટી વિન્ડો વાપરી શકાય. સ્કોલબારનું માપએ ડોકયુમેન્ટની લંબાઈ સૂચવે છે.

4. મેનુ બાર (Menu Bar)

Windows 98 *ઘણી બધી જાતના મેનુ આપે છે જે વાપરીને તમે અલગઅલગ પ્રકારના કાર્યો કરી શકો.* Windows 98 *માં વારંવાર વપરાતું મેનુએ* Start *મેનું છે, જે તમે ભણી ગયા છો.* Start *મેનુ સિવાય બીજા ઘણા બધા વિન્ડો અને પ્રોગ્રામો છે જેઓને પોતાના મેનુ અને કમાન્ડ છે.*

પાઠગત પ્રશ્નો 4.2

1. *વિન્ડોઝ એક્સ્પોરરની ખાસીયતો શું છે ?*
2. *ફાઈલને હંમેશા માટે કાઢી નાખવાના પગલાં કયાં છે ?*
3. *શોર્ટકટ શું છે ?*

4.8 શેરિંગ ફોલ્ડર અને પ્રિન્ટર (Sharing Folders and Printers)

તમે તમારા કાગળો (ફાઈલ) અને પ્રિન્ટર જે તમારા કોમ્પ્યુટર સાથે જોડાયેલ છે તેઓને નેટવર્કમાં બીજા લોકોની સાથે શેર (એક સાથે વાપરી) કરી શકો છો. તમારી ફાઈલ અને પ્રિન્ટરને કોઈની સાથે શેર કરતાં પહેલાં નીચેના બેમાંથી કોઈ એક વાપરવાની પદ્ધતિ પસંદ કરવી પડશે :

(અ) શેર-લેવલ એક્સેસ *કન્ટ્રોલ મૂળભૂત સેટિંગ છે. તેમાં સંશાધનો વાપરવા માટે પાસવર્ડ જોઈશે.*

(બ) યુઝર-લેવલ એક્સેસ *કન્ટ્રોલથી તમે કયા વ્યકિતને સંશાધનો શેર કરવાં છે તે જણાવવાનુ કહેશે, પણ તેમાં પાસવર્ડની જરૂર નહી પડે.*

મૂળભૂત રીતે તમને શેર લેવલ એકસેસ કન્ટ્રોલ મળશે. તેને યુઝર લેવલ એકસેસ કન્ટ્રોલમાં બદલવા માટે નીચેના પગલાં અનુસરો :

i. **Start** *પર ક્લિક કરો,* **Setting → Control Panel** *પસંદ કરો, અને પછી નેટવર્ક* (Network) *પર બે વાર ક્લિક કરો.* **Access Control** *ટેબ પર ક્લિક કરો.* **Network** *ડાયલોગ બોક્ષ દેખાશે.* **User - level Access Control** *પર ક્લિક કરો.*

ii. *મળેલા વપરાશકર્તા અને ગ્રુપમાંથી, ડોમેનનું નામ લખો અથવા જે સર્વર વાપરવું છે તેનુ નામ લખો અને પછી* **ok** *કલીક કરો. તમને કદાચ તમે લખેલા ડોમેન અથવા સર્વર વિશે વધુ માહિતી પુછશે.*

iii. **Restart** *તમારું કોમ્પ્યુટર ફરીથી ચાલુ કરો.* (**Restart** *કરો.*)

ફાઈલ-શેરિંગ અને પ્રિન્ટ શેરિંગની સેવા માટે, નીચેના પગલાં અનુસરો :

i. Start *પર ક્લિક કરો,* **Settings→ Control Panel** *પસંદ કરો, અને પછી* **Network** *પર બે વાર ક્લિક કરો.* **Access Control** *ટેબ પર ક્લિક કરો. નેટવર્ક ડાયલોગ બોક્ષ દેખાશે.* **File and Print Sharing** *પર ક્લિક કરો.*

ii. *તમારે શેરિંગનો જે વિકલ્પ જોઈએ તે ચેક બોક્ષને પસંદ કરો પછી* **OK** *ક્લિક કરો, અને પછી ફરીથી* **OK** *ક્લિક કરો.* Windows 98 *ની સી.ડી. નાખવા માટેનો સંદેશો આવશે અથવા* Set up disk *જેથી તમે* File and print sharing *નાખી શકો. નવા સેટિંગને*

નોંધ

વાપરવા માટે તમારે કોમ્પ્યુટરને ફરીથી ચાલુ કરવું પડશે. File-Sharing સેવા ચાલુ થયા પછી તમે ફોલ્ડર અથવા પ્રિન્ટરને શેર કરી શકશો.

ફોલ્ડરને શેરલેવલ એકસેસ કન્ટ્રોલથી શેર કરવા માટે નીચેનાં પગલાં અનુસરો :

i. **My Computer** માં તમારે જે ફોલ્ડરને શેર કરવું છે તેના પર જમણું ક્લિક કરો, અને પછી સબમેનુમાંથી **Sharing** ક્લિક કરો અને પછી **Shared As** પર ક્લિક કરો.

ii. **Share Name** ની અંદર ફોલ્ડર માટેનું નામ લખો. **Comment** ની અંદર ફોલ્ડરને લગતી માહિતી ટૂકમાં લખી શકો છો.

iii. **Access Type** માં Read only ક્લિક કરો અથવા Full depends on Password તમે કોઈપણ પ્રકારનો Access Type પસંદ કર્યો હોય, પણ તમને પાસવર્ડ દાખલ કરવા માટે વિકલ્પ હશે.

iv. જો પાસવર્ડ વાપરવો હોય તો પાસવર્ડ લખી **OK** ક્લિક કરો. ફરીથી પાસવર્ડ લખી ok કલીક કરો. ફોલ્ડર અથવા પ્રિન્ટરનો આઈકન એ ફોલ્ડર અથવા પ્રિન્ટરના હાથવાળા આઈકનથી બદલાશે, જે દર્શાવે છે કે વસ્તુ શેર થઈ ગઈ છે.

ફોલ્ડરને યુઝર લેવલ એકસેસ કન્ટ્રોલથી શેર કરવા, નીચેના પગલા અનુસરો :

ઉપર બતાવ્યા પ્રમાણે પગલાં (i) અને (ii) અનુસરો.

i. Add પર ક્લિક કરો. Add Users ડાયલોગ બોક્ષમાં તમે જેને અનુમતી આપવા માગતા હોય તેવા માણસોના નામ લખો. તમે વપરાશકર્તાની સૂચીને સ્ક્રોલ કરી શકો અથવા નામ લખશો તો સૂચી જાતે ખસશે. જો તમે નેટવર્કમાં જોડાયેલ બધાને અનુમતી આપવા (permission) માંગતા હોય તો The world ને પસંદ કરેલું રહેવા દો.

ii. તમારે પસંદ કરેલા વપરાશકર્તાને જેવા પ્રકારની રજા આપવી હોય તે પસંદ કરો. **Read Only** એટલે વપરાશકર્તા ફકત વાંચી શકશે તેમાં કોઈ ફેરફાર કરી શકશે નહી. **Full Access** એટલે વપરાશકર્તા વાંચી શકે. ડીલીટ કરી શકે અને ફાઈલ બદલી શકે. **Custom** એટલે તમે જે પ્રમાણે અધિકારો (privileges) નું મિશ્રણ કરો તે પ્રમાણે વપરાશકર્તાને મળે.

iii. જયારે, વપરાશકર્તાને પરવાનગી ઉમેરવાનું પતી જાય એટલે **OK** ક્લિક કરો.

પ્રિન્ટરને શેર-લેવલ એકસેસ કન્ટ્રોલથી શેર કરવા નીચેના પગલાં અનુસરો.

i. **My computer** માં **Printers** ફોલ્ડર પર બે વાર ક્લિક કરો. તમારે જે પ્રિન્ટર શેર કરવું હોય તેના પર જમણી ક્લિક કરો, અને પછી **Sharing** ક્લિક કરો. **Sharing tab** ઉપર **Shared As** પર ક્લિક કરો.

ii. **Share Name** ની અંદર પ્રિન્ટરનું નામ લખો. **Comment** ની અંદર પ્રિન્ટરને લગતી ટુંકી માહિતી આપી શકો.

iii. **Password** માં તમે પાસવર્ડ લખી શકો, પાસવર્ડને ચકાસવા ફરીથી લખો. **OK** પર ક્લિક કરો.

યુઝર-લેવલ એક્સેસ કન્ટ્રોલ સાથે પ્રિન્ટરને શેર કરવા, નીચેના પગલાં અનુસરો :

i. ઉપર બતાવ્યા પ્રમાણે પગલાં (i) અને (ii) અનુસરો.

ii. **Add** પર ક્લીક કરો. **Add Users** ડાયલોગ બોક્ષમાં, તમારે જેને અનુમતી આપવી છે તેવા માણસોના નામ પર ક્લિક કરો. તમે વપરાશકર્તાની સૂચીને સ્ક્રોલ કરી શકો અથવા નામ લખશો તો સૂચી જાતે ખસશે. જો તમે નેટવર્કમાં જોડાયેલ બધાને અનુમતી આપવા માંગતા હોય તો The world ને પસંદ કરેલું રહેવા દો.

iii. **Full Access** પર ક્લિક કરો. તમે જે માણસોના નામ પસંદ કરેલા હશે તેઓ સ્ક્રીન પર જમણી બાજુએ આવેલા ખાનામાં ખસશે. જયારે તમારૂં વપરાશકર્તા ઉમેરવાનું અને અનુમતી આપવાનુ પતી જાય ત્યારે **OK** ક્લિક કરો.

જો તમને સાચી પરવાનગી મળી હશે તો, તમે બીજા કોમ્પ્યુટર પરથી શેર થયેલા ફાઈલો અને પ્રિન્ટરો વાપરી શકશો. શેર કરેલા **ફોલ્ડર અને પ્રિન્ટરને વાપરવા,નીચેના પગલાં અનુસરો :**

i. Network Neighborhood પર બે વાર ક્લિક કરો.

ii. તમારે જે ફોલ્ડર અથવા પ્રિન્ટરને વાપરવા છે તેને કન્ટ્રોલ કરતા કોમ્પ્યુટર પર બે વાર ક્લિક કરો. તમે જે કોમ્પ્યુટર શોધો છો એ જો તમારા Workgroup અથવા domain માંના હોય તો પહેલાં Entire Network પર ક્લિક કરો અને પછી યોગ્ય Workgroup અથવા domain પર બે વાર ક્લિક કરો.

iii. શેર કરેલા ફોલ્ડર અથવા પ્રિન્ટરને વાપરવા, તેના આઈકન પર બે વાર ક્લિક કરો.

4.9 MS-DOS આધારીત પ્રોગ્રામ (MS-DOS Based Program)

તમે MS-DOS આધારીત પ્રોગ્રામને Programs મેનુમાં રહેલા MS-DOS Command Prompt વાપરીને ચલાવી શકો છો. **Start** પર ક્લિક કરો અને પછી **Programs → MS-DOS Command Prompt** પસંદ કરો. આ MS-DOS Command વિન્ડો ખોલશે જેની અંદર તમે MS-DOS આધારિત પ્રોગ્રામ ચલાવી શકો છો.

MS-DOS Command વાપરવા : અમે નીચે થોડાક ઉદાહરણો આપ્યા છે જે સમજાવે છે કે **MS-DOS આધારીત કમાન્ડ કેવી રીતે વાપરવા :**

1. ચકાસી લો કે MS-DOS વિન્ડો એકટીવ (ખુલ્લી) હોય.
2. MS-DOS Prompt (C:\) ઉપર DIR લખો અને પછી Enter દબાવો. હાર્ડ ડિસ્કમાં રહેલા તમારા ફોલ્ડરની અંદર જે ફાઈલો અને ફોલ્ડરો હશે તેની સૂચી બતાવશે. DIR કમાન્ડ એટલે "directory" અને તે તમારા વર્તમાન ફોલ્ડરમાં રહેલી બધી ફાઈલો અને ફોલ્ડરોની સૂચી બતાવશે.
3. પછી MS-DOS Prompt પર "CD Windows" લખો અને Enter દબાવો. તે જ ડ્રાઈવમાં ફોલ્ડર બદલાઈને windows ફોલ્ડરમાં બદલાઈ જશે. (જો હોય તો) સી.ડી.કમાન્ડ એટલે "Change Directory" બેકસ્લેસ (\ ) અને Windows એ windows ફોલ્ડરનો ડીરેકટરી પાથ (directory Path) બતાવે છે.

   નીચેનુ કોષ્ટક MS-DOS કમાન્ડની સૂચી આપે છે.

નોંધ

ટેબલ 4.1

DOS Command *ની સૂચી*

Command / ?	DOS Command "command" *માટે સ્વીચોની યાદી આપો*
Path = c:\ windows ; C:\ dos	*બતાવે છે કે ડોસ કમાન્ડ અથવા પ્રોગ્રામને કયાં ફોલ્ડરમાં શોધે*
Prompt Pg	*આનાથી વર્તમાનમાં ડીરેકટરી ડોસ બતાવે*
	dir *હાલમાં વપરાતી ડીરેકટરીમાં ફાઈલોની યાદી બતાવે*
dir/w	*પાંચ કોલમમાં ફાઈલોની યાદી બતાવે*
dir/p	*ફાઈલો બતાવે, એક સમયે એક પાનું*
dirx.exe	*જે ફાઈલોનો અંત* "exe" *થી થાય તેની યાદી બતાવે*
dirz = *???*.exe	*એવી* "exe" *ફાઈલોની યાદી આપે જેનું ચાર અક્ષરોનું નામ હોય અને જેમાં પ્રથમ અક્ષર* z *હોય.*
dir/ winsock.dll/s	*વર્તમાન ડીરેકટરીમાં* "winsock-all" *ફાઈલને શોધે*
type file.txt	file.txt" *ફાઈલમાં રહેલી માહિતી બતાવે.*
edit file-txt,	Dos 5 editor *નો ઉપયોગ કરીને* ' file-txt *ફાઈલમાં ફેરફાર કરી શકાય.*
a:	*ડ્રાઈવને બદલીને* -> a : *કરે*
md c : \ myfiles	"myfiles" *નામની સબડિરેકટરી બનાવે*
cd c : \ myfiles	"myfiles" *સબડીરેકટરીનાં બદલે*
rd c : \ myfiles	"myfiles" *નામની વર્તમાન સબડીરેકટરીને કાઢી નાખે*
del file.txt	"file.txt" *નામની ફાઈલને કાઢી નાખે (ડીલીટ કરે)*
ren f1 f2	"f1" *ફાઈલનું નામ* "f2" *કરે.*
copy f1 f2	"f1" *ફાઈલને* f2 *માં કોપી કરે.*
verify on	*કોપી કમાન્ડની ચકાસણી ચાલુ કરે.*
verify off,	*કોપી કમાન્ડની ચકાસણી બંધ કરે.*
xcopy d1 d2/s,	*બધી ફાઈલો અને સબડીરેકટરીને* "d1" *ડીરેકટરીમાં કોપી કરે*
xcopy d1 d2/p	*દરેક ફાઈલને કોપી કરતાં પહેલા ચકાસણી માગશે*
diskcopy a : b:,	*બે ફલોપી ડ્રાઈવના ઉપયોગથી ડુપ્લીકેટ ડીસ્ક બનાવે*
xcopy a: a :,	*તે જ ફલોપી ડ્રાઈવના ઉપયોગથી ડુપ્લીકેટ ડીસ્ક બનાવે*

format a: / s:,	બુટેબલ ફલોપી ડીસ્કને ફોર્મેટ કરે (સિસ્ટમ સાથે)
back up c:\d1*.	txt, a : "c:\d1\" માં રહેલી બધી ફાઈલો જેનો અંત ".txt"
backup a :	હોય તેનો બેકઅપ "a:"માં રહેલી ફલોપીમાં લેશે
backup :c : \ /s	આખા C: ડ્રાઈવને a : ફલોપી ડ્રાઈવમાં બેકઅપ લેશે.
restore a : c:\d1\x.txt	a: માંથી કેટલીક ફાઈલોને c: માં પાછી લાવે (restore કરે)
restore a :c:\ / s	બેકઅપ ફાઈલો અને સબડીરેકટરીને પાછી લાવે ver DOS નું વર્ઝન બતાવે
time	સિસ્ટમ સમયને ચકાસે અથવા સુધારે
date	સિસ્ટમની તારીખને ચકાસે અથવા સુધારે
cls	સ્કિનને ચોખ્ખી કરે
scandisk, c :	ડીસ્કને સ્કેન કરે અને ચકાસે
chkdsk,	વર્તમાન ડીસ્કને તપાસે અને મેમરીનો વપરાશ ચકાશે
chkdsk /f chkdsk	ધ્વારા બતાવેલ ભૂલોને સુધારે
chkdsk filename,	કોઈ ચોકકસ ફાઈલને તપાસે
chkdsk a :	ખાસ ડીસ્કને ચકાસે (a : ડ્રાઈવમાં રહેલ ફલોપી)
mem	મેમરીનો વપરાશ ચકાસે

નોંધ : *ડ્રાઈવના શબ્દને તમારા કોમ્પ્યુટરની જરૂરીયાત મુજબ બદલો. તમે* DOS *ના કમાન્ડને નાના* (lower) *અથવા મોટા* (upper) *અક્ષરોથી લખી શકો છો કારણ કે* DOS *તે બંનેને સરખા સમજે છે.*

4.10 તમે જે શીખ્યા *(what you have learnt)*

આ પાઠમાં તમે ઓપરેટીંગ સિસ્ટમની જરૂરીયાત અને તેના કાર્યો વિશે શીખ્યા. ખાસ કરીને તમે Windows 98 *ના કાર્યો શીખ્યા.* Windows *એક* GUI *છે જેમાં ચિત્રો, ચિન્હો અને શબ્દોને કોમ્પ્યુટર મોનીટર પર વપરાય છે જેને માઉસ ધ્વારા નિયંત્રિત કરી શકાય છે. તે કોમ્પ્યુટરના આંતરીક કાર્યોને સંભાળે છે અને કોમ્પ્યુટર મેમરી ઈનપુટ અને આઉટપુટ કાર્યોનું ધ્યાન રાખે છે. હવે તમે કોમ્પ્યુટર વાપરવા સક્ષમ છો.*

4.11 પાઠયાંત પ્રશ્નો

1. My Computer *ની* Properties Window *માં આવેલા નીચેના* Tools tab *વિકલ્પોની ઉપયોગીતાઓ જણાવો :*
   (અ) એરર ચેકિંગ સ્ટેટસ (Error-Checking Status)
   (બ) બેકઅપ સ્ટેટસ અને ડી-ફ્રેગમેન્ટેશન સ્ટેટસ (Backup status and fragmentction status)
2. Windows 98 *માં નીચેનાઓની ઉપયોગીતા શું છે ?*
   (અ) માય ડોકયુમેન્ટ (My document)
   (બ) રીસાયકલ બીન (Recycle Bin)
   (ક) ઈન્ટરનેટ એકસપ્લોર (Internet Explorer) *અને*

(ડ) નેટવર્ક નેબરહૂડ (Network Neighborhood)

3. Start બટન અને Taskbar ના કાર્યો ટૂકમાં સમજાવો.
4. ફોલ્ડર બનાવવાનાં પગલાં સમજાવો.
5. ફાઈલ અથવા ફોલ્ડરને ખોલવાના પગલાં સમજાવો.
6. નીચેના સમજાવો :

   (અ) મીનીમાઈઝ, મેકસીમાઈઝ/રીસ્ટોર અને કલોઝ

   (બ) ટાઈટલ બાર (Title Bar)

   (ક) સ્ક્રોલ બોક્ષ (Scroll Box) અને

   (ડ) મેનુબાર (Menu Bar)
7. બીજાઓ ધ્વારા કન્ટ્રોલ કરેલા ફાઈલ અને પ્રિન્ટરને વાપરવાના પગલાં સમજાવો.
8. શોર્ટકટ શું છે ? તમે શોર્ટકટ કેવી રીતે બનાવશો ?

4.12 પાઠગત પ્રશ્નોના જવાબો.

પાઠગત પ્રશ્નો 4.1

1. વપરાશકર્તા અને કોમ્પ્યુટર વચ્ચે કળી (link) બનાવે. તે આંતરિક કાર્યો જેવા કે કોમ્પ્યુટરની મેમરીને જાળવે છે. તે ગ્રાફીકલ યુઝર ઈન્ટરફેસ છે. વર્લ્ડ વાઈડ વેબએ વિન્ડોઝના ઉપયોગથી કોઈપણ જગ્યાએથી સરળતાથી વાપરી શકાય છે. તે ડીવીડી અને ડીજીટલ ઓડીયોને મદદ કરે છે.
2. (અ) માઉસને કોઈ વસ્તુની ઉપર લઈ જવાની પ્રક્રિયાને પોઈન્ટીંગ કહે છે.

   (બ) સ્ક્રીન પર આવેલી કોઈ વસ્તુ પર જવુ અને માઉસનું ડાબુ બટન ઝડપથી દબાવી અને છોડી દેવાની ક્રિયાને કલીકીંગ (clicking) કહે છે.

   (ક) સ્ક્રીન પર આવેલી કોઈ વસ્તુ પર જવુ અને માઉસનું બટન ઝડપથી બે વાર દબાવી અને છોડી દેવાની ક્રિયાને ડબલ કલીકીંગ (double cliking) કહે છે.

   (ડ) સ્ક્રીન પર આવેલી કોઈ વસ્તુ પર જવુ અને માઉસનું બટન દબાવી રાખીને ખસેડવાની ક્રિયાને ડ્રેગીંગ (dragging) કહે છે.
3. ડેસ્કટોપ એ જગ્યા છે જયાં વપરાશકર્તાને ઉપયોગી વસ્તુઓ વાપરવા માટે મુકેલી હોય, તેનો આધાર ઈન્સ્ટોલેશન વખતે પસંદ કરેલા વિકલ્પો પર હોય.
4. My Computer માં હોય છે :

   હાર્ડ ડ્રાઈવ 'C', ફલોપી ડ્રાઈવ, CD ROM ડ્રાઈવ, પ્રિન્ટરો, કન્ટ્રોલ પેનલ, ડાયલ અપ નેટવર્કીંગ અને શિડયુલ્ડ કાર્યો

પાઠગત પ્રશ્નો 4.2

1. ડ્રાઈવ અને ફોલ્ડરને અલગ વિન્ડોમાં ખોલવાને બદલે વપરાશકર્તા એક જ વિન્ડોમાં માહિતીને હાયરાર્ચીકલ સ્ટ્રકચર પ્રમાણે જોઈ શકે છે.
2. ડેસ્કટોપ પર રીસાયકલ બિન (Recycle Bin) પર બે વાર કલીક કરો. Recycle Bin વિન્ડો ખુલશે તમારે જે ફાઈલ અથવા ફોલ્ડરને હંમેશ માટે કાઢી નાખવા છે તેને પસંદ કરો. ટુલબારમાં Delete પર કિલક કરો. Confirm File Delete ડાયલોગ બોક્ષ દેખાશે. પછી Yes પર કિલક કરો. તમારા કોમ્પ્યુટરમાંથી ફાઈલ હંમેશા માટે ડીલીટ થઈ જશે.
3. શોર્ટકટ એ ફકત એક પોઈન્ટર છે જેનાથી ફાઈલ ઝડપથી ખોલી શકાય. તમે શોર્ટકટ ડીલીટ કરો તો મૂળ ફાઈલ ડીલીટ થતી નથી.

5

માહિતી સંચાર અને નેટવર્કીંગ

5.1 પ્રસ્તાવના

આજે કોમ્પ્યુટર ઘણી કચેરીઓ અને ઘરોમાં ઉપલબ્ધ છે, અને તેથી વિવિધ કોમ્પ્યુટરો વચ્ચે માહિતી અને પ્રોગ્રામ શેર (સાથે કામ કરી શકાય તેવું) કરવાની જરૂર છે.માહિતી સંચાર સુવિધાઓની ઉન્નતિ સાથે કોમ્પ્યુટર્સ વચ્ચે સંદેશ વ્યવહારનો વધારો થયો છે અને તેથી કોમ્પ્યુટરની શક્તિ કોમ્પ્યુટર રૂમની બહાર વધી છે. હવે વપરાશકર્તા એક જગ્યાએ બેઠો હોય અને તે પ્રત્યાયન ચેનલ મારફતે કોઈ પણ કોમ્પ્યુટર સાથે વાતચીત કરી શકે છે. આ પાઠનો ધ્યેય કોમ્પ્યુટર નેટવર્કના વિવિધ પાસાઓ રજુ કરવાનો છે.

5.2 હેતુઓ

આ પાઠ નો અભ્યાસ કર્યા પછી, તમે નીચેની પ્રક્રિયા માટેની સ્થિતિમાં હશો :

- *માહિતી સંચાર સિસ્ટમના મૂળભૂત તત્વો જાણવા*
- *પ્રત્યાયન પ્રોટોકોલ અને માહિતી પ્રસારણની સ્થિતિ વર્ણવવા.*
- *કોમ્પ્યુટર નેટવર્કના ઉપયોગ સમજવા.*
- *કોમ્પ્યુટર નેટવર્કના વિવિધ ઘટકો વર્ણવવા.*
- *વિવિધ પ્રકારના નેટવર્કને ઓળખવા.*
- *ઈન્ટરનેટ અને ઈ- મેલનો આધુનિક યુગમાં ઉપયોગ સમજવો.*
- *સેટેલાઈટ કોમ્યુનિકેશનના ઉપયોગની કદર*

5.3 માહિતી સંચાર (Data Communication)

આપણને આપણા રોજીંદા જીવનમાં કેટલાક પ્રકારના સંચારની જરૂર પડે છે. માહિતી અને સંદેશા વ્યવહાર માટે આપણે ટેલીફોન અને પોસ્ટનો સંચાર સિસ્ટમ તરીકે ઉપયોગ કરીએ છીએ. એ જ રીતે એક કોમ્પ્યુટર સિસ્ટમની માહિતી બીજા સીસ્ટમમાં મોકલી શકાય છે અને તે કોઈ પણ ભૌગોલિક વિસ્તારમાં મોકલી શકાય. આ માહિતી મોકલવા કેટલાક પ્રમાણભૂત પધ્ધતિઓના ઉપયોગ થાય છે. આ પધ્ધતિઓમાં કંડકટર સાથે ઈલકેટ્રોનિક સંકેતોનો, ઓપ્ટિકલ ફાયબર અને ઈલકેટ્રોનિક વિસ્તારો નો સમાવેશ થાય છે.

ધારો કે મેનેજરને વિવિધ ગ્રાહકોને માટે ઘણા પત્રો લખવા છે. પ્રથમ તેમણે પત્ર તૈયાર કરવા માટે પીસી અને પ્રોસેસિંગ પેકેજનો ઉપયોગ કરવો પડશે, પીસી જો નેટવર્કીંગ મારફતે બધા ગ્રાહકના પીસી સાથે જોડાયેલ હોય, તે તમામ ગ્રાહકને પત્રો મોકલી શકે અને તે પણ થોડીક મિનિટોમાં જ, આમ કોઈ પણ વિસ્તારમાં, જો પીસી પ્રત્યાયન ચેનલ મારફતે જોડાયેલ હોય, ડેટા અને માહિતી, કોમ્પ્યુટર ફાઈલો અને અન્ય પ્રોગ્રામ અન્ય કોમ્પ્યુટર પર સેકેન્ડોમાં મોકલી શકાય છે. ઈમેલ અને ઈન્ટરનેટ જેવી સંચાર સુવિધા, આધુનિક યુગમાં માત્ર કોમ્પ્યુટર નેટવર્કીંગથી જ શક્ય છે.

સંચાર પધ્ધતિના મૂળભૂત તત્વો (Basic Elements of a communication System)

એક સમયના કામ માટે નીચે પ્રમાણેની મૂળભૂત પધ્ધતિની જરૂરિયાત હોય છે.

1. પેષક (સ્ત્રોત) જે સંદેશા બનાવે.
2. એક માધ્યમ જે સંદેશને વહન કરે.
3. રીસીવર (મેળવનાર) જે સંદેશને મેળવે છે.

માહિતી વાતચીત કરવામાં ચાર મૂળભૂત શબ્દો વાંરવાર ઉપયોગ કરવામાં આવે છે. તેઓ આ પ્રમાણે છે :

- **માહિતી (Data)** : હકીકતોનો સંગ્રહ જે કાચા સ્વરૂપમાં હોય છે પછી તેને પ્રોસેસીંગ કરવામાં આવે છે.
- **સંકેતો (Signals)** : માહિતીના ઈલેકટ્રીક અથવા ઈલેકટ્રોમેગ્નેટીક એન્કોડિંગ
- **સંકેતો (Signaling)** : સંકેતોની પ્રક્રિયા દ્વારા પ્રાપ્ત માહિતીના કોમ્યુનિકેશન.
- **પ્રવાહન (Transmission)** : સિગ્નલના પ્રોસેસીંગ દ્વારા માહિતીનો સંચાર

5.4 કોમ્યુનિકેશન પ્રોટોકોલ (Communication Protocol)

તમને કદાચ આશ્ચર્ય થશે, કોમ્પ્યુટર માહિતી કેવી રીતે મોકલે અને મેળવે છે. જેનો જવાબ ડેટા સંચાર સોફટવેર છે આ તેજ સોફટવેર છે, જે અન્ય સિસ્ટમો સાથે વાતચીત કરવા માટે સુવિધા પુરી પાડે છે. સંચાર સોફટવેર કોમ્પ્યુટર સિસ્ટમો અને ઉપકરણોને કેવી રીતે એક જગ્યાએથી બીજી જગ્યાએ સ્થાનાંતરિત કરવી તેની સૂચના આપે છે. આ રીતે સોફટવેર સ્વરૂપમાં માહિતીનું રૂપાંતર સામાન્ય રીતે પ્રોટોકોલ તરીકે ઓળખાય છે. ડેટા ટ્રાન્સમિશન સોફટવેર અથવા પ્રોટોકોલ

નીચેની કામગીરી માહિતીના કાર્યક્ષમ અને ભુલ વગર ટ્રાન્સમિશન માટે કરે છે.

1. **માહિતી ક્રમ** :*એક લાંબા સંદેશાને વિભાજીત કરી નાના પેકેટોમાં વહન કરી ભુલ વગર પ્રસારણ માટે.*

2. **માહિતી રાઉટિંગ** :*તે ડેટા મોકલતા પહેલા સ્રોત અને અંતિમ મુકામ વચ્ચે, સૌથી વધુ અસરકારક માર્ગ શોધવાની પ્રક્રિયા છે.*

3. **પ્રવાહ નિયંત્રણ** :*બધા મશીનો ઝડપની દૃષ્ટિએ સમાન કાર્યક્ષમ ન હોય તેથી તેના પ્રવાહ નિયંત્રણ વચ્ચે માહિતી મોકલવાની પ્રક્રિયા નિયમન ફાસ્ટ મોકલનાર અને ધીમા રીસીવરથી થાય છે.*

4. **ભૂલ નિયંત્રણ** : *ભુલ શોધવી અને પુન : પ્રાપ્તિ એ પ્રત્યાયન સોફટવેરનું મુખ્ય કાર્ય છે એ ખાતરી કરે છે કે માહિતી કોઈ પણ ભૂલ વિના મોકલાય છે.*

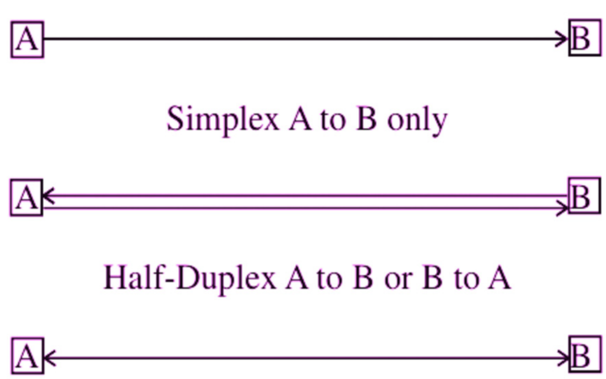
5.5 માહિતી ટ્રાન્સમિશન સ્થિતિઓ

આકૃતિ 5.1 માં બતાવ્યા પ્રમાણે એક બિંદુ પરથી માહિતી પરિવહન માટે ત્રણ વિકલ્પો છે.

1. **સિમ્પલેક્સ** : *સિમ્પલેકસ સ્થિતિમાં માહિતીનું વહન કોઈ એક જ દિશામાં થાય છે. રીસીવર આ વહન ઉપકરણમાંથી સિગ્નલ મેળવે છે. આ સ્થિતિમાં માહિતીનો વહાવ એક જ દિશામાં હોય છે. તેથી તે ભાગ્યે જ માહિતી સંદેશા વ્યવહાર માટે વપરાય છે.*

2. **અર્ધ બેતરફી** :*અર્ધ બે તરફી સ્થિતિમાં સંચાર ચેનલોનો ઉપયોગ બંને દીશામાં થાય છે, પરંતુ એક સમયે માત્ર એક જ દિશા. આમ અર્ધ બેતરફી સ્થિતિમાં વાકયો એકાંતરે મોકલી અને માહિતી પ્રાપ્ત કરી શકાય છે.*

3. **સંપૂર્ણ દ્વિગુણિત** :*સંપૂર્ણ દ્વિગુણિત સ્થિતિમાં સંચાર તે જ સમયે બન્ને દિશાઓમાં થાય છે. સંપૂર્ણ દ્વિગુણિત સ્થિતિમાં કાય ક્ષમતા વધે છે, કારણ કે અર્ધ - દ્વિગુણિતની જેમ જરૂરી લાઈન ટર્નઅરાઉન્ડ **(જઈને પાછા આવવું)** સમય તરીકે દૂર થાય છે. આ ટ્રાન્સમિશન સ્થિતિનું ઉદાહરણ ટેલીફોન છે.*



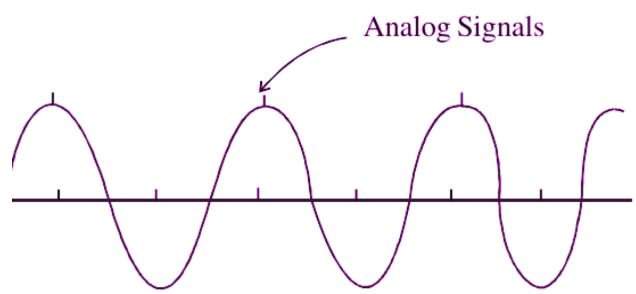
આકૃતિ 5.1. સિમ્પલેક્સ, અર્ધ - બે તરફી, સંપૂર્ણ દ્વિગુણિત

5.5.1 ડિજિટલ અને એનાલોગ ટ્રાન્સમિશન

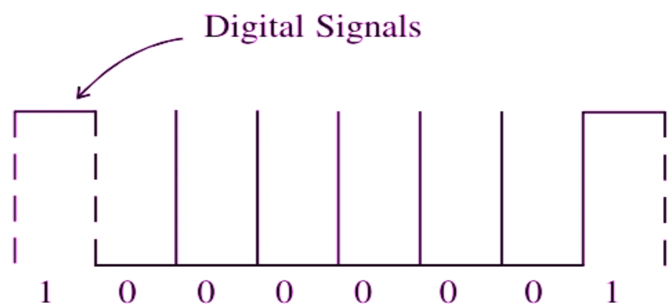
માહિતી એક માધ્યમથી બીજા માધ્યમ સુધી વિદ્યુત સંકેતોથી ફેલાય છે, જે ડિજીટલ અને એનાલોગ સ્વરૂપમાં હોય શકે છે. તેથી એનાલોગ અને ડિજિટલ સંકેતો વચ્ચેના તફાવતની ખબર હોવી જોઈએ. એનાલોગમાં ટ્રાન્સમિશનમાં સિગ્નલ પાવર અવાજ, પ્રકાશ અને રેડિયો તંરગો સાથે સતત બદલાય છે. બીજી તરફ, ડિજિટલ સિગ્નલની કિંમત આપેલ શ્રેણીમાંથી ફકત સ્વતંત્ર સમૂહ ધારણ કરી શકે છે. (જુઓ આકૃતિ. 5.2 અને 5.3) ઉદાહરણોના કોમ્પ્યુટર અને સંબંધિત કોમ્પ્યુટરના સાધનો છે. એનાલોગ સિગ્નલ વોલ્ટસમાં માપવામાં આવે છે અને તેની આવૃત્તિ હર્ટઝમાં હોય છે. ડિજિટલ સિગ્નલ વોલ્ટેજ એક બાઈનરી સ્વરૂપે રજુ થાય છે. જયારે ડિજીટલ માહિતી એનાલોગ ફોર્મમાં થવું જરૂરી છે અને જે ટેકનિકથી ડિજિટલને સિગ્નલને એનાલોગમાં ફેરવાય છે, તેને મોડયુલેશન તરીકે ઓળખાય છે, અને રિવર્સ પ્રક્રિયા, કે જેમાં એનાલોગ ફોર્મનું ડિજિટલમાં રૂપાંતર થાય છે. તેને ડીમોડયુલેશન તરીકે ઓળખાય છે. જે ઉપકરણ એનાલોગમાંથી ડિજિટલ સિગ્નલ ફેરવે છે તેને મોડેમ કહેવાય છે.

5.5.2 અસુમેળ અને સુમેળ ટ્રાન્સમિશન (Asynchronous and synchronous Transmission)

એક માધ્યમ મારફતે પ્રસારણ અસુમેળ અથવા સુમેળ પ્રકારનું હોય શકે છે. જો તમે કીબોર્ડથી ટાઈપ કરતા હોય તો સુમેળ ટ્રાન્સમિશનમાં માહિતી અક્ષર દ્વારા ફેલાય છે. આથી અક્ષરો વચ્ચે જગ્યા અનિયમિત હોય છે. જો કે તે અમલ કરવામાં સસ્તી છે, કેમ કે તેમાં ડેટા સેવ કરવાની જરૂર નથી. બીજી તરફ, અસુમેળ સ્થિતિમાં, ડેટા સાચવીને માહિતી ફેલાય છે અને તે પણ બ્લોક દ્વારા થાય છે. દરેક બ્લોક ઘણા અક્ષરો સમાવી શકે છે. સુમેળ ટ્રાન્સમિશન દુરસ્થ સંચાર માટે યોગ્ય છે, જેમ કે કોમ્પ્યુટર, કાર્ડ રીડર અને પ્રિન્ટરો જેવા સંબંધિત ઉપકરણો માટે.



આકૃતિ.5.2 એનાલોગ સિગ્નલ



આકૃતિ. 5.3 ડિઝિટલ સિગ્નલ

5.6 કોમ્યુનિકેશન સેવાઓના પ્રકાર

વાર્તાલાપ સેવાના ડેટા સંભાળવાની ક્ષમતાના વર્ણન માટે વપરાતા શબ્દને બેન્ડવીડથ કહે છે. માહિતીના પ્રસારણ માટેની ફ્રિકવન્સીઝ શ્રેણી માટે બેન્ડવીડથ હોય છે. એક વાર્તાલાપ સીસ્ટમમાં ફ્રિકવન્સીઝની સાંકડી રેન્જને નાના વ્યાસ વાળી બગીચામાં વપરાતી નળી સાથે સરખાવી શકાય. આવા સિસ્ટમના માહિતીના પ્રવાહ તેની માહિતીનો દર પ્રતિબંધિત છે, જેમ કે સાંકડી ટોટીમાં પાણીનો પ્રવાહ હોય છે. વિશાળ બેન્ડવીડથ વધુ ઝડપી માહિતી પ્રવાહની પરવાનગી આપે છે. આ વાતચીત ડેટા ટ્રાન્સફરનો દર baud *માં માપવામાં આવે છે. બોડ દર બીટસ પ્રતિ સેકન્ડ જેવું છે તેથી* 300 *બોડ એટલે* 300 *બિટ્સ પ્રતિ સેકન્ડ હોય છે.*

પ્રત્યાયન કંપનીઓ જેવી કે અમેરિકન ટેલિફોન અને ટેલિગ્રાફ (એટીએન્ડ ટી) અને વેસ્ટર્ન યુનિયન સામાન્ય કેરિયર કહેવાય છે અને તેઓ ત્રણ સામાન્ય સર્વિસ જેમ કે વોઈસ અને માહિતી સંચાર માટેના વર્ગો પૂરા પાડે છે.

1. Narrowband *નીચા માહિતી વોલ્યુમમાં સંભાળે છે. માહિતી પ્રસારણનો દર* 45 *થી* 300 *બોડ હોય છે ઓછી સ્પીડ ધરાવતા ઉપકરણની સાંકડી બેન્ડ સંચારનો ઉપયોગ કરે છે.*
2. Voiceband 300 *અને* 9600 baud *ની વચ્ચેના માહિતી પ્રસારણ વોલ્યુમમાં સંભાળે છે. તેઓ સી.આર.ટી., લાઈન પ્રિન્ટર સુધીના એપ્લીકેશન માટે વપરાય છે. તેમની મુખ્ય એપ્લિકેશન ટેલીફોન અવાજ માટે છે. તેથી સંચાર, શબ્દ વોઈસ બેન્ડ છે.*
3. *બ્રોડબેન્ડ માહિતીના ખૂબ મોટા વોલ્યુમો સંભાળે છે. આ સિસ્ટમો પૂરા* 1 *મિલિયન બોડ કે તેથી વધુ માહિતી પ્રસારણ માટે છે. હાઈ - સ્પીડ ડેટા સંચાર સિસ્ટમનો વિશ્લેષણ અને ઉપગ્રહ સંચાર , બ્રોડબેન્ડના ઉદાહરણો છે.*

5.7 કોમ્યુનિકેશન મીડિયા

નીચે પ્રમાણેના મોટા સંચાર ઉપકરણોને વાંરવાંર ઉપયોગ કરવામાં આવે છે.

1. **વાયર જોડી :** *વાયર જોડીઓ સામાન્ય સ્થાનિક ટેલિફોનમાં વપરાય છે. અને તે પણ ટૂંકા અંશના ડિજિટલ માહિતીની સંચાર સંદેશા વ્યવહાર માટે વપરાય છે. તેઓ સામાન્ય રીતે કોપરની બનેલી છે. અને વાયરોની જોડી એક સાથે વળાંક વાળી હોય છે. માહિતી પ્રસારણની ઝડપ* 100 *મીટરની અંદર પ્રતિ સેકન્ડે* 9600 *બિટસ હોય છે.*
2. **સમાન ધરીવાળું કેબલ્સ (Coxial Cables) :** *સમાન ધરી વાળું કેબલ એ ખાસ અવરિત જુથો સાથે ઉંચા દરે ડેટા ટ્રાન્સફર કરવા માટે સક્ષમ હોય છે. જેમાં કેન્દ્રીય કોપર વાયર ઈન્સ્યુલેશનથી ઘેરાયેલો હોય છે. અને સાથે જાળીદાર આવરણ પણ હોય છે. તેઓ લાંબા અંતરના ટેલિફોન વાયર અને સ્થાનિક માટે વપરાય છે કારણ કે તેઓ અવાજ પ્રતિરક્ષા અને ઝડપી ડેટા ટ્રાન્સફર માટે સક્ષમ છે.*
3. **માઈક્રોવેવ :** *માઈક્રોવેવ સિસ્ટમ ખૂબ ઉંચા આવર્તન રેડિયો સંકેતો વાપરે છે. જેના મારફતે માહિતી વહન થઈ શકે. ટ્રાન્સમીટર અને રિસીવર માઈક્રોવેવ સિસ્ટમમાં એક બીજાની*

નોંધ

સામ સામે હોવા જોઈએ કારણ કે રેડીયો સિગ્નલ વળતા નથી. માઈક્રોવેવ સાથે ખૂબ લાંબા અંતરનું પ્રસારણ શકય નથી. સામ સામે (લાઈન ઓફ સાઈટ) ની સમસ્યાઓ દૂર કરવા માટે તથા નબળા સિગ્નલ માટે એમ્પલીફિકેશન, 25 થી 30 કિ.મીના અંતરે રીપીટર વપરાય છે.

4. **કોમ્યુનિકેશન સેટેલાઈટ :** લાઈન દ્દષ્ટિ અને રીપીટરની સમસ્યાનો ઉકેલ ઉપગ્રહનો ઉપયોગ કરીને દૂર કરી શકાય, જે આધુનિક સમયના પ્રસારણ માપન તરીકે સૌથી વધુ ઉપયોગમાં લેવાય છે. આ એક વાર્તાલાપ ઉપગ્રહ છે. જેને માઈક્રોવેવ રિલે સ્ટેશન તરીકે બ્રાહ્ય અવકાશમાં મૂકવામાંઆવે છે. ઈનસેટ - 1 (બી) એક એવો ઉપગ્રહ છે જે ભારતમાં ગમે ત્યાંથી ઉપયોગમાં લઈ શકાય છે. સેટેલાઈટ માઈક્રોવેવનું પ્રત્યાયન ટ્રાન્સીમીટર પરથી થાય છે. સેટેલાઈટ નબળા સિગ્નલમાં વધારો કરી પાછા રિસીવર માટે તેનું પ્રસારણ કરે છે. સેટેલાઈટનો મુખ્ય લાભ એ છે કે એક જ માઈક્રોવેવ રીલે સ્ટેશનની જરૂર પડે છે. માઈક્રોવેવમાં માહિતી પ્રસારણનો દર પ્રતિ સેકન્ડે 16 ગીગા બીટસ છે. તેઓ મોટા ભાગે મેટ્રોપોલિટન શહેરોને જોડવા કરવા માટે વપરાય છે.

પાઠગત પ્રશ્નો - 5.1

---

1. કોમ્યુનિકેશન પ્રોટોકોલ શું છે ?
2. અસુમેળ અને સુમેળ ટ્રાન્સમિશન વચ્ચે શું તફાવત છે ?
3. નીચેના વિધાન માટે સાચુ કે ખોટું લખો.

   (અ) એક વાર્તાલાપ સિસ્ટમની કામ માટે મૂળભૂત જરૂરિયાતો મોકલનાર માધ્યમ અને રીસીવર હોય છે.

   (બ) માહિતીના ઈલેક્ટ્રોનિક અથવા ઈલેક્ટ્રોમેગ્નેટિક એન્કોડિંગને ટ્રાન્સમીશન કહેવામાં આવે છે.

   (ક) સંપૂર્ણ દ્વિગુણીતમાં સંચાર ચેનલ તે જ સમયે બંને દિશામાં કામ કરે છે.

   (ડ) એનાલોગ સિગ્નલ વોલ્ટસમાં મપાય છે અને તેની આવૃત્તિ હર્ટ્ઝમાં હોય છે.

   (ઈ) ડિજિટલ સિગ્નલના ફોર્મને એનાલોગમાં ફેરવવા જે ટેકનિક વપરાય છે તેને મોડયુલેશન તરીકે ઓળખવામાં આવે છે.

5.8 કોમ્પ્યુટર નેટવર્ક

એક કોમ્પ્યુટર નેટવર્ક પર સ્થિત વિવિધ કોમ્પ્યુટર સિસ્ટમો વિવિધ સ્થળોએ ઈન્ટરકનેકશનથી જોડાયેલા હોય છે. કોમ્પ્યુટર નેટવર્કમાં બે કે તેથી વધુ કોમ્પ્યુટર્સ માહિતી સંચાર માટે શેર કરેલા સંશોધનો સાથે મળીને કામ કરે છે. જે કોમ્પ્યુટર નેટવર્ક ઉપર અન્ય કોમ્પ્યુટરો માટે સંશાધનો

પૂરી પાડે છે, તે કોમ્પ્યુટર સર્વર તરીકે ઓળખાય છે. નેટવર્કની અંદર જ કોમ્પ્યુટર શેર કરેલા નેટવર્ક સ્રોતોનો ઉપયોગ કરે છે તેને નોડ તરીકે ઓળખાય છે.

5.9 નેટવર્ક્સના પ્રકાર

નેટવર્ક ઘણા વિવિધ પ્રકારના હોય છે. જો કે વપરાશકર્તાની દ્રષ્ટિએ બે મુખ્ય પ્રકાર હોય છે.

- **લોકલ એરિયા નેટવર્ક્સ (લેન)**

  જે કોમ્પ્યુટર ભૌગોલિક રીતે નજીક હોય. (એક જ બિલ્ડીંગમાં).

- **વાઈડ એરિયા નેટવર્ક (વેન)**

આ કોમ્પ્યુટર દૂર હોય છે અને ટેલિફોન લાઈનો અથવા રેડિયો તરંગો દ્વારા જોડાયેલા હોય છે.

આ સાથે વધુમાં, નીચેની લાક્ષણિકતાઓ પણ વિવિધ પ્રકારના નેટવર્કને વર્ગીકૃત કરવા માટે વપરાય છે.

- **ટોપોલોજી**

  કોમ્પ્યુટર સિસ્ટમની ભૌમિતિક વ્યવસ્થા. સામાન્ય ટોપોલોજીસમાં બસ, સ્ટાર અને રીંગનો સમાવેશ થાય છે.

- **પ્રોટોકોલ**

  પ્રોટોકોલ નિયમનો એ નેટવર્ક પર કોમ્પ્યુટર્સ સંપર્ક વ્યવહાર કરવા કે સંકેતો સામાન્ય સમુહ વ્યાખ્યાયિત કરે છે. લેન માટે સૌથી વધુ લોકપ્રિય પ્રોટોકોલ છે જે ઈથરનેટ. બીજા જાણીતો PC માટેનો પ્રોટોકોલ આઈ.બી.એમ ટોકન રીંગ નેટવર્ક છે.

- **આર્કિટેકચર**

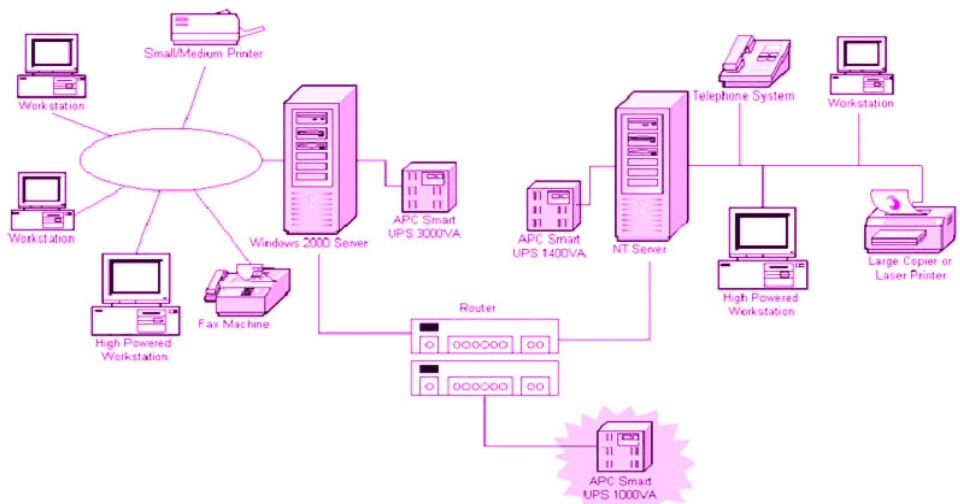
  નેટવર્કનો વ્યાપક રીતે ઉપયોગ કરીને પીઅર ટુ પીઅર અથવા કલાઈન્ટ/સર્વર આર્કિટેકચર તરીકે વર્ગીકૃત કરી શકાય છે. નેટવર્ક ઉપરના કોમ્પ્યુટરોને નોડ કહેવામાં આવે છે. નેટવર્ક માટે જે સંશાધનો ફાળવતા કોમ્પ્યુટર્સ અને ઉપકરણોને સર્વર કહેવામાં આવે છે.

- **લોકલ એરિયા નેટવર્ક (લેન)**

  (LAN) લેન પ્રમાણમાં નાના વિસ્તારમાં છવાયેલા કમ્પ્યુટર્સનું નેટવર્ક છે.મોટા ભાગના લેન ઈમારતો એક જ બિલ્ડિંગમાં અથવા બિલ્ડીંગના જૂથ માટે મર્યાદીત કરવામાં આવે છે. જો કે, એક લેન ટેલિફોન લાઈનો અને રેડિયો તરંગો દ્વારા કોઈ પણ અંતર પર બીજી લેન સાથે જોડાયેલ હોઈ શકે છે. આ રીતે જોડાયેલ લેન સીસ્ટમને વાઈડ એરિયા નેટવર્ક (વેન) કહેવામાં આવે છે.

આકૃતિ. 5.4. માં બતાવ્યા પ્રમાણે મોટા ભાગના લેન લેયર વર્કસ્ટેશનો અને વ્યકિગત કોમ્પ્યુટરો

સાથે જોડાયેલા હોય છે. એક લેનમાં દરેક નોડ (વ્યક્તિગત કોમ્પ્યુટર) તે એપ્લીકેશન ચલાવે છે કે જેની સાથે તેના પોતાના સી.પી.યુ હોય પરંતુ તે પણ લેન પર ગમે ત્યાં ડેટા અને ઉપકરણો વાપરવા માટે સમર્થ છે. આનો મતલબ એ થાય કે ઘણા વપરાશકર્તાઓ મોંધા ઉપકરણો શેર કરી શકે છે. જેમ કે લેસર પ્રિન્ટરો, સાથે સાથે માહિતી પણ વપરાશકર્તાઓ પણ એકબીજા સાથે વાતચીત કરવા માટે લેનનો ઉપયોગ કરી શકે છે, ઈમેઈલ મોકલીને અથવા ચેટ સેશન દ્વારા. લેન - ટોકન - રીંગ નેટવર્કસ ઘણા વિવિધ પ્રકારના હોય છે. ઈથરનેટ અને ARCnets પી.સી માટે સૌથી સામાન્ય હોય તેવા નેટવર્ક છે.



આકૃતિ. 5.4 લોકલ એરિયા નેટવર્ક

લેન ખૂબ ઝડપી દરે માહિતી પરિવહન માટે સક્ષમ હોય છે, એક ટેલિફોન લાઈન કરતા પણ ડેટા ઝડપથી પ્રસારિત થાય છે, પરંતુ અંતર મર્યાદિત છે, અને એક જ લેન માં જોડાવા માટે કોમ્પ્યુટર્સની સંખ્યા પણ મર્યાદીત છે.

વાઈડ એરિયા નેટવર્ક (વેન)

વેન એ મોટા ભૌગોલિક વિસ્તારમાં છવાયેલો કોમ્પ્યુટર્સ નેટવર્ક છે. લાક્ષણિક રીતે વેન એ બે કે તેથી વધુ લેનનો સમાવેશ છે. વેન નેટવર્ક સાથે જોડાયેલા કોમ્પ્યુટર્સ ઘણી વખત જાહેર નેટવર્ક દ્વારા જોડાયેલા હોય છે. જેમ કે ટેલિફોન સિસ્ટમ, તેઓ પણ ભાડે કે લીઝ પર આપવામાં આવતા ઉપગ્રહો દ્વારા જોડાયેલ હોય શકે છે. હાલમાં ઈન્ટરનેટએ સૌથી મોટું વેન છે.

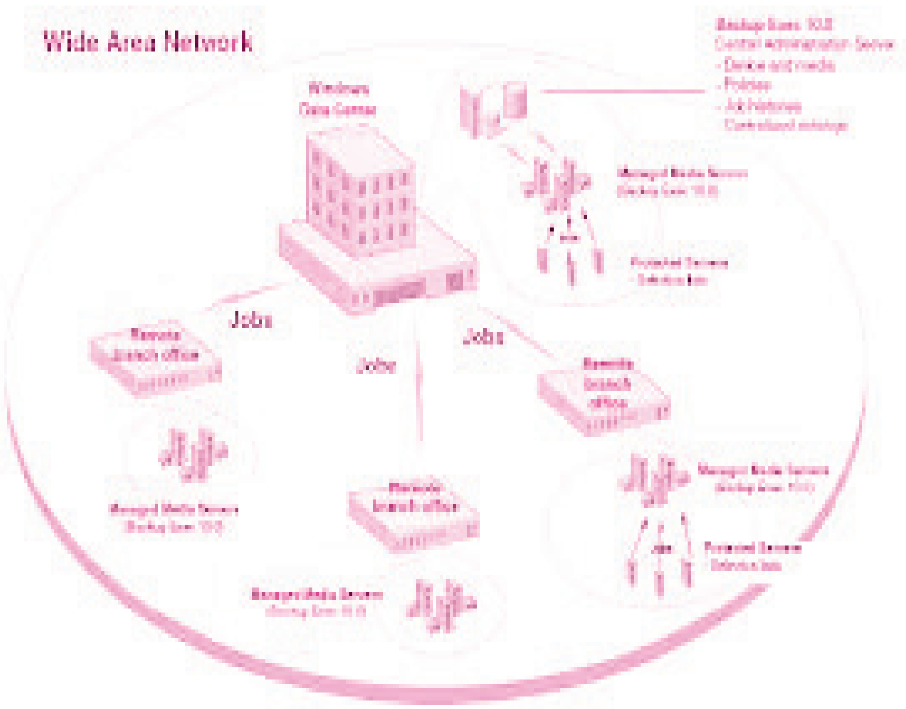
એક લાક્ષણિક વેન આકૃતિ. 5.5 માં બતાવ્યા પ્રમાણે છે.

નેટવર્ક ટોપોલોજી

આપણે આગળ જોયું તેમ ટોપોલોજી એ કોમ્પ્યુટર્સની નેટવર્કમાં ભૌમિતિક વ્યવસ્થા છે. સામાન્ય ટોપોલોજીમાં સ્ટાર, રિંગ અને બસ નો સમાવેશ થાય છે.

સ્ટાર નેટવર્ક : આકૃતિ. 5.6. માં દર્શાવ્યા પ્રમાણે સ્ટાર નેટવર્ક વાંરવાર જરૂરી સંખ્યામાં નાના

કોમ્પ્યુટર્સ અથવા સી.પી.યુ અથવા પેરિફરલ ઉપકરણો સાથે જોડાવા માટે વપરાય છે. ઘણા ઓર્ગેનાઈઝેશમાં ટાઈમ શેરીંગ સિસ્ટમમાં સ્ટાર નેટવર્કનો ઉપયોગ કરે છે, જેમાં કેટલાક વપરાશકર્તાઓ કેન્દ્રિય પ્રોસેસર શેર કરવા માટે સક્ષમ છે.



આકૃતિ. 5.5 વાઈડ એરિયા નેટવર્ક

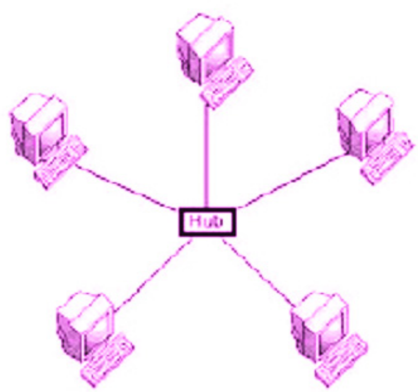
એક ટાઈમ શેરીંગ સંયોજનમાં, દરેક ટર્મિનલને C.P.U નો એક સરખો ટાઈમ મળે છે. જેને ટાઈમ સ્લાઈસ કહેવાય છે. જો તમે ટર્મિનલ પર બેઠા છો અને ટાઈમ સ્લાઈસ દરમિયાન તમારા કાર્યો પૂર્ણ કરી શકતા નથી. તો કોમ્પ્યુટર્સ તેમને એ કામ પુરું કરવા માટે ફરી પરવાનગી આપશે. વાસ્તવમાં, C.P.U એ ટર્મિનલ કરતાં ખૂબ જ ઝડપમાં ચાલે છે.જેથી તમે કદાચ C.P.U દૂર હોય એ નોટીસ પણ ના કરી શકો. સમય વહેંચણીની આવી સ્થાપનાના કરીને, મોટી સંસ્થામાં ઘણા લોકો કેન્દ્રીય કોમ્પ્યુટીંગ સુવિધા વાપરી શકે છે, જે એક નાની કંપની માટે કામ કરવા આર્થિક રીતે સરળ માર્ગ છે, કેમ કે તે વધારે કોમ્પ્યુટર વસાવી શકે એમ નથી. સ્ટાર નેટવર્ક વાંરવાર અનેક માઈક્રો કોમ્પ્યુટરો સાથે જોડાવા માટે લેન નો ઉપયોગ કરે છે. જે એક કેન્દ્રિય એકમ અને સંચાર નિયંત્રક તરીકે કામ કરે છે. એક માઈક્રોકોમ્પ્યુટરના વપરાશકર્તા બીજા વપરાશકર્તા માટે દસ્તાવેજ અથવા સંદેશ મોકલવા માંગે તો, સંદેશ કેન્દ્રિય સંચાર નિયંત્રક મારફતે જાય છે. સ્ટાર નેટવર્કનો બીજો સામાન્ય ઉપયોગ એ છે કે તે સંસ્થાના માઈક્રો કોમ્પ્યુટર્સને જોઈને ફ્રેમ સાથે જોડી ડેટાબેઝ વાપરવાની પરવાનગી આપે છે.

પ્રવેશ અને સ્ટાર નેટવર્ક પર નિયંત્રણ સામાન્ય રીતે એક મશીન સિસ્ટમ દ્વારા જાળવવામાં આવે

છે. પોલીંગનો અર્થ એ છે કે કેન્દ્રિય કોમ્પ્યુટર, અથવા સંચાર નિયંત્રક "પોલ" અથવા નેટવર્કમાં દરેક ઉપકરણને પૂછે છે, કે જો મોકલવા માટે સંદેશ હોય તો પછી માહિતી વહન કરવા વારાફરતી પરવાનગી આપે.

રીંગ નેટવર્ક

રીંગ નેટવર્ક (જુઓ આકૃતિ. 5.7) જેની ટોપોલોજી એક લેન છે. રીંગ નેટવર્ક એ કોમ્પ્યુટરને અલગ - અલગ સ્થળોએ એક જગ્યાએથી બીજી જગ્યાએ જોડાય તો સરળ પડ છે, તે પણ કોઈ કેન્દ્રિય યજમાન કોમ્પ્યુટર અથવા સંચાર નિયંત્રક વગર. તે તમામ એક બંધ લૂપ(રીંગ) માં જોડાયેલા છે. સંદેશાઓ રીંગની આસપાસ વહન થાય છે. દરેક નોડ તેને મળતા સંદેશાઓને વાંચે છે, જે તેમના માટે લખાયેલ હોય. રીંગ નેટવર્કનો એક લાભ એ છે કે તેનું નેટવર્ક અન્ય પ્રકારના નેટવર્ક કરતા વધારે અંતર સુધી હોઈ શકે છે. જેમ કે બસ નેટવર્ક, કારણે કે દરેક નોડ સંદેશને પુનર્જીવન આપીને પસાર થયા છે.



આકૃતિ. 5.6 સ્ટાર ટોપોલોજી

આકૃતિ. 5.7 રીંગ ટોપોલોજી

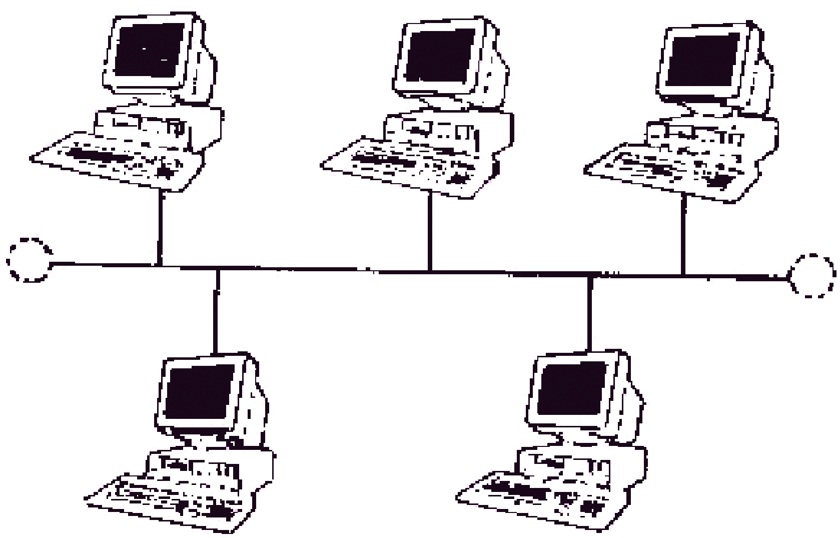
પ્રવેશ અને રીંગ નેટવર્ક પર નિયંત્રણ ખાસ કરીને "ટોકન પસાર" સીસ્ટમ દ્વારા જાળવવામાં આવે છે. IBM ટોકન રીંગને પોતે વિકાસ માટેના તુલનાત્મક એક વોટરશેડ સાથે હોવાનું કેટલાક નિરીક્ષકો દ્વારા માનવામાં આવે છે, કારણ કે ટોકન - રીંગ નેટવર્કની રચના એક સાથે તમામ પ્રકારના કોમ્પ્યુટર્સને લીંક કરવા થઈ છે. જેમાં પર્સનલ કોમ્પ્યુટર અને મેઈન ફ્રેમનો પણ સમાવેશ થાય છે.

રીંગ 5.7 માં બતાવ્યા પ્રમાણે ટોકન રીંગ Marry-go-round (ચગડોળ) જેવું લાગે છે. એક સંદેશ પહોંચાડવા માટે બે તમે Marry go round પર સવાર છો (ટોકન) તો તમારી નોંધને ધ્યાનમાં લેવાશે અને તેને યોગ્ય સ્થળે બંધ કરવામાં કે મૂકવામાં આવ્યો.

બસ નેટવર્ક

બસ નેટવર્ક (આકૃતિ. 5.8 પ્રમાણે) રીંગ નેટવર્ક જેવું જ છે પણ તેના છેડા જોડાયેલા હોતા નથી. બધા સંચાર એક સામાન્ય કેબલ પર કરવામાં આવે છે, અથવા બસ, અને નેટવર્ક પરના દરેક ઉપકરણ પર તે ઉપલબ્ધ હોય છે.

પ્રવેશ અને બસ નેટવર્ક પર અંકુશ સામાન્ય રીતે એક કંટેન્શન તરીકે ઓળખાતી પધ્ધતિથી કરવામાં આવે છે. જો લાઈન ફ્રી હોય તો, ટર્મિનલ અથવા ઉપકરણ, તેનો સંદેશ પ્રસારિત કરી શકે છે., પરંતુ બે કે તેથી વધુ ટર્મિનલ એક સાથે સંદેશાઓ શરૂ કરે તો, તેઓને બંધ કરીને વિવિધ સમયાંતરે ફરી વહન કરવું જ જોઈએ.



આકૃતિ. 5.8 બસ નેટવર્ક

5.10 નેટવર્કના પ્રોટોકોલ

પ્રોટોકોલ બંધારણમાં બે ઉપકરણો વચ્ચે માહિતીના પરિવહન માટે સંમત થાય છે.

પ્રોટોકોલ નીચે પ્રમાણે નક્કી કરે છે. :

- *ભૂલ ચકાસણી કરવા માટેના પ્રકાર.*
- *ડેટા સંકોચન પધ્ધતિ, જો કોઈ હોય તો*
- *સંદેશ મોકલવા માટેનું ઉપકરણ થકી સંદેશ મોકલાયો છે કે કેમ તે જણાવવાની રીત*
- *પ્રાપ્ત ઉપકરણ થકી સંદેશ પ્રાપ્ત થયો છે કે કેમ તે જાણવાની રીત.*

એવા ઘણા પ્રોટોકોલ છે જેને પ્રોગ્રામરો પસંદ કરી શકે છે. દરેકના પોતાના વ્યક્તિગત ફાયદા અને ગેરફાયદા છે દાખલા તરીકે, અમુક અન્ય કરતાં સરળ છે, કેટલાક વધુ વિશ્વસનીય છે અને કેટલાક વધુ ઝડપી છે.

એક વપરાશકર્તાની દૃષ્ટિએ, પ્રોટોકોલ વિશે એક માત્ર રસપ્રદ પાસું છે જે તમે અન્ય કોમ્પ્યુટર્સ સાથે વાતચીત કરવા માંગતા હોય તો તમારા કોમ્પ્યુટર્સ અથવા ઉપકરણો એ યોગ્ય મુદ્દાઓ ઉપર જ આધાર રાખવો જોઈએ. પ્રોટોકોલ હાર્ડવેર અથવા સોફટવેરમાં અમલ કરી શકાય છે. કેટલાક લોકપ્રિય પ્રોટોકોલ જેવા કે TCP/IP, HTTP, FTP, SMTP, POP, Token - ring, Ethernet, Xmodem, Kermit,MNP વગેરે છે.

5.11 નેટવર્ક આર્કિટેકચર

આર્કિટેકચર શબ્દ માટે હાર્ડવેર અથવા સોફટવેરનો ઉલ્લેખ કરી શકાય છે, અથવા હાર્ડવેર અને સોફટવેરનું મિશ્રણથી પણ એક સિસ્ટમનું આર્કિટેકચર હંમેશા તેની વ્યાપક રૂપરેખાથી વ્યાખ્યાયિત થાય છે. અને સાથે સાથે ચોક્કસ પધ્ધતિમાં પણ વ્યાખ્યાયિત કરે છે.

એક ઓપન આર્કિટેકચર સિસ્ટમ અન્ય ઉત્પાદકોના ઉપકરણો અને કાર્યક્રમો સાથે સરળતાથી જોડાઈ શકે છે. ઓપન આર્કિટેકચર તૈયાર ઘટકોનો ઉપયોગ કરી અનુક્રમે ધોરણોની મંજૂરી આપે છે. એક બંધ આર્કિટેકચર સિસ્ટમ સાથે, તો બીજી બાજુ, એક ડિઝાઈન માલિકીની છે, તે બીજી સિસ્ટમોને પોતાની સિસ્ટમો સાથે જોડાવનું મુશ્કેલ બનાવે છે, આપણે આગળ જોયું તેમ, નેટવર્ક આર્કિટેકચર નો ઉપયોગ ખાસ કરીને કયાં પીઅર - ટુ - પીઅર અથવા કલાઈન્ટ/સર્વર તરીકે વર્ગીકૃત કરી શકાય છે.

પીઅર - ટૂ - પીઅર આર્કિટેકચર

આ નેટવર્કનો એક એવો પ્રકાર છે જેનાં દરેક વર્કસ્ટેશનની ક્ષમતાઓ અને જવાબદારીઓ સમકક્ષ છે, જે ક્લાઈન્ટ/સર્વર આર્કિટેક્ચર થી અલગ પડે છે, જેમાં વર્કસ્ટેશનો અન્ય સેવા આપવા માટે સમર્પિત હોય છે. પીઅર - ટૂ પીઅર નેટવર્ક સામાન્ય રીતે સરળ અને ઓછા ખર્ચાળ છે. પરંતુ તેઓ સામાન્ય રીતે ભારે લોડ (કામ) હેઠળ એક જ પ્રકારની કામગીરી કરતા નથી.

ક્લાઈન્ટ/સર્વર આર્કિટેકચર

આ નેટવર્ક આર્કિટેક્ચરમાં નેટવર્કમાં દરેક કોમ્પ્યુટર અથવા પ્રક્રિયાએ ક્લાઈન્ટ અથવા સર્વર હોય શકે છે. સર્વરની ડિસ્ક ડ્રાઈવોની વ્યવસ્થા માટે સમર્પિત શક્તિશાળી કોમ્પ્યુટર્સ અથવા પ્રોસેસરો હોય છે. આ જ પ્રમાણે આવી જ વ્યવસ્થા પ્રિન્ટરો અને નેટવર્ક ટ્રાફિક માટે પણ હોય છે. ક્લાઈન્ટ વર્કસ્ટેશનના ઓછી શક્તિવાળા PC વર્કસ્ટેશનો જેમાં વપરાશકર્તા તેની એપ્લીકેશનને રન કરે છે. ક્લાઈન્ટો સ્ત્રોત માટે સર્વરો પર આધાર રાખે છે, જેમ કે ફાઈલો, ઉપકરણો અને પ્રક્રિયા શક્તિ માટે પણ.

પાઠગત પ્રશ્નો 5.2

---

1. LAN અને WAN વચ્ચેનો તફાવત જણાવો.

૨. નેટવર્ક ટોપોલોજી કેટલા પ્રકારની હોય છે ?

3. **નીચેના નિવેદનો માટે સાચુ (T) અથવા ખોટું (F) લખો :**

(અ) *નેટવર્ક એક જ રૂમમાં કોમ્પ્યુટરોને ઈન્ટરકોનેક્ટ કરવા માટે વપરાય, અને એક મકાન કે એક સાઈટ પર ઈમારતોની અંદરના નેટવર્કને WAN કહેવામાં આવે છે.*

(બ) *WAN પ્રાદેશિક, રાષ્ટ્રીય અથવા વૈશ્વિક વિસ્તારમાં ફેલાયેલું નેટવર્ક છે.*

(ક) *માહિતી પ્રસાશનની ઝડપ LAN કરતાં WAN માં વધારે હોય છે.*

5.2 નેટવર્કીંગના ઉપયોગમાં વપરાતી મહત્વની બાબતો.

(અ) ઈન્ટરનેટ

સંસ્થાની અંદર વાપરવા માટેના પ્રોગ્રામમાં સૌથી નવું એ ઈન્ટરનેટ કે ઈન્ટરનેટ વેબ છે. આવા નેટવર્ક સરળતાથી વાતચતી કરવા માટે કોઈપણ પ્રકારના કોમ્પ્યુટર્સને સક્રિય કરે છે. હાર્ડવેર અને સોફટવેરની જરૂરિયાતો માટે ઈન્ટરનેટ એક કારણ છે. ખાસ કરીને TCP/IP,સર્વર અને બ્રાઉઝર સોફટવેર WWW માટે છે કારણ કે ઘણી બધી સંસ્થાઓ લોકો અને માહિતીને લીંક કરવા માટે વધુ ગતિશીલ છે. ઈન્ટરનેટ માર્કેટ દિવસે - દિવસે વિસ્તરી રહ્યું છે. આ ઉપરાંત, જો કોઈ નવો વપરાશકર્તા ઉમેરાય ત્યારે નેટવર્કને વ્યવસ્થિત કરવાની જરૂર નથી. ઈન્ટરનેટની મદદથી, સંસ્થામાં તેમાંના કોમ્પ્યુટર એકલા સિસ્ટમો તરીકે કામ કરી શકે છે, જે મેઈનફ્રેમમાં જોડાયેલા અથવા LAN અને WAN ના ભાગરૂપે હોય છે.

(બ) ઈમેઈલ

ઈ મેઈલ એટલે ઈલેક્ટ્રોનિક મેલ. આ ઈન્ટરનેટમાં સૌથી વધુ ઉપયોગમાં લેવાતું લક્ષણ છે. મેઈલનો આજે નિયમિત રીતે ઉપયોગ થાય છે. કારણ કે પોસ્ટેજ સ્ટેમ્પની મદદ વગર આપણે તેને વિશ્વમાં ગમે ત્યાં મોકલી શકીએ છીએ. ઈ - મેઈલ ઈલેક્ટ્રોનીક મેલ સર્વિસ જેવી જ છે. પરંતુ અહીં માહિતી ઈન્ટરનેટ દ્વારા ફેલાય છે અને તેથી એક મિનિટની અંદર અંતિમ મુકામે અથવા વિશ્વમાં ગમે ત્યાં પહોંચી શકે છે. તેથી ઈમેલ મારફતે મેઈલિંગ સિસ્ટમ વધુ ઝડપી બની છે અને મેલ ટ્રાન્સફર માટે ઈમેઈલનો વ્યાપક ઉપયોગ કરવામાં આવી રહ્યો છે.

(ક) વોઈસ મેસેજિંગ

આ એક ઉંચો સંચાર અભિગમ છે અને તે ઈલેક્ટ્રોનિક મેલ જેવી જ છે. સિવાય કે તે એક ઓડિયો સંદેશ પ્રક્રિયા છે. મોકલનાર લખવાને બદલે ટેલિફોનમાં બોલે છે. પ્રાપ્તકર્તાના નામ અને સંદેશ આપવામાં આવે છે અને મોકલનારનો અવાજ પછી ડિજિટલાઈઝડ અને સંગ્રહિત કરવામાં આવે છે. સિસ્ટમ પછી ભવિષ્યમાં ચોક્કસ સમયે સંદેશ વિવરિત કરી શકે છે. અથવા પ્રાપ્તકર્તા તેને સાંભળી શકે, જેથી તે વિવરિત અથવા મેળવાય છે, જે ટેલિફોન પર મોકલનાર મૂળ વ્યકિતનો જ અવાજ હોય છે. વોઈસ સંદેશા માટે એ જરૂરી છે કે, તેને ડિજિટલ સ્વરૂપમાં સંગ્રહી શકે તેવું કોમ્પ્યુટર તેમજ ચકાસણી કરવા માટે ઓડિયો સ્વરૂપે પણ કોમ્પ્યુટરની જરૂર

પડે છે. દરેક વપરાશકર્તા ગૌણ સંગ્રહ તરીકે એક અવાજ મેઈલબોક્ષ ધરાવે છે અને ખાસ સાધનો જે સંદેશાને મેલ પર વોઈસમાં બદલે છે. મેઈલનો મુખ્ય લાભ એ છે કે, મોકલનારને લખવું પડતું નથી. વોઈસ મેઈલ કંપનીના પર્યાવરણમાં લોકોનો સમાવેશ કરવાનું સરળ બનાવે છે.

નોંધ

ડ. ઈ- કોમર્સ

ઈલેકટ્રોનિક કોર્મસ અથવા ઈ - વાણિજિયિક તરીકે લોકપ્રિય છે. તે કાગળ વગર બિઝનેશમાં ઈલેકટ્રોનિક ડેટાનું આદાનપ્રદાન કરવા ઈલેકટ્રોનિક મેલ, ઈલેકટ્રોનિક બુલેટિન બોર્ડ, ઈલેકટ્રોનિક ફંડ ટ્રાન્સફર અને નેટવર્ક આધારિત ટેકનોલોજી વાપરે છે. ઈલેકટ્રોનિકસમાં કોમર્સ EC માત્ર જો પ્રક્રિયા અને કાગળ વ્યવહારો જ આપમેળ નથી કરતી પરંતુ તે સંસ્થાને સંપૂર્ણપણે ઈલેકટ્રોનિક પર્યાવરણમાં ખસેડવામાં મદદ કરે છે અને તેને સામાન્ય રીતે ઓપરેટ પણ કરે છે. થોડી સંસ્થાઓએ તાજેતરમાં ઈન્ટરનેટ પર EC કરવા શરૂ કર્યું છે અને વૈકલ્પિક રીતે ઓછી કિંમત વાળું નેટવર્ક હોય છે. EC ધોરણો જો કે વિકાસ હેઠળ છે. ઈલેકટ્રોનિક ડેટા ઈન્ટરચેન્જ (EDI) હજુ EC ના પ્રભુત્વ હેઠળ છે.

માહિતી ટેકનોલોજીથી લોકોના કામ કરવાની રીતમાં સફળ થઈ શકે છે. ઈલેકટ્રોનીકસ કોમર્સ (EC) ક્રાંતિ સમાન છે. જે બિઝનેશ ગૃહો ઉત્પાદનો અને સેવાઓ ખરીદવા અને વેચવા ઉપયોગ કરે છે. **EC** નો સંબંધ કોમ્પ્યુટર સંચાર નેટવર્કસ પર ઉત્પાદનો અને સેવાઓ ખરીદી અને વેચાણ સાથે છે. EC સ્વાયત રીત કોમ્પ્યુટર માટે કોમ્પ્યુટરથી ઈલેકટ્રોનિક માહિતી પરિવહન કરે છે. EC હકીકતમાં જે રીતે વપરાય છે તે સંસ્થાઓની કાયાપલટ કરી રહી છે.

(ઈ) ઈલકેટ્રોનિક ડેટા ઈન્ટરચેન્જ (EDI)

EDI પ્રમાણભૂત બંધારણમાં બિઝિનેશ દસ્તાવેજોના કોમ્પ્યુટરથી - થી કોમ્પ્યુટરમાં એકસચેન્જ છે. આ બંધારણો ઘણી રીતે પ્રમાણભૂત સ્વરૂપોની જેમ જ દેખાય છે અને અત્યંત વ્યવસ્થિત ગોઠવણ વાળા હોય છે.

(ઈ) ટેલીકોન્ફરન્સીંગ

તે ઈલેકટ્રોનિક બેઠકનો ઉલ્લેખ કરે છે, તેમાં Physicially અલગ જગ્યા પર હોય તેવા લોકોનો સમાવેશ થાય છે. દૂરસંચાર સહભાગીઓ એક બીજા સાથે વાતચીત કરવા માટે પરવાનગી આપે છે અને તે પણ એક જગ્યાએથી બીજી જગ્યાએ ગયા વગર.

5.13 તમે જે શીખ્યા

આ પાઠ તમને કોમ્પ્યુટર દ્વારા સંદેશા વ્યવહારનું મહત્વ અને સ્થિતિઓ વિશે શીખવે છે. કોમ્પ્યુટર્સ એક બીજા સાથે કોમ્પ્યુટર નેટવર્કની મદદથી વાત કરી શકે છે. કોમ્પ્યુટર નેટવર્કના બે પ્રકાર હોય છે : LAN અને WAN. આપણે નેટવર્ક ટોપોલોજીમાં ભૌતિક કોમ્પ્યુટરની વ્યવસ્થા અને ભાગો વિશે ચર્ચા કરી. નેટવર્કના ટોપોલોજીના ત્રણ પ્રકાર હોય છે : સ્ટાર, બસ અને રીંગ ટોપોલોજી. અંતે આપણે ઈન્ટરનેટ અને ઈ મેઈલ વિશે પણ ચર્ચા કરી છે.

5.14 પાઠયાંત પ્રશ્નો

1. *કોમ્પ્યુટર નેટવર્ક શું છે ? તેના મુખ્ય હેતુઓ શું છે ?*
2. *એનાલોગ અને ડિજિટલ માહિતી પ્રસારણ વચ્ચે શું તફાવત છે ?*
3. *સંચારના અલગ - અલગ પ્રકારના મિડિયા સમજાવો.*
4. *સિમ્પ્લેકસ અને સંપૂર્ણ દ્વિગુણિત ટ્રાન્સમિશન વચ્ચે શું તફાવત છે ?*
5. *નેટવર્ક ઓર્કિટેકચર ટૂંકમાં સમજાવો.*
6. Teleconferencing *ને વ્યાખ્યાયિત કરો.*

5.15 પાઠગત પ્રશ્નોના જવાબ

પાઠગત પ્રશ્નો 5.1

1. *ડેટા સંચાર સોફટવેર કોમ્પ્યુટર સિસ્ટમો અને ઉપકરણો કેવી રીતે માહિતી એક અને બીજા સ્થળ પરથી સ્થાનાંતરિત કરી તેની સૂચનાઓ આપે છે. સોફટવેર સ્વરૂપમાં માહિતી રૂપાંતર પ્રક્રિયા સામાન્ય રીતે પ્રોટોકોલ તરીકે ઓળખાય છે.*
2. *અસુમેળ ટ્રાન્સમિશન માહિતીમાં અક્ષર જ ફેલાય છે, જેને જેમ તમે કી બોર્ડ ઉપર ટાઈપ કરતાં જશો. સુમેળ સ્થિતિમાં આ સાચવેલી માહિતી બ્લોક દ્વારા મોકલાય છે ?*
3. *(અ) સાચું (બ) ખોટુ (ક) સાચું (ડ) સાચું (ઈ) સાચુ*

પાઠગત પ્રશ્નો 5.2

1. LAN *એ મર્યાદિત ભૌગોલિક વિસ્તાર માટે પ્રતિબંધિત છે. જયારે* WAN *લાંબા અંતરને આવરી લે છે.* LAN *માં કોમ્પ્યુટર ટર્મિનલસ અને પેરિફરલસ ઉપકરણ વાયર અને કેબલ સાથે જોડાયેલા હોય છે. જયારે* WAN *માં વાતચીત કરવા માટે ટેલિફોન લાઈન અને સેટેલાઈટ લીંકનો ઉપયોગ થાય છે. માહિતી પ્રસારણની ઝડપ* WAN *કરતાં* LAN *માં વધારે હોય છે.*
2. *નેટવર્ક ટોપોલોજીના ત્રણ મુખ્ય પ્રકારો છે. તેઓ સ્ટાર ટોપોલોજી, બસ ટોપોલોજી અને રિંગ ટોપોલોજી છે.*
3. *(અ) ખોટુ (બ) સાચું (ક) ખોટુ*

6

ઈન્ટરનેટ અને જાવા પ્રોગ્રામિંગના મૂળભૂત તત્વો

6.1 પ્રસ્તાવના

ઈન્ટરનેટ એ વિશ્વનું સૌથી મોટું કોમ્પ્યુટર નેટવર્ક છે. નેટવર્કોનું નેટવર્ક જે આખા વિશ્વમાં ફેલાયેલું છે. એ ૨૫ વર્ષ પહેલાં યુ.એસ.ના ડિપાર્ટમેન્ટ ઓફ ડિફેન્સના પ્રોજેક્ટ તરીકે રચાયેલું છે. આનો ઉદ્દેશ એવી રીત વિકસાવવાનો હતો જે વિશાળ વિસ્તારમાં ફેલાયેલા કોમ્પ્યુટરના વચ્ચે માહિતીનું આદાન-પ્રદાન કરી શકે, ન્યૂકિલયર હુમલાના સમયે પણ ઈ.સ.1960 માં નાની સંખ્યામાં વપરાશકર્તા અને કોમ્પ્યુટરોમાંથી અત્યારે ઈન્ટરનેટ, હજારો પ્રાદેશિક નેટવર્કોમાં વધ્યું છે જે લાખો વપરાશકર્તાઓને જોડે છે. ઈન્ટરનેટનો ઉપયોગ કરીને સંગઠન/કંપની આખા વિશ્વમાં માહિતીની આપલે, માણસો સંદેશાની આપલે ઝડપી અને અસરકારક રીતે કરી શકે છે. વિડીયો કોન્ફરસીંગ (video-conferencing) ની મદદથી હવે એ શકય બન્યુ છે કે વાતચીતના સમયે માણસો એકબીજાને જોઈ પણ શકે છે. એક માણસ એની બધી ખરીદી પણ ઘેર બેઠા કરી શકે છે. એને એનો મૂલ્યવાન સમય બગાડવાની કે વસ્તીવાળા માર્કેટમાં જવાની પણ ચિંતા રહેતી નથી. ધીમેધીમે દુકાનદારો પણ ઈ-કોમર્સ (E-Commerce) માં રસ લેવા લાગ્યા છે જે તેમને ઝડપી અને તેમનો ધંધો ઈન્ટરનેટ પર કરવા માટે વધુ સારા રસ્તા આપે છે. જો કાલે તમને ખબર પડે કે તમારા વિસ્તારનો પાનવાળો ઈન્ટરનેટ પર પાન વેચે છે તો આશ્ચર્યચકિતના થઈ જતા.

પરિણામે, ઈન્ટરનેટ એ ઘણાં ટેકનોલોજીકલ પડકારો આપ્યા છે જેવાં કે ઈન્ટરેનટનો ઉપયોગ ઝડપી, ખાનગી અને સુરક્ષિત વાતચીત અને વિડીયો અને ત્રિપરિમાણીય (3 D) આકૃતિઓને આધાર આપવો. ઈન્ટરનેટની આ રૂપરેખા મુજબ અને, લેખકો એપલેટને વેબદસ્તાવેજમાં સમાવિષ્ટ કરી શકે તેના માટે નિર્વિવાદ પ્રોગામીંગ ભાષાની જરૂર છે. છેલ્લે આવી Java પ્રોગામીંગ

ભાષાનો જન્મ થયો. આપણે આ ભાષા વિષે આ પાઠના બીજા ભાગમાં વિગતવાર ચર્ચા કરીશું.

6.2. હેતુઓ

આ પાઠના ભણ્યા પછી તમે નીચેની મુદ્દાઓ માટે સક્ષમ હશો :

- ઈન્ટરનેટમાં વપરાતી વિવિધ પરિભાષાને સમજાવી શકશો.
- ઈન્ટરનેટની વિવિધ સેવાઓનો ઉપયોગ કરી શકશો.
- ઈન્ટરનેટ પરથી જોઈતી માહિતી શોધી શકશો.
- ઈન્ટરનેટ વાપરવાનો આનંદ માણી શકશો.
- ઈમેલ અને તેની વિવિધ લાક્ષણિકતાઓને વ્યાખ્યાયિત કરી શકશો.
- www ની લાક્ષણિકતાઓ સમજાવી શકશો.
- Java ભાષાની ઉપયોગિતા અને તેની લાક્ષણિકતાઓ સમજાવી શકશો.
- Java અને ઈન્ટરનેટ વચ્ચેનો સંબંધ જાણી શકશો.

6.3 ઈન્ટરનેટ - ઈતિહાસ (Internet-The History)

ઈ.સ.1969 માં યુ.એસ.ડિફેન્સ ડિપાર્ટમેન્ટએ બોમ્બમારામાં પણ વાપરી શકાય છે તેવું નેટવર્ક બનાવવા ફંડ આપ્યું હતું. મૂળભૂત ઉદ્દેશ્ય એવું નેટવર્ક બનાવવાનો હતો કે જે ન્યુકિલયર હુમલા પછી પણ વાપરી શકાય. આ પ્રોજેક્ટ ARPANET તરીકે ઓળખાય છે. વિચારેલ નેટવર્કમાં ધારેલ કે તેને મધ્ય નિયંત્રક ના હોય કે જે દેખીતુ લક્ષ્ય હોય. દસ વર્ષનું સંશોધન LAN (Local Area Networcs) લાવ્યું અને LAN સાથે જોડાવવા મશીનો (work stations) વિકસાવવામાં આવ્યા. આ મશીનો અને LAN પછી ARPANET સાથે જોડવામાં આવ્યા. પછીના દસકામાં ARPANET વિકસ્યુ અને તેની વિકેન્દ્રીકરણની લાક્ષણિકતા એ તેને વિકસાવવામાં મદદ કરી. ARPANET સાથે જોડાયેલ કોમ્પ્યુટરોએ આદર્શ અથવા નિયમ એકબીજા સાથેના વ્યવહારમાં ઉપયોગ કર્યો. આ આદર્શ જે ARPANET એ ઉપયોગ કર્યો તે NCP (National Control Protocol) તરીકે ઓળખાય છે. પ્રોટોકોલએ નેટવર્કની પરિભાષા છે જે નેટવર્ક દ્વારા આપલે વખતે વપરાતા આદર્શને દર્શાવવા માટે વપરાય છે. પણ સમયની સાથે અને ટેકનોલોજીમાં ઘણા બદલાવો NCP ની જગ્યાએ TCP/IP લાવ્યા. TCP મેસેજને પેકેટમાં ફેરવી નાખે છે અને નિદિષ્ટ સ્થાન પર પેકેટને જોડીને મેસેજ (સંદેશ) બનાવે છે. ઈન્ટરનેટ પ્રોટોકોલ (IP) આ પેકેટને મોકલવાનું કાર્ય કરે છે.IP મશીનના સરનામાનો હવાલો સંભાળે છે અને પેકેટ એના સ્થાને પહોચે તેની ખરાઈ કરે છે. એના માટે પેકેટ અલગ-અલગ સાધનોમાંથી અને અલગ-અલગ નેટવર્ક જે અલગ અલગ પ્રોટોકોલ વાપરે છે તેમાંથી પસાર

થાય છે. TCP/IP ની આ એક કરતા વધારે પ્રોટોકોલને હેન્ડલ કરવાની ક્ષમતાને લીધે બીજા નેટવર્કને ARPANET સાથે જોડાવા માટે પીઠબળ પૂરૂ પાડયું. ધીમેધીમે ARPANET નેટવર્કનું નેટવર્ક બની ગયું અને હવે એ Internet તરીકે ઓળખાય છે.

નોંધ

ઈન્ટરનેટ, ઘણાં લોકો દ્વારા વિશ્વનું મોટામાં મોટું પ્રજાસત્તાક તંત્ર તરીકે પણ ઓળખાય છે એનો કોઈ વડો (હેડ) નથી

માણસોને ઈન્ટરનેટ સાથે કેમ જોડાવું હોય છે ? કદાચ એ જે સ્વતંત્રતા આપે છે તેના માટે ઈન્ટરનેટએ વિશ્વના સૌથી મોટા પ્રજાસત્તાક તંત્રનો અસામાન્ય દાખલો છે. જેના કોઈ વડા નથી, કોઈ બોસ (BOSS) કે બોર્ડ ઓફ ડિરેક્ટર નથી. ઈન્ટરનેટનું કોઈ નિયંત્રણ કરતું નથી અને કોઈપણ કોમ્પ્યુટર બીજા કોઈપણ કોમ્પ્યુટર સાથે વાતચીત કરી શકે છે જ્યાં સુધી એ TCP/IP ના નિયમો પાળે છે. ઈન્ટરનેટની આ સ્વતંત્રતાએ એને તેના મૂળભૂત પાયા જેવા કે મીલીટરી અને સંશોધન સંસ્થાઓમાંથી બહાર જેવા સ્કૂલ, જાહેર પુસ્તકાલય, કોલેજો અને શાકભાજીની દુકાન સુધી પ્રસરવામાં એની મદદ કરી છે.

6.4 ઈન્ટરનેટ, ઈ-મેઈલ, FTP, Telnet અને WWW ની સેવાઓ

ઈન્ટરનેટ સાથે એક વ્યકિત શું કરી શકે ? કદાચ ચાર વસ્તુઓ જેવી કે મેઈલ (Mail) , ચર્ચા કરી શકે, લાંબા અંતરના કોમ્પ્યુટરોમાં ગણતરી અને ફાઈલની ટ્રાન્સફર. ઈમેલ, સામાન્ય પોસ્ટ ટપાલ કરતા ઘણું ઝડપી છે. એમા જોડાણ તરીકે સોફટવેર અને અમુક પ્રકારની કોમ્પ્રેસ ડીઝીટલ ઈમેજ પણ મોકલી શકાય છે. ન્યૂઝગૃપ કે ડિસ્કશન ગૃપ, ઈન્ટરનેટ વપરાશકર્તાને અલગ અલગ પ્રકારની ચર્ચા અને નવી માહિતીના અદલા-બદલીમાં મદદ કરે છે. ARPANET ના વિકાસની પ્રેરણા લાંબા અંતરની કોમ્પ્યુટરને લગતી પ્રવૃત્તિઓ હતી અને હજુપણ એ ઘણી ઉપયોગી સેવા ઈન્ટરનેટ પર આપે છે. પ્રોગ્રામરો, દૂર અંતર આવેલ પાવરફુલ કોમ્પ્યુટરો જે પ્રોગ્રામનો અમલ કરે છે તેની પર એકાઉન્ટ જાળવી શકે છે. FTP સેવા, ઈન્ટરનેટ વપરાશકર્તાને દુર અંતરે આવેલ મશીન (કોમ્પ્યુટર) માંથી માહિતી અને પ્રોગ્રામને વાપરી શકે છે.

(અ) E - mail (Electronic Mail)

ઈમેઈલ એ પેપર વગરની સંદેશો મોકલવાની રીત છે. ઈમેઈલથી એક સમયે એક કે એક કરતા વધારે લોકોને સંદેશો મોકલી શકાય છે, ઈન્ટરનેટના માધ્યમથી ઈમેલ સેવા ખૂબ ઝડપી છે. ઈમેઈલ સંદેશા સામાન્ય રીતે થોડીક સેકન્ડોમાં એની જગ્યાએ પહોંચી જાય છે. કોઈપણ સમયે, રાત્રિના કે દિવસ દરમ્યાન તે મોકલી શકાય છે અને તરત તેની સોપણી પણ થઈ જાય છે. તમારે પોસ્ટ ઓફિસ ખૂલવાની કે રજાઓ વિશે ચિંતા કરવાની જરૂર નથી. તે ર4 કલાક અને અઠવાડિયાના સાતે દિવસ કાર્યરત રહે છે. બીજુ કે, તમે મોકલેલ સંદેશો તમે અડધી રાતે પણ જોઈ શકો છો.

તમને વધારાની વસ્તુ જેવી કે ફાઈલ ચિત્ર, ગ્રાફિક્સ મોકલવાની સવલત પણ છે. સૌથી મોટો ઈમેલ વાપરવાનો ફાયદો એ છે કે તે બહુ સસ્તુ, વિશેષમાં જયારે સંદેશો બીજા રાજય કે દેશમાં અથવા એક કરતા વધારે લોકોને એક જ સમયે મોકલવો હોય. ઈમેલ ઝડપી અને સસ્તુ છે તે છતાં પણ સામાન્ય ટપાલના ઘણા ભાગ એમાં છે તે તમને સંદેશો બનાવવામાં, લેનારનું સરનામું મેળવવામાં અને મોકલવા માટે મદદ કરે છે. એકવાર સંદેશો મળી જાય પછી તેને ફોરવર્ડ કરી શકાય છે કે તેનો જવાબ પણ આપી શકાય છે. કોઈ એને ભવિષ્યના ઉપયોગ માટે સંગ્રહ પણ કરી શકે છે કે એને કાઢી પણ શકે છે. ઈમેઈલમાં મોકલનાર, સંદેશો મળ્યાની રસીદ માટે આવેદન પણ કરી શકે છે અને તે વાંચી શકે છે.

1. ઈ-મેઈલની લાક્ષણિકતાઓ

- એકથી એક અથવા એકથી વધારે સાથે સંદેશા વ્યવહાર
- તાત્કાલિક સંદેશા વ્યવહાર
- લેનારની હાજરીની જરુર નથી
- સૌથી સસ્તી અને 24 કલાક અને અઠવાડિયાના સાતે દિવસ ઉપલબ્ધ
- તે અનૌપચારિક સંદેશાવ્યવહારને પ્રોત્સાહન આપે છે.

2. ઈમેઈલ એડ્રેસના ભાગો

સામાન્ય ટપાલની જેમ ઈમેઈલ સેવામાં પણ લેનારના સરનામાની જરૂર પડે છે. ઈમેઈલ એડ્રેસ, સંદેશાને લેનાર સુધી પહોંચાડવા માટે જરૂરી બધી માહિતી આપે છે. નીચેનું ઈમેઈલ એડ્રેસ ધ્યાનમાં લો.

john@hotmail.com

ઉપરના સરનામામાં John સ્થાનિક ભાગ છે જે નિર્ધારિત લક્ષ્ય પર આવેલ mail box છે જયાં અંતમાં સંદેશા પહોંચાડવામાં આવશે. Hotmail, સંદેશા માટેનું મેઈલ સર્વર છે જયાં John નું મેઈલ બોક્સ (Mail box) છે અને .com સંસ્થાનો પ્રકાર છે.

તેના છ મુખ્ય પ્રકાર છે.

.com	ધંધાર્થી સંસ્થાઓ
.edu	શૈક્ષણિક સંસ્થાઓ
.gov	સરકારી સંસ્થાઓ
.mil	સૈન્યદળ (military) સંસ્થા
.net	ગેટવે અને વહીવટી સંસ્થાઓ
.org	પ્રાઈવેટ સંસ્થાઓ

(બ) FTP (File Transfer protocols) ફાઈલ ટ્રાન્સફર પ્રોટોકોલ :

FTP એ ફાઈલ ઈન્ટરનેટ ઉપરથી લેવા કે મૂકવા માટે ઉપયોગી કોમ્પ્યુટર પ્રોગ્રામ છે. તે દૂરના

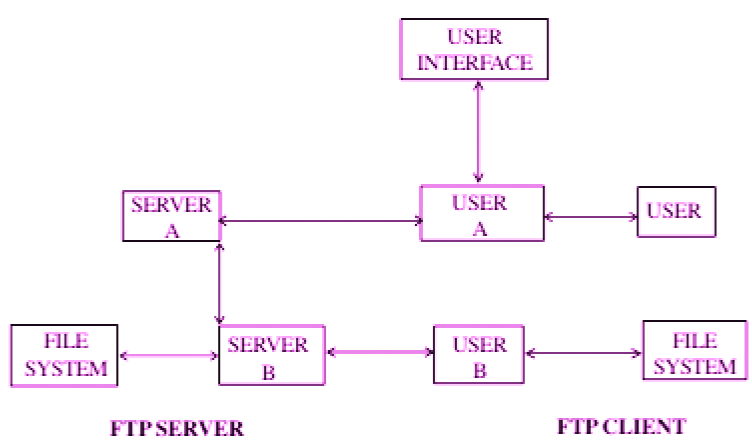
નોંધ

કોમ્પ્યુટરો પર હાજર ફોલ્ડરોને વાપરવા દે છે અને પ્રોગ્રામ, ડેટા અને ટેક્સ ફાઈલોને અલગ-અલગ કોમ્પ્યુટરો વચ્ચે બદલી કરવામાં મદદ કરે છે. FTP, કલાયન્ટ/સર્વરના આદર્શ પર જ કામ કરે છે. FTP Client એ તમારા કોમ્પ્યુટરમાં ચાલતો પ્રોગ્રામ છે જે તમને દૂરના કોમ્પ્યુટરમાંથી વસ્તુઓ લેવા માટે અને વાત કરવા સક્ષમ બનાવે છે. FTP Client, FTP કમાન્ડ લે છે અને તેને FTP Server ને માહિતી મેળવવા માટે વિનંતીના સ્વરૂપમાં મોકલે છે. FTP સર્વર સાથે જોડાય છે ત્યારે તે યુઝરનેમ અને પાસવર્ડ માટે પુછે છે. જો FTP સર્વરમાં ખાતું ના હોય તો પણ નનામુ (canonymous) લોગીન વાપરીને FTP સર્વર સાથે જોડાઈ શકો છો જે આકૃતિ 6.1 માં દર્શાવેલ છે.

નનામું ખાતું વાપરીને કોઈપણ FTP સર્વરમાં લોગીન થઈ અને પબ્લીક ફાઈલનો ભંડાર વાપરી શકે છે. કોઈપણ સરળતાથી FTP માં લોગીન કરી શકે છે જેના માટે anonymous યુઝરનેમ અને ઈમેઈલ એડ્રેસ પાસવર્ડ તરીકે વાપરવું પડે છે.

FTP ના પાયાના ઉદ્દેશો

- કોમ્પ્યુટર પ્રોગ્રામ, ફાઈલ અને ડેટાને વહેંચવાની પ્રવૃત્તિને પ્રોત્સાહન આપવા
- નેટવર્ક ઉપર ભરોસાપાત્ર અને અસરકાર રીતે ડેટા મોકલવા માટે
- દૂરના કોમ્પ્યુટરને વાપરવાને પ્રોત્સાહન આપવા
- ફાઈલ સંગ્રહક સીસ્ટમમાં ફેરફારથી વપરાશકર્તાને રક્ષણ આપવા



આકૃતિ. 6.1

FTP સેશનના પાયાના પગલાં :

- તમારૂં FTP Client શરૂ કરો તેના માટે સીસ્ટમના command prompt માં FTP ટાઈપ કરો. c:\> prompt(અથવા FTP ના ચિહ્ન ઉપર બે વાર કલીક કરો જો તમે વિન્ડોઝમાં હોય)
- FTP Client ને જે FTP સર્વર સાથે જોડાવું હોય તેનું સરનામું આપો.
- તમારી ઓળખાણ માટે લોગીન નામ આપો.
- દૂરની સાઈટને પાસવર્ડ આપો.
- દૂરની સાઈટ (Remote site) લોગીન નેમ/પાસવર્ડની ચકાસણી કરશે.
- FTP સર્વરમાં ફાઈલ માટે ડિરેકટરી જોવો
- જો જરૂર જણાય તો ડિરેકટરી બદલો
- બદલી મોડ સેટ કરો (optional)
- જોઈતી ફાઈલ મેળવો અને
- બંધ કરો

(ક) Telnet (Remote Computing) - ટેલનેટ

Telnet એ દૂરસંચાર માટે ઉપયોગી પ્રોગ્રામ છે, જે હાજર દૂરસંચારની સગવડો વાપરીને તમને દૂરના કોમ્પ્યુટર ઉપર વપરાશકર્તા બનવાની છૂટ મળી જાય પછી તમારા ઈચ્છિત કાર્ય માટે તેનો ઉપયોગ કરો. Telnet સ્ટેમ્પ બાય સ્ટેપ રીતથી કાર્ય કરે છે. કલાયન્ટ કોમ્પ્યુટર ઉપર ટાઈપ કરેલ કમાન્ડ સ્થાનિક ઈન્ટરનેટ સેવા આપનાર અને ત્યાંથી રીમોટ સર્વર જેને વાપરવાની છૂટ મળે છે તેના પર મોકલવામાં આવે છે. મોટાભાગના ઈન્ટરનેટ સેવા આપનાર, Telnet ની વ્યવસ્થા તમારા ખાતામાં જ આપે છે જયારે તમે તમારા ધંધાના કામે મુસાફરી કરતા હોય, ત્યારે તમારા ઈ- મેઈલ બીજા શહેરમાંથી પણ ચેક કરી શકો છો. Telnet સેશન માટે નીચેના પગલાં જરૂરી છે.

- Telnet પ્રોગ્રામ શરૂ કરો
- Telnet પ્રોગ્રામને જોડાવવા માટે એડ્રેસ આપો
- escape અક્ષર શું છે તેની નોટ બનાવો
- રીમોટ કોમ્પ્યુટરમાં લોગીન કરો
- ટર્મીનલ ઈમ્યુલેશન સેટ કરો
- રીમોટ કોમ્પ્યુટર વાપરો
- બંધ કરો.

નોંધ

પાઠગત પ્રશ્નો 6.1

1. *ઈન્ટરનેટને ટૂકમાં વ્યાખ્યાયિત કરો.*

૨. *ઈ-મેલના લક્ષણો લખો.*

3. FTP *ના મૂળભૂત હેતુઓ શું છે ?*

4. **ખાલી જગ્યા પુરો**

(અ) ....એ સામાન્ય પોસ્ટલ મેલ કરતાં ઘણું ઝડપી છે.

(બ) ....એ ઈન્ટરનેટ પર ફાઈલ અપલોડ અને ડાઉનલોડ કરવા માટે વપરાતુ ઉપયોગીતા વધારતું સોફ્ટવેર છે.

(ક) કોમ્પ્યુટર પર લખાયેલ આદેશ (command) *સ્થાનિક ....પર મોકલવામાં આવે છે.*

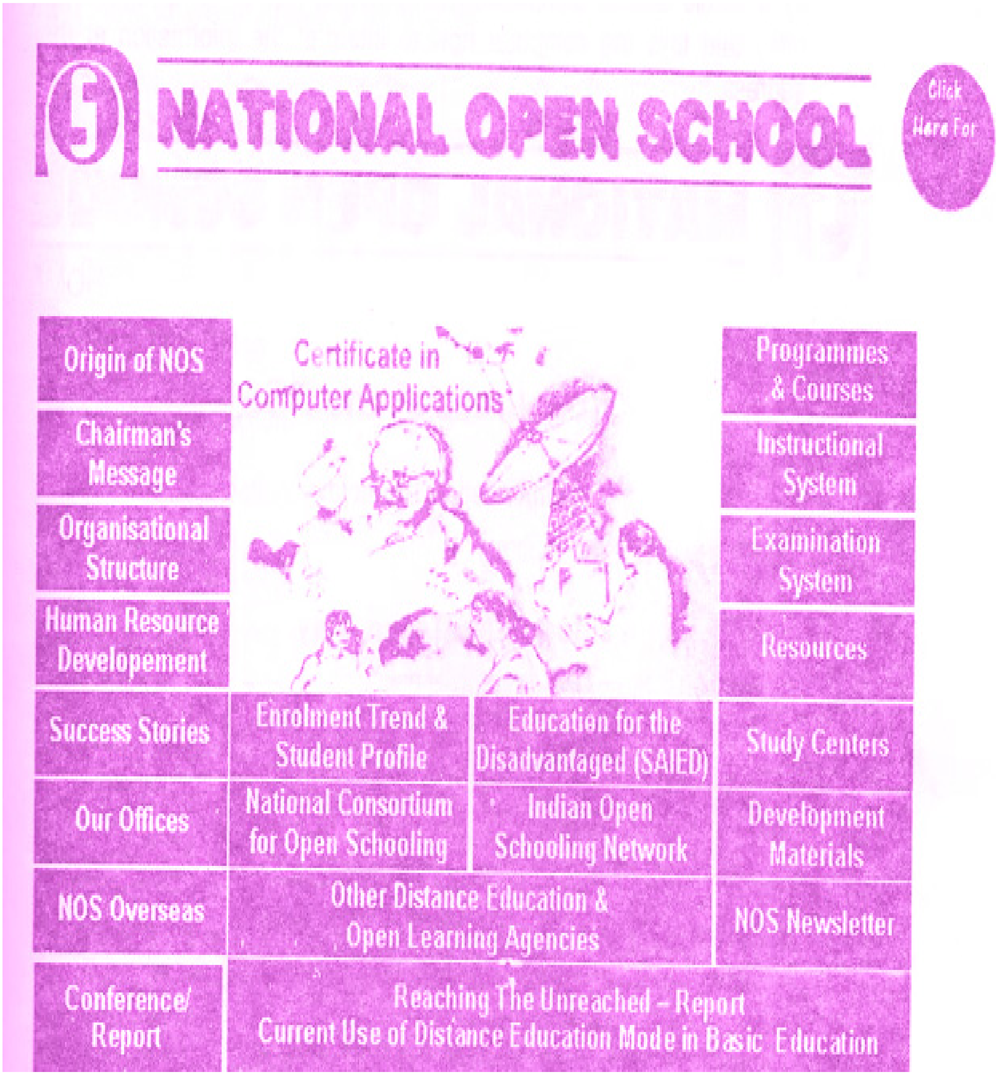
(ડ) ઈન્ટરનેટને વિશ્વના સૌથી મોટા કોમ્પ્યુટર નેટવર્ક તરીકે ગણવામાં આવે છે, તે ....નું નેટવર્ક છે જે સમગ્ર વિશ્વમાં પથરાયેલ છે.

6.5 વર્લ્ડ વાઈડ વેબ (WWW)

WWW*એ વર્લ્ડ વાઈડ વેબ માટેનું ટૂકું નામ છે. તે સામાન્ય રીતે 'ધ વેબ' તરીકે ઓળખાય છે. આ* WWW *એક હાઈપરટેક્સ્ટ આધારિત માહિતી પુનઃપ્રાપ્તિ માટેનુ સાધન છે. આપણે સરળતાથી કડીઓનો ઉપયોગ કરીને એક દસ્તાવેજ પરથી બીજા દસ્તાવેજ પર કુદીને વેબ સર્ફ કરી શકીએ છીએ. આ દસ્તાવેજો ઘણા બધા અલગ પ્રકારના હોઈ શકે જેમ કે ગ્રાફિક્સ, એનિમેશન, લખાણ, ધ્વનિ અને તાજેતરનાં વિડિયો તેઓ પણ આ બધાનુ એક જોડાણ હોઈ શકે. ઈન્ટરનેટ પર આ બધી માહિતી વપરાશકર્તાને એક દસ્તાવેજ સ્વરૂપે રજૂ કરાય છે જે વેબપેજ તરીકે ઓળખાય છે. આ બધા વેબપેજ એકબીજા સાથે કડી* (link) *થી જોડાય છે અને આ બધી કડીઓને હાયપર લિંક્સ* (hyper links) *તરીકે ઓળખાય છે.*

ઈન્ટરનેટ પર આ બધા વેબ પાનાને જોવા માટે વપરાતા સાધનને ઈન્ટરનેટ બ્રાઉઝર અથવા માત્ર બ્રાઉઝર તરીકે ઓળખાય છે. આ એક ખાસ પ્રકારનું સોફ્ટવેર છે જે વપરાશકર્તાની વિનંતી મુજબ માહિતીને ઈન્ટરનેટ પરથી શોધી તેને વેબપેજ રૂપે રજુ કરે છે. બજારમાં ઘણા બ્રાઉઝર ઉપલબ્ધ છે. માઈક્રોસોફ્ટનું ઈન્ટરનેટ એક્સપ્લોર અને નેટસ્કેપનું નેટસ્કેપ નેવીગેટર સૌથી વધુ લોકપ્રિય છે. ઈન્ટરનેટ પરથી માહિતી શોધીને બ્રાઉઝરમાં જોવાની પ્રક્રિયાને બ્રાઉઝિંગ અથવા

સર્ફિંગ તરીકે ઓળખાય છે.



આકૃતિ. 6.2

(અ) ઈન્ટરનેટ સરનામું (Internet Address)

જેવી રીતે દરેક ઘર, દરેક ઓફિસ અને દરેક જગ્યાનું એક સરનામું હોય છે તેવી જ રીતે દરેક પાનાનું ઈન્ટરનેટ પર એક અનન્ય (unique) સરનામું હોય છે. આ સરનામાંનો ઉપયોગ ઈન્ટરનેટ પરથી વેબ પાનાને વપરાશકર્તા સુધી લઈ આવવા માટે થાય છે.

જેવી રીતે દરેક ઘર અથવા ઓફિસનું સરનામું પોસ્ટલ સરનામા તરીકે ઓળખાય છે તેમ ઈન્ટરનેટ પરનું સરનામું યુનિફોર્મ રિસોર્સ લોકેટર (Uniform Resource locator) તરીકે ઓળખાય છે. એક લાક્ષણિક ઈન્ટરનેટ સરનામું અથવા URL નીચે પ્રમાણે દેખાશે :

http://www.nios.org/computers/internet/url.htm

URL એ ઈન્ટરનેટ સાથે જોડાયેલ બધા કોમ્પ્યુટરમાંથી ચોકકસ વેબપેજની સ્થિતિ દર્શાવે છે. URL ના ઘટકો પ્રોટોકોલ, સર્વર અને વસ્તુનું પાર્થનેમ આપે છે. હવે આપણે ઉપર જણાવેલ URL તપાસીએ

પ્રોટોકોલ પછી કોલમ (:) હોય છે. (http:), સર્વરના નામ પહેલા બે સ્લેશ આવે છે (//www.nios.org) અને પાથનેમના દરેક ભાગમાં એક સ્લેશ આવે છે. (/computers/internet/url.htm) . પ્રોટોકોલએ નિયમોનો સમૂહ છે જે કોમ્પ્યુટરને જણાવે છે કે આપેલ સરનામા પરથી આવતી માહિતીનું અર્થઘટન કેવી રીતે કરવું.

આકૃતિ. 6.3

પ્રથમ ઘટક છે, પ્રોટોકોલ જે કોમ્પ્યુટરને જણાવે છે કે માહિતીનું અર્થઘટન કેવી રીતે કરવું. ઘણાં ઈન્ટરનેટ પરનાં પાના HTTP (Hyper Text Transfer Protocol), વાપરે છે બીજા કેટલાંક ઈન્ટરનેટ ઉપર વપરાતા પ્રોટોકોલ છે. : ફાઈલ ટ્રાન્સફર પ્રોટોકોલ (FTP), NEWS (Usenet news groups protocol), અને GOPHER (an alternative transfer protocol). ગોફર પ્રોટોકોલ મોટેભાગે હવે વપરાતો નથી.

બીજો ઘટક છે, સર્વર (www.nios.org) તે એવા કોમ્પ્યુટરને ઓળખી બતાવે છે કે જ્યાં તમને જોઈતી માહિતી સંગ્રહિત હોય અને તેની આગળ હંમેશા બે સ્લેશ (//) આવે છે. સર્વરએ એક કોમ્પ્યુટર છે જેમાં બધી માહિતી સંગ્રહિત હોય છે અને કલાઈન્ટ જયારે વિનંતી મોકલે ત્યારે

તેને માહિતી મોકલે છે. ઈન્ટરનેટ પરના દરેક સર્વરનું એક અનન્ય સરનામું હોય છે જેનું નામ, સર્વર જાળવતી સંસ્થાનો ઉલ્લેખ કરે છે.

છેલ્લો ઘટક (/computers / internet/), સર્વરની અંદર વિનંતી કરેલ માહિતી (url.htm) નો રસ્તો (પાથ) આપે છે. મોટાભાગના વેલ પેજના નામની પાછળ .htm અથવા .html હોય છે.

6.5.1. ઈન્ટરનેટ સાથે કેવી રીતે જોડાણ મેળવવું (How to get connected to Internet)

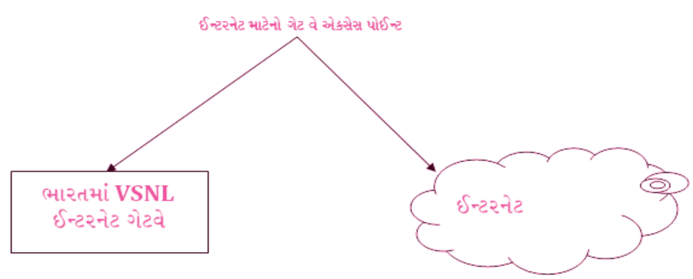
ઈન્ટરનેટ સાથે જોડાવવા માટેના ઘણા અલગ અલગ પ્રકારના જોડાણ ઉપલબ્ધ છે. આ બધાને નીચે મુજબ વર્ગીકૃત કરી શકાય.

- ગેટવે એક્સેસ (Gateway access)
- ડાયલ-અપ જોડાણ (dial-up connection)
- શેલ જોડાણ (shell connection)
- TCP/IP જોડાણ
- લીઝડ જોડાણ (Leased connection)

ગેટ વે એક્સેસ (Gateway Access)

ગેટવે એક્સેસને લેવલ વન જોડાણ તરીકે પણ ઓળખાય છે. તે એવા નેટવર્ક પરથી ઈન્ટરનેટનો એક્સેસ અપાવે છે જે ઈન્ટરનેટ સાથે જોડાયેલ ના હોય. ગેટ વે બે અલગ અલગ પ્રકારના નેટવર્કને વાત કરવામાં મદદ કરે છે. પરંતુ ગેટવે નેટવર્કના વપરાશકર્તાને ઈન્ટરનેટનો મર્યાદિત વપરાશ મળે છે. તેઓ ઈન્ટરનેટ પરના બધા સાધનો વાપરવા માટે સક્ષમ ન પણ હોય. સ્થાનિક ઈન્ટરનેટ સેવા પ્રદાતા (ISP) સામાન્ય રીતે આ મર્યાદા વ્યાખ્યાયિત કરે છે.

ભારતની અંદર લેવલ વન સાથે નેટવર્કનું સારું ઉદાહરણ એ VSNL (વિદેશ સંચાર નિગમ લિમિટેડ) છે. ભારતમાંથી ઈન્ટરનેટનો બધો વપરાશ VSNL ગેટ વે ધ્વારા આકૃતિ 6.4 માં જણાવ્યા પ્રમાણે મળે છે.

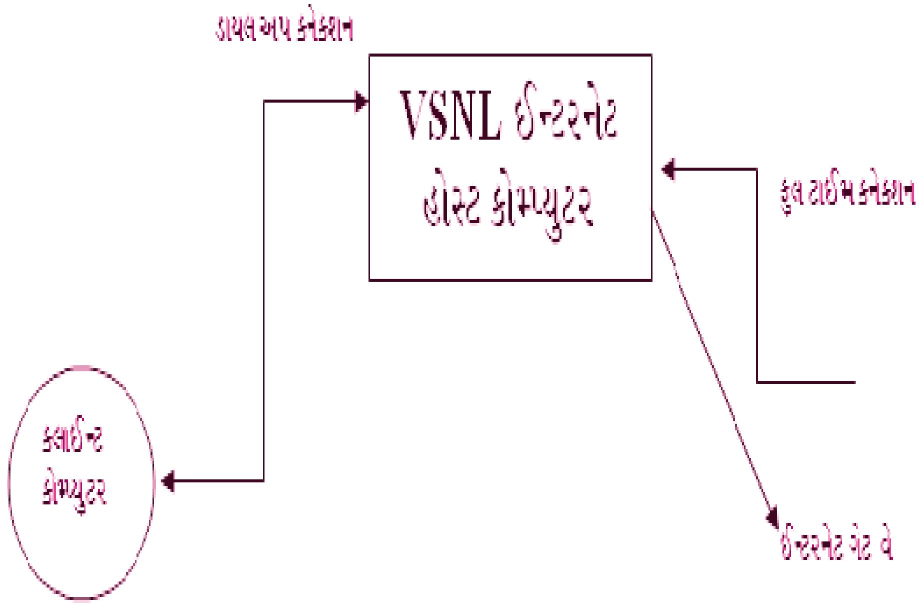


આકૃતિ. 6.4

ડાયલ-અપ જોડાણ (Dial-up connection)

ડાયલ-અપ જોડાણએ લેવલ બે પ્રકારના જોડાણ તરીકે ઓળખાય છે. તે ડાયલ-અપ ટર્મિનલ

જોડાણ મારફતે ઈન્ટરનેટનું જોડાણ આપે છે. જે કોમ્પ્યુટર ઈન્ટરનેટ એકસેસ અપાવે છે તેને હોસ્ટ (Host) તરીકે ઓળખાય છે અને જે કોમ્પ્યુટર માહિતી મેળવે છે તેને ક્લાઈન્ટ અથવા ટર્મિનલ કહેવાય છે. ક્લાઈન્ટ કોમ્પ્યુટર હોસ્ટનો વપરાશ કરવા માટે મોડેમનો ઉપયોગ કરે છે અને ટર્મિનલ તરીકે કામ કરે છે, જો તે સીધું જ હોસ્ટ સાથે જોડાયેલ હોય. તેથી આવા પ્રકારના જોડાણ એ 'રિમોટ મોડેમ એકસેસ' જોડાણ તરીકે પણ ઓળખાય છે. અને ખરેખર ક્લાઈન્ટ જે હોસ્ટ સાથે જોડાય છે તે પૂર્ણ સમય માટે ઈન્ટરનેટ સાથે જોડાય છે, જે આકૃતિ 6.5 માં બતાવ્યું છે.



આકૃતિ. 6.5

ઈન્ટરનેટ પરના ડાયલ આપ જોડાણમાં, હોસ્ટ બધાં જ આદેશો (commands) જે ક્લાઈન્ટ મશીન પરથી આવે છે તેને ઈન્ટરનેટ તરફ આગળ મોકલે છે. તે ડેટા અને માહિતીને ઈન્ટરનેટ તરફથી મેળવી તેને ક્લાઈન્ટ તરફ મોકલે છે. અહિં ક્લાઈન્ટ કોમ્પ્યુટર એક 'મૂક' (dumb) ટર્મિનલ તરીકે રિમોટ હોસ્ટ સાથે જોડાય છે.

આ પ્રકારના જોડાણ નીચે મુજબ બે ભાગમાં વર્ગીકૃત કરી શકાય.

(3) શેલ જોડાણ (shell connection)

આ પ્રકારના ઈન્ટરનેટ જોડાણમાં, વપરાશકર્તાને ફક્ત લખાણ (texual) ધરાવતું વેબ પેજ મળે છે. આ જોડાણ ગ્રાફિક્સ બતાવી શકતું નથી પરંતુ વપરાશકર્તા ઈન્ટરનેટ સર્ફ કરી શકે, FTP કરી શકે અને મેલ મેળવી શકે છે. બહુવર્ષો પહેલા ઈન્ટરનેટ પર જયારે ગ્રાફિક્સ પ્રચલિત નહોતું ત્યારે શેલ પ્રકારના ઈન્ટરનેટ જોડાણ વપરાતા હતા.

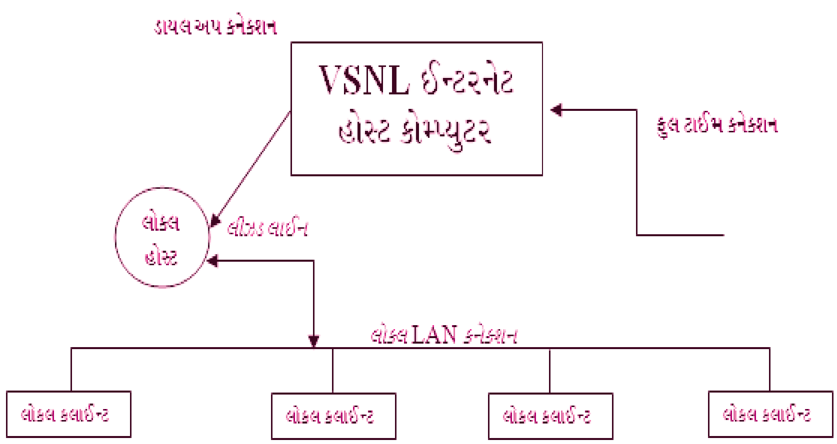
(4) TCP/ IP જોડાણ

આજના ગ્રાફિકસ વર્લ્ડ વાઈડ વેવ બ્રાઉઝર્સમાં સરળતાથી મલ્ટીમીડીયા અવાજ અને ચિત્રો વાપરી શકાય છે. શેલ અને TCP/IP પ્રકારના ખાતા વચ્ચે અગત્યનો તફાવત એ છે કે શેલ ફકત લખાણ બતાવી શકે છે અને તે ગ્રાફિકસ બતાવી શકતું નથી. જયારે TCP/IP બંને બતાવી શકે છે આથી TCP/IP વધારે પ્રચલિત ઈન્ટરનેટ જોડાણ છે. શેલ ખાતા ઈન્ટરનેટ પરથી ધીમેથી સમાપ્ત થઈ રહ્યા છે.આમાંથી કોઈ પણ ડાયલ અપ એકાઉન્ટ માટે નીચેની વસ્તુઓ જોઈએ :

- *કોમ્પ્યુટર*
- *મોડેમ*
- *ટેલિફોન જોડાણ*
- ISP *પાસેથી શેલ અથવા* TCP/IP *ખાતું*
- *ઈન્ટરનેટ બ્રાઉઝર જેવા ઈન્ટરનેટ કલાઈન્ટ સોફટવેર*

લીઝ્ડ જોડાણ (Leased connection)

લીઝ્ડ જોડાણએ સીધુ ઈન્ટરનેટ એકસેસ અથવા લેવલ ત્રણ પ્રકારના જોડાણ તરીકે પણ ઓળખાય છે. તે સુરક્ષિત, સમર્પિત અને સૌથી મોઘું ઈન્ટરનેટ જોડાણ છે. લીઝ્ડ જોડાણ ધ્વારા તમારું કોમ્પ્યુટર સમર્પિત અને સીધીરીતે ખૂબ જ ઝડપી ટ્રાન્સમિશન લાઈન ધ્વારા ઈન્ટરનેટ સાથે જોડાય છે. તે દિવસના ચોવીસ કલાક અને અઠવાડીયામાં સાતે દિવસ ઓનલાઈન (ચાલું) રહે છે. આ પ્રકારના જોડાણ ખૂબ જ મોટા કોર્પોરેશન અને યુનિવર્સિટીઓ સુધી મર્યાદિત છે કે જેમને આ જોડાણની કિંમત પરવડે. આ વિગતો આકૃતિ 6.6 માં બતાવી છે.



આકૃતિ. 6.6

નોંધ

પાઠગત પ્રશ્નો 6.2

1. વર્લ્ડ વાઈડ વેબનો સંક્ષિપ્તમાં વ્યાખ્યાયિત કરો.
૨. ઈન્ટરનેટ પર હૂક સાથે જોડાણ મેળવવાની વિવિધ રીતો શું છે ? તેને ટૂકમાં સમજાવો.
3. નીચેના નિવેદનો સાચા છે કે ખોટા તે જણાવો.

   (અ) www એ વર્લ્ડ વાઈડ વેબનું ટૂકું નામ છે.

   (બ) TCP/IP પ્રકારના જોડાણમાં વપરાશકર્તાને વેબપેજમાં માત્ર લખાણ મળશે.

   (ક) લીઝ્ડ જોડાણને સીધા ઈન્ટરનેટ એકસેસ તરીકે પણ ઓળખાય છે.

   (ડ) ડાયલ-અપ જોડાણને લેવલ બે પ્રકારના જોડાણ તરીકે પણ ઓળખાય છે.

6.6 Java and C++

Java એ C અને C++ નું derivative છે. સરળ, પરિચિત વાકયરચના અને ઓછા complex લક્ષણો ધરાવે તે ન સમજી શકે એવા, ગૂંચગણભર્યા C++ જેવા લક્ષણો ધરાવતી નથી. તેમાં Pointer નથી ફકત બાઉન્ડેડ array હોય છે. જેનાથી મેમરીમાં over writing અને અજાણતા માહિતી ભ્રષ્ટ થવાની શકયતા ઘટી જાય છે. પોઈન્ટર C અને C++ ના પ્રોગ્રામમાં કાચા શકિત સ્રોત બની રહયા છે. જયારે સાથેસાથે દરેક પ્રોગ્રામમાં ભુલો શોધવામાં પ્રાથમિકતતા દર્શાવે છે. java એ પ્રોગ્રામરોને જટિલ વિચારોને વ્યકત કરવામાં મદદરૂપ થાય છે આપો-આપો ગારબેઝ કકટર દ્વારા મેમરી પણ મેનેજ કરે છે. વધુમાં તે C અને C++ જેમ અને Strucures અને enum ને સપોર્ટ કરતું નથી. દરેક java ના પ્રોગ્રામમાં Object એ એક Class છે. આ કલાસ (Class) એ બન્ને static અને dunamic વારસા માટે અને ફરીથી એકનો અંક Code ફરીથી પુનઃઉપયોગ કરવાની પરવાનગી આપે છે. java એ multiple વારસાને આધાર આપતું નથી કે જયાં સિમેન્ટીક (semantic) અને અન્ય ભાષાઓમાં તદ્દન વિવાદાસ્પદ હોય. એવા ઘણા માળખાગત Program છે જેમાં goto વિધાન ન હોય તથા automatic ટાઈપ કાસ્ટીગ અથવા operator ના હોય.

6.7 Characterstics of Java (Javaની લાક્ષણિકતા)

(1) Simple and small (સરળ અને નાના)

Java માં સરળતાથી Program લખી શકાય છે, ઓછામાં ઓછી જરૂરિયાત સાથે જયારે તે C++ આધારિત છે. મોટા ભાગના tools ઉપલબ્ધ છે. જે મોટા અને Complex છે, અને સાથે powerful (શક્તિશાળી) પ્લેટફોર્મ પણ જરૂરી છે. પરંતુ Java માના Component ઘણા નાના છે. જે નાના મશીનો ઉપર Computer software સાથે એકલા ચાલી શકે છે.

(2) Object Origented (ઓબ્જેક્ટ કેન્દ્રિત)

Object *કેન્દ્રિત આધારિત ડિઝાઈન, નાના, મોડયુલર છતાં* Self contained *વિકાસ સક્રિય કરે છે, જેને દરકેને પાછળથી કોઈ પણ બ્લોકમાં વાપરી શકાય છે. તે પણ ઈન્ટરફેસથી સરળ વ્યાખ્યા આપે છે, ફરીથી વાપરી શકાય તેવા સોફટવેર ઘટકો કે જે જેમાં 'પ્લગ અને પ્લે' ની પરવાનગી આપે છે કે જે વિવિધ હાર્ડવેરમાં* distribute *હોય છે.*programmer *પોતાની* application *અને* data *પણ* object *સાથે જેથી* (attack) *કરી શકે* Java *માં એવા ઘણા* object *કેન્દ્રિત લક્ષણો છે જે* C *અને* C++ *જેવા લાગે છે. તેથી,* Programmer Java *માટે સરળતાથી સ્થાનાંતરિત કરી શકે છે અને જે ઝડપથી આગળ વધી શકે છે.*

એક સાચી Object *કેન્દ્રિત* Programming *માટે ઓછામાં ઓછી ચાર લાક્ષણિકતાઓ હોવી જોઈએ.*

- *ઈનકેપ્સુલેશન* (Encapsulation) *: માહિતી છુપાવી અને મોડ્યુલારીટી અમલમાં મૂકી શકે.*
- *પોલીમોરફીઝમ* (Polymorphism) *: વિવિધ* Object *ને મોકલવામાં આવતા સંદેશના પરિણામો એ* Object *ના સ્વભાવ કે વર્તન ઉપર આધાર રાખે છે.*
- *વારસા* (Inheritance) *: નવા વર્ગ* (Class) *ની ડેફીનેશન માટે હાલના વર્ગો તથા તેમના અંદરના કોડને કોઈ ઉપયોગમાં લઈ શકાય છે.*
- *ગતિશીલ બંધનકર્તા* (Dynamic binding) *:* Object *ગમે ત્યાં કદાચ આખા નેટવર્કમાં આવી શકે. તમે* Object *વગર સંદેશા મોકલી શકે તે પણ તેની* type *જાણ્યા વગર,* Program *લખતા સમયે ગતિશીલ બંધનકર્તા મહત્તમ સુગમતા* (flexibility) *પૂરી પાડે છે.*

હવે, Object *ઉપર ધ્યાન આપો. તેઓ સોફ્ટવેર પ્રોગામીંગ મોડેલ છે. તમારી રોજીંદા જીવનમાં તમે* Object *દ્વારા દર્શાવેલા છે. કાર, કોફી મશીન, વૃક્ષ અને તેનાથી પણ આગળ. સોફટવેર* Program *નો સમાવેશ થાય છે : ઈન્ટરફેસ અને સ્પ્રેડશીટ ઉપર બટનો અને તેનાથી પણ આગળ, આ* Object *નું* State *અને* behaviour *(વર્તન) હોય છે. તમે આ દરેક* Object *ને સોફ્ટવેરની રચનામાં ઉપયોગ કરી શકો છો, જેને તેમનાં* state *અને* behavour *થી વ્યાખ્યાયિત કરી શકાય છે. કાર ને* Object *થી બનાવી (*Modded*) શકાય છે. કારને* state *(તે કઈ દીશામાં, કેટલી ઝડપમાં અને કેટલી ફયૂઅલ વાપરશે, અને* behaviour *(શરૂ અને બંધ) હોય છે વગેરે.*

વર્ગ (class) *એ (*Java*) નો મૂળભૂત એકમ છે કેમ કે તે એક* Object *કેન્દ્રિત ભાષા છે. આ વર્ગો દ્વારા સામાન્ય રીતે માહિતીનો ઉપયોગ સરળ બને છે, અને સુરક્ષીત પણ બને છે અને ઈન્ટરફેશમાં તેના ઉપયોગ પણ (*Data *બદલવા માટે) સરળ બને છે. તે* Java *નાં પહેલાથી*

લખેલા Code ને વાપરવામાં (ઉપયોગમાં લેવામાં) સરળ છે અને તે પણ તે કેવી રીતે Implement થયો છે તેની ચિંતા વગર. Java માં ફરીથી વાપરી શકાય એવા Code નો વિશાળ ભંડાર છે. જેને API (Application Programming Interface) કહેવાય છે. જે Programming ને ખૂબ સરળ બનાવે છે.

(3) Portable અને Interpreted (પોર્ટેબલ અને ઈન્ટરપ્રિન્ટેડ)

Java માં લખાયેલ સોફ્ટવેર એપ્લિકેશન, તે એક શબ્દ પ્રોસેસર અથવા સ્પ્રેડશીટ કરી, એક મલ્ટીમીડિયા સાધન અથવા વર્ચ્યુઅલ રિયાલીટી ડિઝાઈનર દ્વારા કોઈ પણ કોમ્પ્યુટર ઉપર અને કોઈ પણ પ્લેટફોર્મ ઉપર સંપૂર્ણ પણે ચાલશે. Java એપ્લિકેશન એ complied અને Interpreted ભાષા છે. પ્રોગ્રામર દ્વારા લખવામાં આવેલો Code પ્રથમ Byte Code માં Compile થાય છે જે બધા મશીનો માટે એક સમાન છે. આ Byte code એ Java Virtual Machine સાથે સુસંગત છે. Interpreter એ Byte code ને અમલ માટે તેને એક Machine Code માં ફેરવે છે, કે જેનો સીધો અમલ કોઈ પણ Machine ઉપર થઈ શકે છે.

તેથી, તમે ગમે ત્યાં તમારો Code લખી શકો છો અને ગમે ત્યાં ચલાવી શકો છો. આ ખૂબ ઉપયોગી છે કેમ કે Programmer ને તેમનો Code બીજા મશીન પર રન થશે કે નહિ તેની ચિંતા હોય છે.

(4) Architecture - Neutral (આર્કિટેકચર - ન્યુટ્રલ)

Java કંપાઈલર એક architecutre Nautral ફાઈલ બહાર (generate) પાડે છે, generic બાઈટ કોડના સૂચનો, જેને કોઈ પણ Computer ના architecture સાથે કંઈ લેવા દેવા નથી. આ complied code કોઈ પણ પ્રોસેસર ઉપર execute થઈ શકે છે. જેના માટે Java runtime machine ની હાજરીથી જરૂરી છે જે તેને મૂળ machine code માં રૂપાંતર (convert) કરશે. એક જ સોફ્ટવેરની આવૃત્તિ તમામ પ્લેટફોર્મ પર ચાલે છે, કોઈ પણ CPU અથવા હાર્ડવેરની ચિંતા કર્યા વગર.

(5) Dynamic (ડાયનેમીક)

રન ટાઈમ વખતે interconnection ના લીધે Java ને સંપૂર્ણપણે બીજા સોફ્ટવેર Components પર આધાર રાખવો પડતો નથી. જેમ કે libraries. જેથી Java ના Program હંમેશા રન કરવા માટે ready હોય છે અને તે કોઈ પણ બીજા બહારી બદલાવની ચિંતા વગર.

(6) Distributed (વિતરણ)

Java એ TCP/IP, HTTP જેવા નેટવર્કીંગ પ્રોટોકોલ ના પાલન માટે જરૂરી Programs અને તે પણ વ્યાપક પ્રમાણ ધરાવે છે. Java એપ્લિકેશન Object Access કોઈ પણ URL થી કરી શકે છે, જેવી રીતે પ્રોગ્રામરનો પ્રોગ્રામ નેટવર્ક પર સ્થાનિક ફાઈલ કે ઓબ્જેક્ટ વાપરી શકે છે.

(7) Automaic Memory Management

Java Garbage Collector એ જેટલા object હોય એ બધાનું track ધ્યાન રાખે છે, કે જેથી તે object આગળ કોઈ ઉપયોગ ના હોય તેવા object ની memory મુક્ત કરી શકાય. આ એ કોડ મજબૂત બનાવવા માટે ખૂબ ઉપયોગી છે.

(8) Multi - Threaded

Multithreading ની મદદથી ઘણી બધી પ્રક્રિયા થકી એક એપ્લીકશેન બનાવી શકાય છે. Multithmeding ની મદદથી પ્રોગ્રામનું execation સરળ બને છે કે જેમાં ઘણા કાર્યક્રમો (threads) હોય છે. જેથી Garbage collector ખૂબ અસરકારક રીતે કામ કરી શકે છે. Multithreading તો thread class liberaies નો વારસા તરીકે સપોર્ટ મળે છે. તેનું પરિણામ સારું અને વાસ્તવિક સમય સાથે હોય છે. આ મુદ્દા એ Java interface તથા operating system ના ઘણા લક્ષણોને આધાર આપવા માટે મદદરૂપ છે.

મોર્ડન નેટવર્ક આધારિત એપ્લીકેશન જેવી કે હોટ Java WWW બ્રાઉઝર કે જેમાં એક સમય ઘણી બધી કામગીરી કરવી પડે છે. જે વપરાશકર્તા હોટ Java માં કામ કરતો હોય છે, તે animation ને લગતી ઘણી Image ને એક સાથે download અને Scroll કરી શકે છે. Java ના multimending ના મુદ્દા ને લીધે ઘણી બધી એપ્લીકેશનમાં એક સાથે ચાલતી પ્રક્રિયાઓને જાળવી શકાય છે.આને કારણે multithreading નો ઉપયોગ વપરાશકર્તા માટે ઘણા ઉંચા પ્રમાણમાં કરવામાં આવે છે.

(9) Robust (રોબસ્ટ)

પ્રોગ્રામ એ Robust છે કારણ કે Java માં પ્રોગ્રામર દ્વારા explicit મેમરી મેનિપ્યુલેશન થતું નથી.

(10) Secured (સુરક્ષીત)

Java ના પ્રોગ્રામ સુરક્ષીત છેઃ distributed એપ્લીકેશનમાં સુરક્ષા ઉચ્ચત્તમ સ્તર ઉપર હોય છે, જે Java interpreter નો ટાઈપકોડ ખાતરી આપે છે કે કંપાઈલ્ડ કોડ સખત રીતે ભાષાને અનુરૂપ હોય, જે ખોટી રીતે બદલાવથી બચાવે છે.

6.8.How Java ignores the identity of your chip ? ("Java" કેવી રીતે તમારા ચીપની ઓળખ અવગણે છે.

કોમ્પ્યુટરની અંદરની ચીપ ફક્ત મશીન કોડ જ સમજે છે. આ ખૂબ જટિલ હોવાથી પ્રોગ્રામરો ઊંચા સ્તરોની ભાષા વાપરે છે. આ સ્ત્રોત કોડ છે જેને CPU *સમજી શકે તેવા* Machine Code *માં રૂપાંતરિત થાય છે. જો કે,* Machine code *એક ચીપથી બીજી ચીપમાં બદલાય છે. ભાષાનું રૂપાંતર કરવા માટે* Complier *નો ઉપયોગ થાય છે. જો કે, રૂપાંતર એ સોફ્ટવેર ડેવલપરની ઓફીસમાં થાય છે અને એ રૂપાંતરિત* Version *તમે ખરીદી અને તેને તમારા* Computer *માં* Install *કરો છો. સામાન્ય રીતે, જે* Application *ને* Intelchip *માટે* Machine Code *માં રૂપાંતરિત કરી હોય તે આવે તો તે મોટોરોલા ચીપ કરતાં અલગ હોય છે. એ જ રીતે, તે* Space Workstation *માટે પણ અલગ હશે અને એમાં એવું કંઈ પણ સામાન્ય નહી હોય જે* Silicon *ગ્રાફિક્સ મશીન માટે જરૂરી હોય.*

પરંતુ, Java *એક અલગ માર્ગ અનુસરે છે. આ સંદર્ભમાં આપણે* Compling *અને* Interpreting *વિશે વધુ જાણવું જોઈએ. સંકલિત કાર્યક્રમ* (Complied Programms) *એક જ થઈને કોમ્પ્યુટર પર સમગ્ર મશીન કોડ ઉપલબ્ધ બનાવે છે. જેટલી વખત તમે એપ્લીકેશનનો ઉપયોગ કરશો, દરેક વખત તમારી કોમ્પ્યુટરની મેમરીમાં દરેક લાઈનનો મશીન કોડ સ્થાપિત થશે. તેનાથી વિપરીત, અર્થઘટન* (Interpreted) *સોફ્ટવેર તમારા આદેશો અનુસાર ફેરવાય છે. દરેક વખતે તમે એક ચોક્કસ ક્રિયા* (Operation) *કરવા માંગો છો. ત્યારે* Interpreter source code *ની લાઈનની શોધ કરે છે અને તેમને રૂપાંતરિત કરે છે, જેથી તે વધુ સમય લે છે.*

આપણે આગળ જોયું એ પ્રમાણે Java *અલગ કોમ્પ્યુટર પ્લેટફોર્મમાં સંપૂર્ણપણ પોર્ટેબલ છે. પ્લેટફોર્મ એ ચીપ અને ઓપરેટિંગ સીસ્ટમનું સંયોજન છે જેમ કે,* Intel - plus Windows 95. Java *એ ખરેખરે એક સંકલિત અને અર્થઘટન ભાષા સંયોજન છે. પ્રથમ* Source Code *ને પરંપરાગત સીસ્ટમ દ્વારા કંપાઈલ કરી મશીન કોડમાં ફીટ કરવામાં આવે છે. આ* Byte *તરીકે ઓળખાય છે.* Java *ની સંકલિત આવૃત્તિ બધા કોમ્પ્યુટર માટે સમાન છે. આ* Byte *એ જ કોમ્પ્યુટર સાથે સુસંગત* (Compatible) *છે. જે કોમ્પ્યુટર હાર્ડવેર તરીકે નહી પણ સોફ્ટવેર તરીકે અસ્તિત્વમાં હોય :* The Java Virtual Machie (JVM)

આવશ્યક પણે JVM *એ એક દુભાષિયો* (interpreter) *છે કે જે* Byte - code *સ્વીકારે છે અને તેને તમારા કોમ્પ્યુટરના વાસ્તવિક ચીપ માટે ચોક્કસ મશીન - કોડમાં ફેરવે છે. જે એક ચોક્કસ પોર્ટબીલીટીની તક આપે છે. એવા બધા કોમ્પ્યુટર કે જે* Java *ની એપ્લીકેશન ચલાવવા માંગે છે, જેમને વિવિધ ચીપ્સ માટે અલગ યોગ્ય* interpleter *ની જરૂર પડશે. સોફ્ટવેર ડેવલોપર્સ માટે, તમારું કોમ્પ્યુટર લાંબા સમય માટે* Intel PC *કે* Powermac, Alpha workstation *અથવા* sun spare station *નથી, જે સરળ રીતે, એક* JVM *જ છે તેથી તેમને માત્ર એક જ રીતે* Program *ની માત્ર એક આવૃત્તિ લખવી પડે છે, કે જેનાથી* software & hardware *ની વચ્ચે સુસંગતતા* (Compatibility) *ની ચિંતા કરવાની રહેતી*

નથી. Java ની પહેલા અને પછીનું Hardware રૂપરેખાંકન એ આકૃતિ. 6.7 અને 6.8 માં બતાવવામાં આવ્યું છે.

Fig.6.8 Hardware Compatibility Post-Java

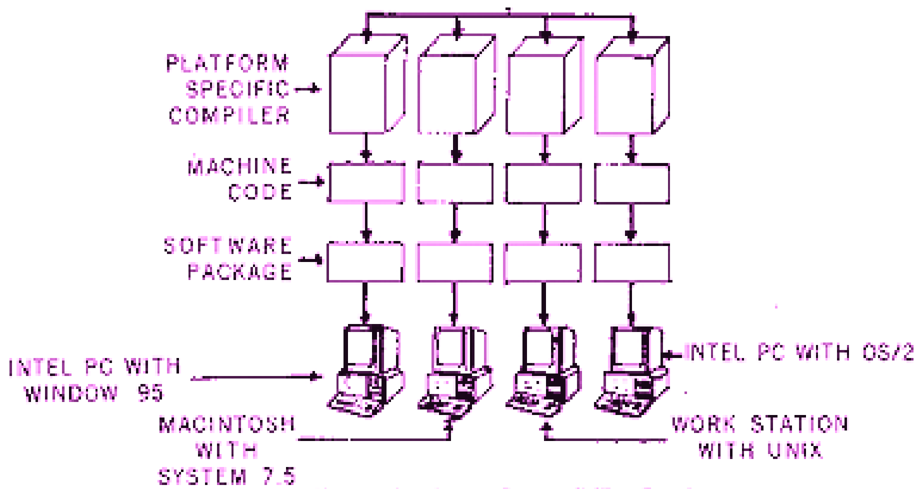


Fig. 6.7 Hardware Compatibility Pre-Java

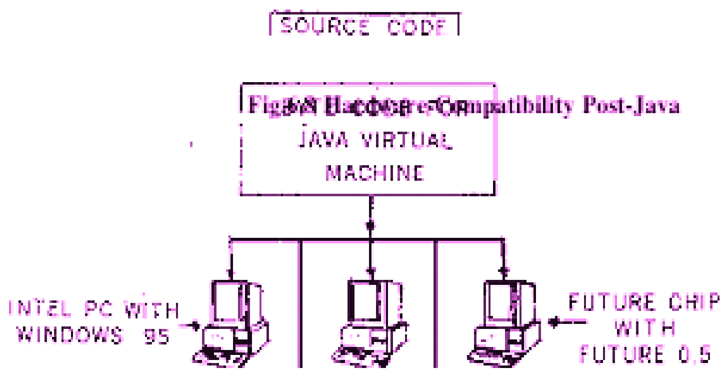


Fig.6.8 Hardware Compatibility Post-Java

આકૃતિ. 6.7 અને 6.8

6.9. Software Bussiness after Java

વિશાળ સોફટવેર પેકેજ નો વિશેષતાઓને લીધે તમે તેનો કયારે ઉપયોગ ના કરવાથી તે બિનજરૂરી બની જશે. તેની જગ્યાએ કમ્પ્યુટીંગની નાની એપ્લીકેશન કે જેને applets કહેવાય જે ઉપયોગમાં લેવાશે અને તે પણ ખાસ પ્રકારના કામ માટે.

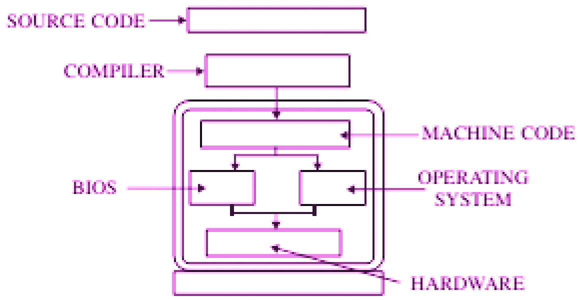
Applets એ કશું નથી પણ Java ના પ્રોગ્રામ જ સામાન્ય રીતે તેઓ નાના અને download કરી શકાય છે અને રન કરી શકાય છે. આ વેબપેજમાંનુ ડિઝાઈનર ભાગ પર ઓછામાં ઓછા પ્રયત્નો સાથે Static તેવા સમાવિષ્ટો માટે જીવન ઉમેરે છે.

આ Applets એ HTML ના ખાસ <applet> ટેગમાં લખાય છે. Applets તમારા સ્થાનિક મશીન પર અથવા બીજે Internet પર બહાર તમે ગમે ત્યાં સ્થિત કરી શકે છે. તે એક URL તરીકે ઉલ્લેખિત કરી શકાય છે. Appletનું સ્થાન વપરાશકર્તા સંપૂર્ણપણે અદશ્ય છે. Browser તેને મેળવશે અને ચલાવશે. એક એનાલિસ્ટ જે વિશ્વભરન સ્ટોક કવોટેશન ડાઉનલોડ કરે છે અને તેમના ઉપર ગણતરીઓ કરે છે. આજે તેમની પાસે તેમની હાર્ડ ડિસ્ક પર સોફટવેર હોવું જરૂરી છે. અને માહિતી તરીકે નેટનો ઉપયોગ થશે. Post Java માં વ્યક્તિને Net પરથી download નહી કરે, પરંતુ તેની સાથે એક applet પણ મેળશે અને તે કોમ્પ્યુટરની મેમરીમાં એક અડધા કલાકમાં Load થઈ જશે. તેની કાર્ય તુલના કરવાની થશે, જો Analyst ડેટાને Bar chart માં ફેરવવા માંગતો હોય તો તેમણે એક અલગ applet નો ઉપયોગ કરવો પડશે જે પણ Net ઉપર ઉપલબ્ધ છે. તેના ઉપયોગ બાદ, એટલે આ કોમ્પ્યુટર ઉપરથી અદશ્ય થઈ જાય છે. તમારે પેકેજો અને તેમના સુધારા તથા તેમના ઉંચા ભાવની વપરાશર્કતાને ચિંતા કરવાની જરૂર નથી. ફકત એક સરળ શબ્દ પ્રકિયા પેકેજની જરૂર પડશે. જો તમે કોઈ અન્ય પ્રસંગોપાત ઉપયોગ માટે અલગ feature ની જરૂર હોય, તો તેને Net ઉપરથી થોડા સમય માટે download કરી શકો છો, તમારે કાયમી માલિકી માટે પેકેજો ખરીદી કરવાની જરૂર નથી. તેની જગ્યાએ તમારે ઉપયોગના પ્રમાણે બિલ બનાવવામાં આવશે.

વધુમાં, Java ને લીધે મોંઘી વિતરણ અને રિટેલ નેટવર્કસની એન્ટ્રીની અડચણ તોડી શકાશે. જેથી નાની કંપનીમાં સોફટવેર વેન્ડરને ફાયદો થશે. નાના અને ઓછી કિંમતવાળા પ્રોગ્રામ ઉપર Net સધ્ધર બની જશે. નવીનીકરણ અને પ્રોગ્રામીંગ કુશળતાને Java ભોગે પ્રભુત્વ રહેશે. સોફટવેર સંદર્ભમાં પૂર્વ અને પોસ્ટ Java નીચેની આકૃતિ. 6.9 અને 6.10 માં સમજાવવામાં આવી છે.

6.10. જાવા અને ઈન્ટરનેટ

જાવા અને પ્રોગામીંગ ભાષાના ઉપયોગથી વેબ ડોકયુમેન્ટના લેખકો નાના એપ્લીકેશન પ્રોગ્રામને HTML માં મૂકી શકે છે. જેને કોઈ પણ વપરાશકર્તા બ્રાઉઝ કરી શકે છે. પાના જીવંત બને છે કારણ કે તે ગમે સ્કોર બોર્ડ, એનમિટેડ કાર્ટૂન, ધ્વનિ ફાઈલ અને વિડિયો કિલિપ બતાવી શકે છે, વધુમાં, તેણે ઈન્ટરનેટ વાપરવાની રીત બદલી છે અને અલગ અલગ પ્રકારનાં નેટવર્કમાંથી ડાયનેમિક રીતે કોડને સીસ્ટમમાં ગમે ત્યાંથી લોડ કરી WWW એ પારદર્શક રીતે ચલાવે છે. જાવા પહેલાં અને પછી ઈન્ટરનેટનો વપરાશ આકૃતિ. 6.11 અને 6.12 માં બતાવ્યો છે.

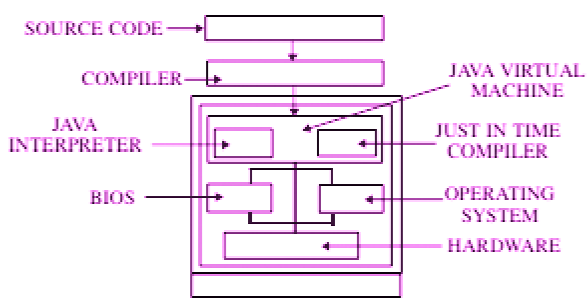


આકૃતિ. 6.9 (અ) સોફ્ટવેર આર્કીટેકચર પ્રી - જાવા

જાવાની સાથે, ગણતરીના મુખ્ય બે ઘટકો - ડેટા અને પ્રોગ્રામને તમારા કોમ્પ્યુટરમાં સંગ્રહ કરવાની જરૂર નથી. ઈન્ટરનેટ પર તે ગમે ત્યાં રહી શકે છે. જયારે તેની જરૂર પડે ત્યારે તેને બોલાવવામાં આવે છે. આ www એ વિશ્વમાં કરોડો વપરાશકર્તાઓને ઈન્ટરનેટની રજૂઆત કરી હતી. પરંતુ તેમના વશીકરણ માટે, મોટા ભાગના વેબપેજને પુસ્તક અથવા મેગેઝીનની જેમ સ્ટેટીક (સ્થિર) રાખ્યા છે. વેબ તકનીકી રીતે અવાજ અને વિડિયો પરું પાડવા સજજ છે. પરંતુ તેના માટે વપરાશકર્તાએ યોગ્ય સોફટવેર વાપરવું પડે છે.

જાવા તમને માહિતી સાથે સોફટવેર પેકેજ માટે પરવાનગી આપે છે. અન્ય શબ્દોમાં કહીએ તો જયારે તમે વિડીયો કિલપ ડાઉનલોડ કરો છો, ત્યારે આપો આપ તેને પ્લે (વગાડવા) કરવા માટેનું જરૂરી સોફટવેર મેળવે છે.

એક વાત ચોક્કસ છે વેબ એ શક્તિશાળી કોમ્પ્યુટર બન્યું છે, જે કોઈ ઉપયોગ કરી શકે, જેથી મોંઘા ડેસ્કટોપ કોમ્પ્યુટર તરત એક ઈન્ટરનેટ સાથે જોડાયેલા સરળ કોમ્પ્યુટર દ્વારા બદલાયેલા હશે. અને તેઓ તકનીકી રૂપે કોઈ પણ પ્રોગ્રામ દુનિયામાં કોઈ પણ જગ્યાએથી ડાયનેમીકલી લોડ કરી ચલાવી શકવા સક્ષમ હશે. કંપનીઓ ઈન્ટરનેટ પર કમ્પ્યુટિંગ ક્ષમતાઓ વેચશે, તેવી જ રીતે જેમ વીજળી બોર્ડ આપણને પાવર આપે છે. નજીકના ભવિષ્યમાં આપણી મોટા ભાગની મનપસંદ પ્રવૃત્તિઓ ઈન્ટરનેટ પર હશે અને આપણી સફળતાનો આધાર ઈન્ટરનેટ ટેકનોલોજી સ્વીકારવાની અને તેનો ઉપયોગ કરવાની ક્ષમતા પર હશે.

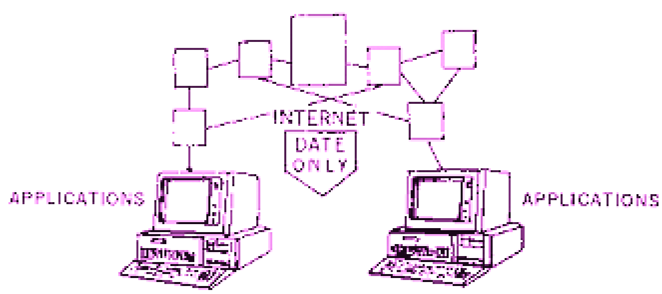


આકૃતિ. 6.10 સોફ્ટવેર આર્કીટેકચર પોસ્ટ જાવા

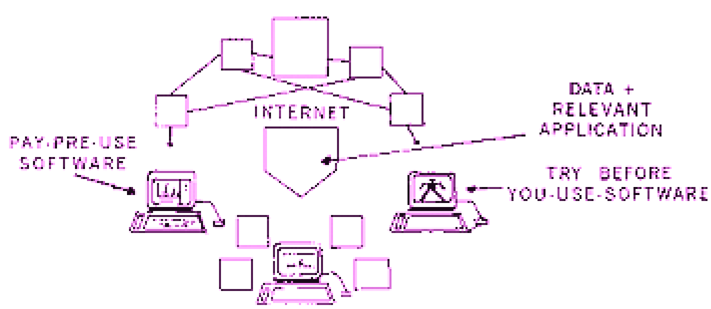
જાવાનું ભવિષ્ય (Future of JAVA)

જાવા પરંપરાગત પ્રોગ્રામિંગ ભાષાઓ કરતાં ધીમે ચાલે છે અને સોફ્ટવેર લાયબ્રેરી જે પ્રોગ્રામરના કાર્યને સ્ટ્રીમ લાઈન કરે છે, તે હજુ પણ લખવામાં આવી રહી છે. જાવા એ સર્વરની મદદથી ક્લાઈન્ટ માટેની પ્રોગ્રામિંગ ભાષા બની રહી છે જાવામાં બનેલા મોટા ભાગના પ્રોગ્રામ એ એપલેટ (applets) *છે જે નાના કાર્યો કરવા માટે રચાયેલા કે જેવા કે લોગોનું એનિમેશન, કાર્ટૂનના કરેક્ટર બનાવવા, વગેરે. ક્લાઈન્ટ સર્વરને લગતા સોફ્ટવેર બનાવવા પડે છે. આ પ્રકારની એપ્લિકેશન માટે, જાવાને ડેટા એક્સેસ કરવા માટે સ્વચ્છ, ઉચ્ચ સ્તરીય ડેટાબેઝ અને સર્વર વચ્ચે ઈન્ટરફેસની જરૂર પડે છે, જેવા કે* ODBC *અને* OLE (Object Linking and Embedding).

જયાં આ મુદ્દાઓ સંબોધવામાં આવી રહ્યા છે, ત્યારે ભારતમાં પણ જાવા સાથે વધુને વધુ લોકો સંકાળાયેલા છે. જાવાને ભારતીય સોફ્ટવેર ઉદ્યોગ માટે ખૂબ જ સારી તક તરીકે જોવામાં આવે છે. વિશ્વ સાથે ઓનલાઈન રીતે જોડાઈ ઉદ્યોગોને સતત વૃધ્ધિ દર, જાવા કદાચ ભારતીય સોફ્ટવેર ઉદ્યોગના નવસંસ્કરણ માટે ઉત્પ્રેરક તરીકે કામ કરશે અને સોફ્ટવેર ઉદ્યોગની વૈશ્વિક છાપ બનાવશે.



આકૃતિ. 6.11 ઈન્ટરનેટનો વપરાશ - જાવા



આકૃતિ. 6.12 ઈન્ટરનેટનો વપરાશ - જાવા

પાઠગત પ્રશ્નો 6.3

1. જાવા પ્રોગ્રામિંગ ભાષાની વિવિધ લાક્ષણિકતાઓ લખો.
2. નીચેના વિધાનો સાચા છે કે ખોટા તે જણાવો.

   (અ) જાવામાં કોઈ પોઈન્ટર નથી, તે પૂર્ણ પણે એરેના સીમિત છે.

   (બ) જાવા C++ ના જેવી Object Oriented સુવિધાઓ ધરાવતું નથી.

   (ક) જાવા એપ્લિકેશનને કંપાઈલ્ડ અને ઈન્ટરપ્રિન્ટેડ ભાષા સંયોજન છે.

   (ડ) જાવાના પ્રોગ્રામ મજબૂત પરંતુ સુરક્ષાના અભાવવાળા હોય છે.

6.11. તમે જે શીખ્યા (What you have learnt)

આ પાઠમાં આપણે ઈન્ટરનેટને લગતા મૂળભૂત ખ્યાલ, તેનો ઈતિહાસ અને ઈન્ટરનેટ દ્વારા અપાતી વિવિધ સેવાઓ વિશે શીખ્યા. આ પ્રક્રિયામાં, આપણે ઈ- મેલ અને ફાઈલ ટ્રાન્સફર પ્રોટોકોલના ઉપયોગ વિશે શીખ્યા. હવે, આપણી પાસે ઈન્ટરનેટ વાપરવા માટેનું પુરતું જ્ઞાન છે, અને તેને વિવિધ જગ્યાએ વાપરી શકીશું. પાઠના છેલ્લા ભાગમાં જાવા પ્રોગ્રામિંગ ભાષાની લાક્ષણિકતાઓ ટૂંકમાં સમજાવાઈ છે.

6.12. પાઠયાંત પ્રશ્નો (Terminal Question)

1. ઈન્ટરનેટનો ઈતિહાસ સંક્ષિપ્તમાં લખો.
2. ઈ- મેઈલની વિવિધ સુવિધાઓ લખો.
3. FTP ના મૂળભૂત હેતુઓ શું છે ? ટૂંકમાં સમજાવો.
4. ઈન્ટરનેટ સાથે જોડાવાના વિવિધ પ્રકારો વિશે ટૂંકમાં સમજાવો.
5. તમે જાવા પ્રોગ્રામિંગ ભાષા વિશે શું જાણો છો ?

6.13 પાઠગત પ્રશ્નોના જવાબ

પાઠગત પ્રશ્નો 6.1

1. ઈન્ટરનેટ ને નેટવર્કના નેટવર્ક તરીકે ગણવામાં આવે છે. જે પુરી દુનિયામાં પથરાયેલ છે. તે નિઃશકપણે વિશ્વનું સૌથી મોટું નેટવર્ક છે. જે સૂર્ય હેઠળ આવતા કોઈપણ વિષયની માહિતીનું વહન કરે છે.
2. ઈ- મેઈલના લક્ષણો નીચે મુજબ છે. :

   (અ) એકથી એક અથવા એકથી ઘણા બધા સંચાર

(બ) ત્વરિત સંચાર

(ક) પ્રાપ્તકર્તાની ભૌતિક હાજરી જરૂરી નથી.

(ડ) સૌથી સસ્તી મેલ સેવા છે જે ર4 કલાક અને સાતે દિવસ ચાલે છે.

(ઈ) અનૌપચારિક વાતચીતને પ્રોત્સાહન આપે છે.

3. FTP ના મૂળભૂત હેતુઓ નીચે મુજબ છે. :

(અ) સુગમતા આપે છે અને કોમ્પ્યુટર પ્રોગ્રામ, ફાઈલો અને ડેટાના શેરિંગને પ્રોત્સાહન આપે છે.

(બ)નેટવર્ક પર અસરકારક અને વિશ્વસનીય રીતે માહિતીની બદલી કરવા માટે.

(ક) ઈન્ટરનેટનો ઉપયોગ કરી દૂરના કોમ્પ્યુટરના ઉપયોગને ગર્ભિત (implict) કે પરોક્ષ (indirect) રીતે પ્રોત્સાહિત કરવા.

(ડ) ફાઈલ સંગ્રહ કરતી સિસ્ટમમોમાં ભિન્નતા દ્વારા વપરાશકર્તાને રક્ષણ.

4. (અ) ઈ- મેલ (બ) FTP (ક) ISP (ડ) નેટવર્ક

પાઠગત પ્રશ્નો 6.2

1. વર્લ્ડ વાઈડ વેબ અથવા માત્ર વેબ એ ઈન્ટરનેટની ઉપર બાંધવામાં આવે છે. તે હાઈપર ટેકસ્ટ આધારિત માહિતી પુનઃ પ્રાપ્તિ સાધન છે.

2. જોડાણના વિવિધ વિકલ્પો નીચે મુજબ છે :

(અ) ગેટ વે એકસેસ

(બ) ડાયલ અપ જોડાણ

(ક) શેલ જોડાણ

(ડ) TCP/IP જોડાણ

(ઈ) લીઝ્ડ જોડાણ

3. (અ) ખોટું (બ) ખોટું (ક) સાચું (ડ) સાચું

પાઠગત પ્રશ્નો 6.3

1. વિવિધ લાક્ષણિકતાઓ નીચે મુજબ છે :

(અ) સરળ અને નાના

(બ) ઓબ્જેક્ટ ઓરીએન્ટેડ

(ક) પોર્ટેબલ અને અર્થઘટન (interpreted)

(ડ) ડાયનેમિક અને વિતરણ (distributed)

(ઈ) મજબૂત અને સુરક્ષિત

2. (અ) સાચું (બ) ખોટું (ક) સાચું (ડ) ખોટું

નોંધ

7

C++ ની પ્રસ્તાવના

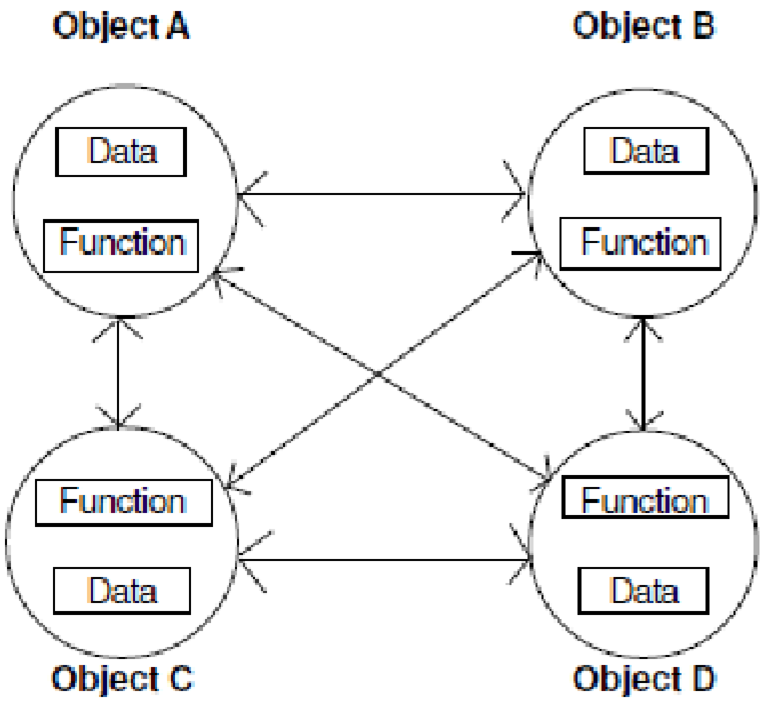
7.1 પ્રસ્તાવના

C^{++} એ C ને આગળ વધારતી પ્રોગ્રામિંગ ભાષા છે. તે 1980ના શરૂઆતમાં AT & T લેબોરેટરીમાં Bjarne Stroustrup વિકસાવેલ છે. તે પરંપરાગત ભાષાથી એવી રીતે અલગ પડે છે, કે જે ઓબ્જેક્ટ ઓરીએન્ટેડ પ્રોગ્રામને મોટા અને અધૂરા પ્રોગ્રામ માટે અનુકુળ છે. ઓબ્જેકટ ઓરીએન્ટેડ ભાષા માહિતીને તેના ફંકશન અથવા કોડ સાથે એવી રીતે જોડે કે જેથી માત્ર કોડ તેના ફંકશન અથવા કોડ દ્વારા જ ઉપયોગ થઈ શકે. આવા માહિતી અને કોડના સંગ્રહને Object (ઓબ્જેકટ) કહે છે. **ઉદાહરણ તરીકે** Tutor નામના ઓબ્જેકટમાં માહિતી અને કોડ નીચે પ્રમાણે હોય છે.

Object : Tutor
DATA Name Dept Empcode |
FUNCTION Tot _ pay ( ) Deduction ( ) Net _ Pay ( ) | |

માહિતીના ભાગમાં Name, Dept અને Employee Code કોડ છે. Part ફંકશન ત્રણ ફંકશન ધરાવે છે. Tot_ pay ( ), Deduction () અને Net_ Pay ( ).

આમ, ઓબ્જેક્ટ ઓરીએન્ટેડ ભાષાના વિવિધ ઓબ્જેક્ટ એકબીજા સાથે તેના કોડ અથવા ફંકશનથી આકૃતિ 7.1 પ્રમાણે જોડાયેલ હોય છે.



આકૃતિ : 7.1 ઓબ્જેક્ટ ઓરીએન્ટેડ માળખું

તે નોંધવા જેવું છે કે ઓબ્જેક્ટની માહિતી માત્ર તેની સાથે સંકળાયેલા ઓબ્જેક્ટના ફંકશન દ્વારા જ ઉપયોગ કરી શકાય છે.

7. 2 હેતુઓ

આ પાઠને તમે અંતે આટલું જાણી શકશો.

- C++ ના Character સેટ, ટોકન અને મુખ્ય ડેટા ટાઈપ.
- C++ માં વપરાયેલ અથવા ઉપયોગમાં લીધેલ વિવિધ ઓપરેટર.
- Implicit અને Explicit બદલી વચ્ચેનો તફાવત જાણે.
- C++ દ્વારા સાથે આવતા ઈનપુટ/ આઉટપુટ સ્ટ્રીમ સમજી શકે.
- C++ પ્રોગ્રામનું માળખું
- C++ માં પ્રોગ્રામ લખવો.

7.3 C++ કેરેક્ટર સેટ

કરેક્ટર સેટ એ ભાષા દ્વારા માન્ય કેરેક્ટરનો સમૂહ છે. કેરેક્ટર કોઈપણ લેટર, ડીજીટ અથવા અન્ય ખાસ કેરેક્ટર લેટર, ડીજીટ અથવા અન્ય ખાસ કેરેક્ટર ધરાવે છે. C++ ના કેરેક્ટર સેટ નીચે મુજબ છે.

લેટર A- Z, a - z

અંક 0 - 9

ખાસ કેરેક્ટર Space + - * / < / ( ) [ ] { } = ! = < > ' " $. ; ; % ! ! < = > = @

વ્હાઈટ સ્પેશ /હોરીઝેન્ટલ/ટેબ (→) કેરીજ રીડર, (↵), Newline, form feed, blank space

7.4 મૂળભૂત ડેટા ટાઈપ

દરેક પ્રોગ્રામમાં એક ઓપરેશન (કાર્યો) કે સમૂહ અમુક માહિતી માટે ચોક્કસ હરોળમાં ફાળવેલ હોય છે. માહિતી ઘણી પ્રકારની હોઈ શકે જેમ કે નંબર, કેરેક્ટર, બુલીઅન કિંમત વગેરે C++ મોટી સંખ્યામાં ડેટા ટાઈપને સાથ આપે છે. C++ મૂળભૂત ડેટા ટાઈપ અથવા બિલ્ટ ઈન (Built in) ને સાથ આપે છે તે Integer, Floating Point અને Character છે તેનું વિસ્તૃત ચર્ચા નીચે મુજબ છે.

7.4.1 Intenger Type (int)

Integer દશાંશ સંખ્યા સિવાય બધા જ નંબર છે. આ નંબર ગણતરી માટે વપરાય છે જેમ કે 26,373,-1729 માન્ય ઈન્ટીજર છે.

સામાન્ય રીતે intenger માં -32768 થી 32767 સુધીના નંબર હોય છે. જો જરૂરિયાત વધારે હોય તો long integer (log int) પણ વપરાય છે.જે 2,147,483,648 થી 2,147,483,647 સુધીના નંબરને સાથે આપે છે.

7.4.2 Floating Point Type (float)

Floating નંબર દશાંશ આંકડાને સાથ આપે છે. જો તે પૂર્ણ સંખ્યા ધરાવે તો તેના છેડે દશાંશ પોઈન્ટ હોવું જ જોઈએ. આ નંબર સંખ્યા માપવા વપરાય છે. તેના માન્ય નંબર 27.4, -927, 40.03 છે. Float type 10^{-38} થી 10^{38} સુધીના નંબર હોય છે. જેમાં છ થી સાત સુધીની ચોક્સાઈ હોય છે. વધારે ચોકસાઈ માટે double વપરાય છે. જે 10^{-38} થી 10^{38} જેમાં 15 આંકડા સુધીની ચોકસાઈ રાખે છે.

7.4.3 Character Type (Char)

તે આંકડાકીય ટાઈપ નથી. તે એક જ નંબર અને આલ્ફાબેટ ધરાવે છે. માન્ય Character type 'A', 'q', 'P', '8', '&' *છે. અહીંયા એ ધ્યાનમાં રાખો કે* 9 *અને* 'g' *એમ બંને અલગ ડેટા ટાઈલ છે. એકનો* type int *અને એક* char type *છે.*

7.5 ટોકનસ (Tokeus)

ટોકન એ Character *નો સમૂહ છે. પ્રોગ્રામર ટોકનસનો ઉપયોગ કરી પ્રોગ્રામ લખી શકે છે.* C++ *માં નીચેની પ્રકારના ટોકન્સ હોય છે.*

- Keywords
- Indeafitiers
- Literals
- Punctuations
- Operators

7.5.1 Keywords

C++ *માં કેટલાક આરક્ષીત શબ્દો છે કે જેને કમ્પાઈલર* Keyword *કહે છે. અમુક સામાન્ય કી વર્ડ નીચે આપેલ છે. "કીવર્ડનું લિસ્ટ"*

List of Keywords

asm	double	new	switch
auto	else	operator	template
break	enum	private	this
case	extern	protected	try
catch	float	public	typedef
char	for	register	union
class	friend	return	unsigned
const	goto	short	virtual
continue	if	signed	void
default	inline	sizeof	volatile
delete	int	static	while
do	long	struct	

7.5.2 Identifiers

પ્રોગ્રામર દ્વારા તેના પ્રોગ્રામમાં C++ માં આકાર ધરાવતા નામ (Symbolic Name) વપરાય છે. ઉદાહરણ, તરીકે, જો પ્રોગ્રામર 50 ની કિંમતને મેમરી લોકેશન પર સ્ટોર કરવા માંગે છે તો તે કોઈપણ Symbolic Name (જેમ કે) નો ઉપયોગ કરી શકે છે.

MARKS = 50

'=' એ assignment ઓપરેટર છે. ઉપરના વાકયનું મતલબ એ છે કે MARKS એ મેમરી લોકેશન માટેનું નામ છે જેની કિંમત 50 છે.

Symbolic નામ સામાન્ય રીતે Identifiers થી ઓળખાય છે. Identifier એ C++ ના કેરેકટર સેટમાં ક્રમીક આવતા કેરેકટર છે. તેના નિયમો નીચે મુજબ છે.

(i) Identifier alphabet આંકડા અને અથવા Underscores ધરાવે છે.

(ii) તે આંકડાથી શરૂના જ થવો જોઈએ.

(iii) C++ કેસ સેન્સીટીવ છે એટલે કે કેપીટલ અને લોવર કેસ લેટરને એકબીજાથી અલગ રીતે લે છે. TOTAL અને total એ બંને અલગ Identifiers છે.

(iv) તે આરક્ષિત શબ્દ ના હોવો જોઈએ.

7.5.3 Literals

Literals (જેને Constants પણ કહેવાય) તે ડેટા આઈટમ છે કે જે જેની કિંમત કયારેય પ્રોગ્રામના એકઝીકયુશનમાં બદલાતી નથી. C++ માં નીચે પ્રકારના Literals છે.

(i) Integer constants

(ii) Character constants

(iii) floating constants

(iv) String constants

(i) Integer Constants : તે પૂર્ણ નંબર છે જે કોઈ પણ અપૂર્ણ ભાગ ધરાવતા નથી. તે + અથવા - નિશાની ધરાવે છે. પરંતુ દશાંશ પોઈન્ટ અથવા અલ્પવિરામ કોઈ Integer constant ધરાવતા નથી. C++ 3 પ્રકારના Integer ને માન્ય કરે છે.

- Decimal (આધાર 10)
- Octal (આધાર 8)
- Hexadecimal (આધાર 16)

- **Decimal Integer Constant**
  તે આંકડાની હરોળ ધરાવે છે અને તે 0 થી શરૂ ના થવો જોઈએ. ઉદાહરણ તરીકે, 124, -179, + 108.
- **Octal Integer Constant**
  તે આંકડાની હરોળ કે જે 0 થી શરૂ થાય છે. તે ધરાવે છે, ઉદાહરણ તરીકે 014,012
- **Hexadecimal Integer Constant**
  તે આંકડાની હરોળ ધરાવે છે. જેની આગળ ox અથવા OX આવે.ઉદાહરણ તરીકે OXD,OXC. કોઈપણ Consant સાથે I અથવા L અને 5 અથવા U આગળ જોડાયેલ હોય છે. જે Long અથવા Unsigned ધરાવે છે.

(ii) Character Constants

C++ માં Character constants એક અથવા વધારે કેરેક્ટર ધરાવે છે જે એક Quatation Mark થી જોડાયેલું હોવું જ જોઈએ. ઉદાહરણ તરીકે 'A','9' વગેરે. C++ nongraphic કેરેક્ટર માન્યતા આપે છે કે જે Keyboard થી સીધા ટાઈપ થતા નથી. ઉદાહરણ Back-space, tab વગેરે. આ કેરેક્ટર escape થી દર્શાવાય છે. escape sequence એ એક કેરેક્ટર બતાવે છે. નીચે સામાન્ય escape Sequence નું લિસ્ટ બતાવેલ છે.

escape Sequence	Graphic ના હોય તેવા કેરેક્ટર
\ a	બેલ (બીપ)
\ b	બેકસ્પેસ
\ f	ફોર્મફીડ
\ n	નવી લાઈન
\ r	carriage રીટર્ન
\ t	સીધો ટેબ
\ v	ઊભો ટેબ
\ ?	પ્રશ્નાર્થ ચિહ્ન
\\	બેકસ્લેસ
\'	સીંગલ કવોટ
\ x hh	હેકઝાડેસીમલ નંબર હેકઝાડેસીમલમાં નંબર (hh હેકઝાડેસીમલમાં બતાવે છે.)
\"	ડબલ કવોટ 0
\000	ઓકેટેલ નંબર(00 ઑકટેલ આંક બતાવે છે.)
\0	કંઈ નહી.(Null)

(iii) Floating constant

તેને real constant પણ કહેવાય છે તે નંબર છે જે અપૂર્ણ ભાગ બતાવે છે તે અપૂર્ણાંકની રીતે અથવા exponent ની રીતે લખાય છે.

real constant અપૂર્ણાંકમાં signed અથવા unsigned નંબર કે જેમાં આંકડામાં દશાંશ ધરાવતો હોય તે ધરાવે છે. ઉદાહરણ તરીકે 3.0, -17, 0 , -0.627 વગેરે.

exponent માં બે ભાગ હોય છે. mantissa અને exponent, mantissa intenger હોય છે અથવા real constant હોય છે. જેમાં E અથવા e હોય છે અને exponent કે જેમાં integer જ હોય છે. ઉદાહરણ તરીકે 2EO3, 1.23E07

(iv) string literals :

કરેક્ટરની હરોળ કે જે બે ક્વોટ (double) થી બંધાયેલ હોય તેને String literal કહે છે. સ્ટ્રીંગના અંતમાં પોતાની રીતે ખાસ કેરેક્ટર '0' સાથે ઉમેરાય જાય છે. ઉદાહરણ તરીકે COMPUTER મેમરીમાં "COMPUTER 0" તરીકે ઓળખાય છે અને તે 9 કેરેક્ટરનું કદ ધરાવે છે.

7.5.4 Punctuators

નીચેના કેરેક્ટર C++ માં Puctuators તરીકે વપરાય છે.

બ્રેકેટસ [ ]	ખૂલતો અને બંધ કૌંસ એક અને વિવિધ પરિમાણીય array બતાવે છે.
પેરેન્થીસ ( )	ખુલ્લો અને બંધ કૌંસ ફંક્શન કોલ, ફંક્શન પેરામીટર ગ્રુપના સમીકરણ માટે વગેરે બતાવે છે.
બ્રેસીસ { }	ખુલ્લો અને બંધ કૌસ કંમ્પાઉન્ડ સ્ટેટમેન્ટની શરૂઆત અને અંત બતાવે છે.
કોમા ,	તે ફંક્શનની આર્ગ્યુમેન્ટ લીસ્ટમાં અલગ કરવા વપરાય છે.
સેમીકોલન ;	તે વાક્યની પૂર્ણાહૂતિ માટે વપરાય છે.
કોલન :	તે labelled વાક્ય અથવા કન્ડીશનલ ઓપરેટરની નિશાની છે.
એસ્ટ્રીક *	તે પોઈન્ટર બતાવવામાં અથવા ગુણાકારની નિશાની તરીકે વપરાય છે.

ઈક્વલ સાઈન = તે એસાઈન્મેન્ટ ઓપરેટર તરીકે વપરાય છે.

પાઉન્ડ # તે પ્રી - પ્રોસેસર તરીકે વપરાય છે.

7.5.5 ઓપરેટર (Operators)

Operators એ ખાસ નિશાની છે જે ખાસ કામ માટે વપરાય છે. C++ માં 6 પ્રકારના ઓપરેટર છે.

- એરેથમેટીક ઓપરેટર
- રીલેશન્લ ઓપરેટર
- લોજીકલ ઓપરેટર
- યુનરી ઓપરેટર
- એસાઈમેન્ટ ઓપરેટર
- કન્ડીશનલ ઓપરેટર
- કોમા ઓપરેટર

એરેથમેટીક ઓપરેટર્સ

ઓપરેટર કે જે ગાણિતિક ક્રિયા કરે + , - ,*, / અથવા % આ ક્રિયા માટે બે કે તેથી વધારે ઓપરન્ડ (ઘટક) જોઈએ. માટે આ ઓપરેટરને બાઈનરી ઓપરેટર કહે છે. નીચેનું ટેબલ ગાણિતિક ઓપરેટર બતાવે છે.

ઓપરેટર	અર્થ	ઉદાહરણ	પરિણામ
+	સરવાળો	8 + 5	13
-	બાદબાકી	8-5	3
*	ગુણાકાર	8 * 5	40
/	ભાગાકાર	8/5	1
%	મોડયુલસા/શેષ	8%5	3

સૂચના : એવી કોઈ ઓપરેટર નથી કે જે C++ માં એક્ષપોનન્ટ (exponent) આપે.

રીલેશનલ ઓપરેટર

રીલેશનલ ઓપરેટર બે કિંમત વચ્ચેનો સંબંધ જાણવા વપરાય છે. બધા રીલેશનલ ઓપરેટર બાઈનરી ઓપરેટર છે અને તેથી તે બે સંખ્યા માંગે છે. જયારે સંબંધ ખોટો હોય છે ત્યારે રીલેશનલ ઓપરેટર 0 રીટન કરે છે અને જયારે સાચુ હોય ત્યારે નોન - ઝીરો રીર્ટન કરે છે. નીચેનું ટેબલ રીલેશનલ ઓપરેટર બતાવે છે.

નોંધ

રીલેશનલ ઓપરેટર	અર્થ
<	કરતાં ઓછું
< =	કરતાં ઓછુ અથવા સમાન
= =	બરાબર
>	કરતાં વધારે
> =	કરતાં વધારે અથવા સમાન
! =	સરખા નથી.

ઉદાહરણ - 1.

```
int x = 2; int I =1 ;
int y = 3;
int z = 5 ;
```

નીચેના વાક્ય સાચા છે.

(i) I = = 1

(ii) x < y

(iii) z > y

(iv) y > =1

(v) x ! = 1

(vi) I < = 1

લોજીકલ ઓપરેટર

લોજીકલ ઓપરેટર એક કે તેથી વધારે રીલેશનલ સમીકરણ જોડવા વપરાય છે. નીચેનું ટેબલ લોજીકલ ઓપરેટર બતાવે છે.

ઓપરેટર	અર્થ
!!	OR
&&	AND
!	NOT

NOT *ઓપરેટરને યુનરી ઓપરેટર કહે છે કારણ કે તેને એક જ સંખ્યાની જરૂર પડે છે.*

ઉદાહરણ - 2

int x =5; int z = 9; int y = 7 ;

(x > y) && (z > y)

પહેલું સમીકરણ (x > y) *ઉકેલે છે, ખોટું અને બીજું સમીકરણ* (z > y) *ઉકેલે છે. સાચું માટે*

અંતનું સમીકરણ ખોટું બને છે.

AND ઓપરેશનમાં જો કોઈ એક સમીકરણ ખોટું હોય તો આખું સમીકરણ ખોટુ બને છે.

OR માં જો કોઈ એક સમીકરણ સાચું હોય તો પુરું સમીકરણ સાચુ બને છે. NOT ઓપરેશનમાં માત્ર એક જ સમીકરણની જરૂર પડે છે.

જો સમીકરણ સાચુ હોય તો NOT સાચાને ખોટું અને તેવી જ રીતે તેનાથી ઊલટી ક્રિયા કરે છે.

યુનરી ઓપરેટર

C++ બે યુનરી ઓપરેટર આપે છે જેમાં એક જ વેરીએબલ જરૂરી છે.

ઉદાહરણ - 3

```
a = -50 ;          a  = -b ;
a = + 50 ;         a = +b ;
```

અહીંયા (+) અને (-) નિશાની યુનરી છે કારણ કે તે બે વેરીએબલ વચ્ચે વપરાતો નથી.

એસાઈમેન્ટ ઓપરેટર

એસાઈમેન્ટ ઓપરેટર "=" સમીકરણની કિંમત જમણી બાજુ કિંમત ડાબી બાજુના ચલમાં સ્ટોર કરે છે.

ઉદાહરણ - 4

```
int m = 5 , n = 7 ;
int x,y,z ;
x = y  = z = 0 ;
```

વધારામાં એસાઈન્મેન્ટ ઓપરેટર ઉપર બતાવ્યા મુજબ C++ કમ્પાઉન્ડ એસાઈનમેન્ટ ઓપરેટરને પણ સાથ આપે છે.

C++ બે ખાસ ઓપરેટર '++' અને ' - -' આપે છે. જે વેરીએબલની 1 કિંમત વધારે અને ઘટાડે છે. વધારવાનો/ ઘટાડવાનો ઓપરેટર કોઈ પણ પ્રકારના વેરીએબલ સાથે વાપરી શકાય છે. પરંતુ તે કોઈ અચળ (Constant) સાથે વપરાતો નથી.

આ ઓપરેટરની આગળ રાખીને ઉપયોગ કોઈ પણ ચલ કરે છે, ત્યારે C++ વધારવાની અથવા ઘટાડવાની ક્રિયા તે ચલની કિંમત સાથે કરે છે. ઉદાહરણ માટે નીચેનો કોડ જુઓ.

```
int sum ctr ;
sum = 12 ;
ctr  =  4 ;
sum = sum + ( ++ ctr)
```

તે sum ની કિંમત 15 આપે છે. કારણ કે C++ પહેલાં વધે છે અને પછી તે ઉમેરાય છે. અને 15 કિંમત આવે છે.

એ જ રીતે નીચેનો કોડ,

```
sum = 12;
ctr  = 4 ;
```

sum = sum + (-ctr)

તે sum ની કિંમત 15 આપે છે. કારણ કે પહેલાં ઘટે છે. પાછળની બાજુ જયારે આ ઓપરેટર હોય છે ત્યારે C++ તે ચલની કિંમત પહેલાં સમીકરણમાં મૂકે છે અને પછી ચલની કિંમત વધે અથવા ઘટે છે. ઉદાહરણ માટે નીચેનો કોડઃ

```
sum = 12;

ctr = 4;

sum = sum + ( ctr ++)
```

તે sum કિંમત 16 આપે છે. કારણ કે Ctrl પહેલા સમીકરણમાં વપરાય છે. જે sum ની કિંમત 16 કરે છે અને પછી Ctr ની કિંમત 1 વધે છે. (ctr હવે 5 બને છે.)

સમાન રીતે નીચેનો કોડ,

```
sum =12 ;

ctr = 4 ;

sum = sum + (ctr -- )
```

જે Sum ની કિંમત 16 કરે છે. કારણ કે ctr પહેલાં તેની કિંમત 4 સાથે વપરાય છે જે sum કિમંત 16 આવે છે અને પછી તેની કિંમત 1 ઘટે છે. (ctr 3 બને છે.)

નીચેનો કોઠો જુઓ.

ઓપરેટર	**ઉદાહરણ**	**સમાન છે.**
+ =	A + =2	A = A + 2
- =	A - = 2	A = A -2
% =	A % = 2	A = A % 2
/ =	A / =2	A = A/2
* =	A * =2	A = A * 2

ઉદાહરણ - 5

```
int x = 2;        // first
x + = 5 ;         // second
```

બીજા વાકયમાં x ની કિંમત 5 છે.

કન્ડીશનલ (શરતી) ઓપરેટર

કન્ડીશનલ ઓપરેટરને ટરનરી ઓપરેટર કહે છે અને તેને ત્રણ ચલની જરૂર પડે છે. તેનું માળખું આ પ્રમાણે છે.

કન્ડીશનલ સમીકરણ ?, સમીકરણ 1, સમીકરણ 2;

જો કંડીશનલ સમીકરણ સાચું હોય તો સમીકરણ 1 ઉકેલાશે અથવા સમીકરણ 2 ઉકેલાશે.

ઉદાહરણ - 6

```
int a =5 ;
int b = 6 ;
big = ( a > b) ? a : b;
```

શરત ખોટી છે માટે big *કિંમત* b *પાસેથી લેશે અને* 6 *બનશે.*

કોમા ઓપરેટર : - કોમા ઓપરેટર ડાબીથી જમણી બાજુનું સમીકરણ ઉકેલે છે. એક જ હરોળમાં એક કરતાં વધારે સમીકરણ કોમા ઓપરેટરથી ઉકેલી શકાય છે.

ઉદાહરણ - 7

```
int i = 20, j = 25 ;
int sq = i * i, cube = j * j * j *
```

ઉપરના વાકયમાં કોમા બે સમીકરણ વાકયને અલગ કરવા વપરાય છે.

7.6 સાઈઝ ઓફ ઓપરેટર

આપણે જાણીએ છીએ અલગ - અલગ પ્રકારના વેરીએબલ અચળ વગેરે ને અલગ - અલગ જથ્થામાં મેમરી સંગ્રહ કરવા જોઈએ છે. સાઈઝ ઓફ ઓપરેટર ઓબજેક્ટને કેટલા બાઈટ મેમરી જોઈશે, તેના માટે વપરાય છે.

ઉદાહરણ - 8

```
size of (char) returns 1
size of (int) returns 2
size of (float) returns 4
```

જો K *એક* integer *વેરીએબલ હોય તો* sizeof (K) 2 *આપે છે.*

Sizeof *ઓપરેટર ઓબજેક્ટના રન સમય કરતાં કમ્પાઈલ સમય પર મેમરી કેટલી જોઈશે તે*

નોંધ

જુએ છે.

7.7 ઓર્ડર ઓફ પ્રેસીડન્સ

7.7 જે ક્રમમાં ગાણિતિક ક્રિયા આપેલા સમીકરણમાં થાય છે, તેને પ્રીસીડન્સનો ક્રમ કહે છે. નીચેના કોઠામાં (+,-, *. /, %) નો ક્રમ બતાવેલ છે.

ક્રમ	ઓપરેટર
પહેલો	( )
બીજો	* , / , %
ત્રીજો	+ , -

નીચેના સમીકરણમાં ગણતરી ડાબીથી જમણી બાજુ થાય છે.

ઉદાહરણ - 9

5 + 4/3 * (5 -3)/2

= 5 + 4/3 * 2/2

= 5 + 1 * 2/2

= 5 + 2/2

= 5 + 1

= 6

નીચેના કોઠો ઓપરેટરનો ++, -- પ્રેસીડન્સ બતાવે છે.

++, - - (post increment / decrement)	Highest
++ (Pre increment) - - (Pre decrement), sizeof ( ), ! (not), - (unary), + (unary)	↑
1, /, %	
+, -	
<, <=, >, >=	
= =, ! =	
&&	
? :	
=	↓
Comma operator	Lowest

પાઠગત પ્રશ્નો 7.1

1. **ખાલી જગ્યા પૂરો.**

(અ) કી વર્ડ એ C++ માં............ શબ્દ છે.

(બ) પાસ્કલએ.......... ભાષા છે જ્યારે C++ ....ભાષા છે.

(ક) Identifier ....ની સાથે શરૂઆત ન જ થવી જોઈએ.

(ડ) ....અચળ એ પૂર્ણ સંખ્યા છે જેમાં કોઈ અપૂર્ણાંક ભાગ નથી.

(ઈ) ડબલ ક્વોટાથી બંધાયેલ કેરેક્ટર ક્રમને .... લીટરલ કહેવાય.

2. **ખરાં કે ખોટાની નિશાની નીચેનામાં કરો.**

(અ) આઈડીન્ટીફાયરનું નામ અન્ડરસ્કોર ' ' થી ચાલુ થાય છે.

(બ) ટોકન એ કેરેક્ટરનો સમૂહ છે જે એકબીજાથી જોડાયેલ હોય છે.

(ક) લીટરલ એ ડેટા આઈટમ છે કે જે સામાન્ય રીતે તેની કિંમત પ્રોગ્રામના execution માં બદલાય છે.

(ડ) એસકેપ સીક્વન્સ એ એક કેરેક્ટર બતાવે છે.

(ઈ) # નિશાની પી - પ્રોસેસર ડાઈરેક્ટીવ તરીકે વપરાય છે.

3. C++ માં વપરાતા ટોકનના અલગ અલગ પ્રકારનું લિસ્ટ બનાવો.

4. C++ દ્વારા અપાતા ડેટા ટાઈપનું લિસ્ટ બનાવો.

7.8 ટાઈપની બદલી

પ્રક્રિયા કે જેમાં પહેલાં જાહેર થયેલ સમીકરણ બીજા પ્રકારમાં બદલાવે તેને બદલી કહે છે. C++ માં બે પ્રકારની બદલી થાય છે.

- ઈમ્પ્લીસીટ બદલી
- એકસપ્લીસીટ બદલી

7.8.1 ઇમ્પ્લીસીટ બદલી

ડેટા ટાઈપ સમીકરણમાં ભેગા થઈ શકે છે.

ઉદાહરણ - 10

```
double a ;
int b = 5 ;
float c  = 8.5 ;
a = b * c ;
```

જયારે બે અલગ પ્રકારના ચલ એક સમાન સમીકરણમાં જોડાયેલ હોય ત્યારે લઘુત્તમ પ્રકાર ધરાવતો વેરીએબલ મહત્તમ પ્રકાર ધરાવતા વેરીએબલમાં બદલાય છે. નીચેનો કોઠો ડેટા ટાઈપનો ક્રમ બતાવે છે.

ડેટા ટાઈપનો ક્રમ	
ડેટા ટાઈપ	ક્રમ
	મહત્તમ ↑
long double	
doulbe	
float	
long	
int	↓
char	લઘુત્તમ

int માં b ની કિંમત float માં બદલાય છે અને float C ની કિંમત ગુણાકાર કરતાં પહેલા ટેમ્પરેરી વેરીએબલમાં આવે છે. તેનું પરિણામ પછી double માં આવે છે તેથી તેને double વેરીએબલ a માં રાખવામાં આવે છે.

7.8.2 એક્સપ્લીસીટ બદલી

તેને ટાઈપ કાસ્ટીંગ પણ કહેવામાં આવે છે. તે જાહેર થયેલ ડેટા ટાઈપમાં થોડો સમય બદલાવી નવો ડેટા ટાઈપ આપે છે. એ વાત ધ્યાનમાં રાખો કે ટાઈપ કાસ્ટીંગ માત્ર જમણી બાજુના એસાઈન્મેન્ટ ઓપરેટરમાં જ લાગુ પડે છે.

T _ Pay = double (salary) + bonus ;

શરૂઆતમાં વેરીએબલ salary float તરીકે રજૂ થાય છે, પરંતુ ઉપરની ગણતરી તેને પહેલા double ડેટા ટાઈપમાં ફેરવી અને પછી વેરીએબલ bonus માં ઉમેરે છે.

7.9 Constants કોન્સ્ટન્ટ (અચળ)

પ્રોગ્રામના અમલ (એકસીક્યુશન) વખતે જેની કિંમત ના બદલાય તેને અચળ કહે છે. કોઈ પણ તેની કિંમત બદલાવાનો પ્રયત્ન કરે તો એરર (ભૂલ) મેસેજ આપે છે. C++ માં અચળ કોઈ પણ ડેટા ટાઈપ હોઈ શકે છે. Const ને અચળ તરીકે નીચેની રીતે બતાવાય છે.

const float Pi = 3.1215 ;

ઉપરનો અર્થ એવો થાય કે P i એક અચળ છે. જેને float (તરીકે જાહેર કરેલ છે અને તેની કીંમત 3.1215 છે.

અચળ જાહેર કરવાના માન્ય, ઉદાહરણ :

ઉદાહરણ : -

```
const int rate = 50 ;
const float pi = 3.1415;
const char ch = 'A';
```

7.10 વેરીએબલ

વેરીએબલ એ કોઈપણ કોમ્પ્યુટર ભાષાની જરૂરિયાત છે. તે કોમ્પ્યુટરની મેમરીની એક જગ્યા છે કે જે માહિતીનો સંગ્રહ કરી શકે છે અને સરળતા માટે એક નિશાનીય (ચોક્કસ) નામ આપે છે. વેરીએબલ પ્રોગ્રામના અમલ વખતે અલગ અલગ કિંમત અલગ અલગ સમયે જકડી રાખવા વપરાય છે.

વધારે સારી રીતે સમજવા આપે નીચેનું વાકય જોઈએ.

Total = 20,00; .... (i)

Net = Total - 12.00; .... (ii)

વાકય (i) માં મેમરીની જગ્યા Total પર કિંમત 20.00 આવે છે. વેરીએબલ Total નો ઉપયોગ વાકય (ii) વેરીએબલ Net ની ગણતરી માટે કરે છે. વાકય (ii) માં વાકય તરીકે નહી પરંતુ તેની કિંમતનો ઉપયોગ થાય છે.

પ્રોગ્રામમાં વેરીએબલના ઉપયોગ કરતાં પહેલાં તેને જાહેર કરવો પડે છે. આ પ્રક્રિયાથી કમ્પાઈલર તેને મેમરીમાં યોગ્ય જગ્યા આપવા શકય બનાવે છે. વેરીએબલની વ્યાખ્યા માટે વેરીએબલ તેના નામના પ્રકાર સાથે જાહેર થયેલ હોવું જોઈએ. ઉદાહરણ તરીકે વેરીએબલ Total તેના પ્રકાર Float સાથે નીચેની રીતે જાહેર થાય છે.

float Total ;

એ જ રીતે, વેરીએબલ Net નીચેની રીતે જાહેર થાય છે.

float Net ;

વેરીએબલ માન્ય રીતે જાહેર કરવાના ઉદાહરણ :

(i) int count ;

(ii) int i, j, k ;

(iii) char ch, first ;

(iv) float Total, Net;

(v) long int sal;

7.11 Input/ Output (I/O)

C++ Input/Output વાકયને સપોર્ટ આપે છે, જે નવી માહિતીને કોમ્પ્યુટરમાં અથવા ચોક્કસ આઉટપુટ સાધન જેવા કે VDU પ્રિન્ટર વગેરેમાં લાવવા વપરાય છે. તે બંને સ્ટ્રીમ

નોંધ

એટલે કે ગોઠવેલ અને ગોઠવેલ વગર આપે છે. નીચે C++ ના સ્ટ્રિમ આપેલ છે જે ઈનપુટ/આઉટપુટ માટે વપરાય છે.

સ્ટ્રીમ	માહિતી
cin	કોન્સોલ ઈનપુટ
cout	કોન્સોલ આઉટપુટ

ઉપરના I/O સ્ટ્રીમમાં બે ઓપરેટર << અને >> બંને વપરાય છે. ઓપરેટેર << મૂકવા માટે અથવા બીટ વાઈસ શીફ્ટ ઓપરેટર તરીકે વપરાય છે. જ્યારે ઓપરેટર >> બહાર લાવવા વપરાય છે.

ચાલો આપણે એક પરિસ્થિતિ જોઈએ જેમાં વપરાશકર્તા એક મેસેજ ‘My first Computer સ્ક્રીન પર ડિસ્પ્લે કરાવવા માંગે તો જે નીચેના વાક્ય પરથી મેળવી શકાય.

cout << “My first computer”;

એકવાર જ્યારે ઉપરનું વાક્ય કોમ્પ્યુટર દ્વારા લેવાય છે પછી સ્ક્રીન પર ‘My first computer’ મેસેજ દેખાય છે. એ જ રીતે નીચેનો પ્રોગ્રામ Integer પ્રકારનો વેરીએબલ sum જાહેર કરે છે. જે કીંમત 100 બહાર પાડે છે અને વેરીએબલની માહિતી સ્ક્રીન પર લાવે છે.

```
:
:
int sum
sum = 100
cout <<." the value of variable sum = ";
cout << sum ;
:
:
```

જ્યારે ઉપરનો પ્રોગ્રામ કોમ્પ્યુટર માન્યતા લે ત્યારે નીચેનું આઉટપુટ સ્ક્રીન પર આવે છે.

The value of variable sum = 100

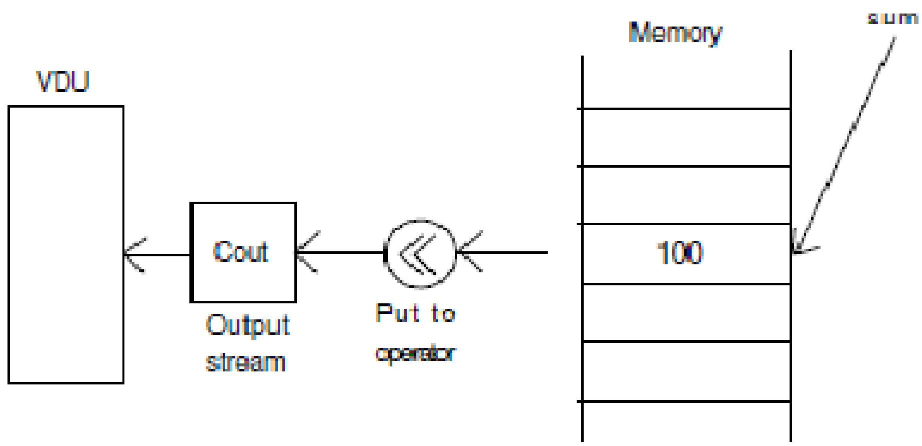
ઉપરની ચર્ચા પરથી આપણે જોઈ શકીએ કે cout એ C++ માં માન્ય (સામાન્ય) આઉટપુટ સ્ટ્રીમ છે જે આઉટપુટ ને સ્ક્રીન પર બતાવે છે. Put ઓપરેટર << ને પણ cout દ્વારા વેરીએબલને માહિતી આપે છે. જે તમે આકૃતિ 7.2. માં જોઈ શકશો.

cin એ સામાન્ય ઈનપુટ સ્ટ્રીમ (કી -બોર્ડના) છે અને તે વપરાશકર્તા દ્વારા કી - બોર્ડની કિંમત ને ઈનપુટ કરવા વપરાય છે. get from >> ઓપરેટર cin દ્વારા ટાઈપ થયેલ કિંમત લે છે. જે તમે આકૃતિ 7.3 માં જોઈ શકશો.

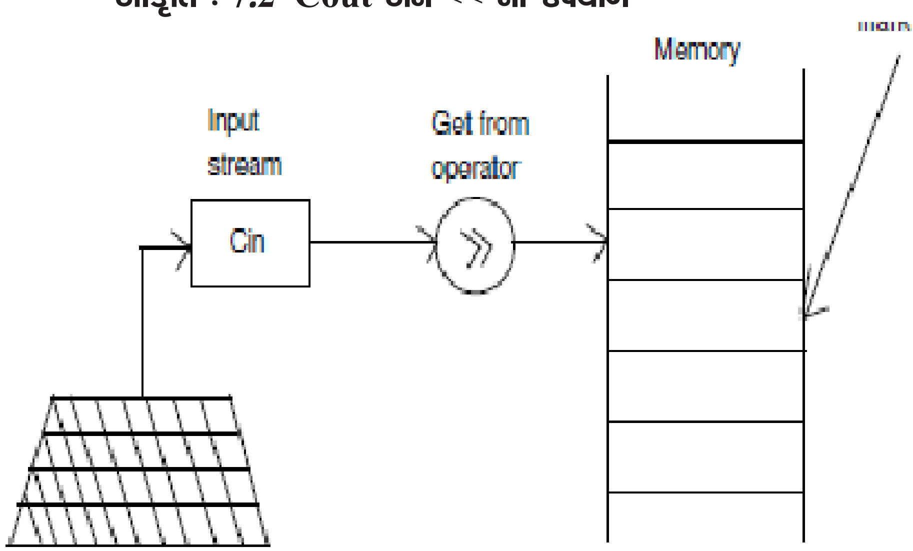
ચાલો આપણે નીચેનો પ્રોગ્રામ ધ્યાનમાં લઈએ.

```
:
int marks ;
:
Cin >> marks ;
:
```

ઉપરના વાક્યમાં વપરાશકર્તા integer પ્રકારનો વેરીએબલ marks પહેલાં વાક્યમા જાહેર કરે છે અને બીજા વાક્યમાં તે કી - બોર્ડ દ્વારા તેની કિંમત લેવાનો પ્રયત્ન કરે છે.



આકૃતિ : 7.2 Cout અને << નો ઉપયોગ



આકૃતિ : 7.3 Cin અને >> નો ઉપયોગ

કોમ્પ્યુટર દ્વારા જે કીબોર્ડથી ટાઈપ કરીએ છીએ ધારો કે 87 તે માન્ય (in) ઈનપુટ સ્ટ્રીમ ને મળે

છે. cin પછી તેને get ઓપરેટરને આપે છે, જે અંતે મેમરી marks તરીકે મુકે છે. વેરીએબલ નો માહિતી સ્ક્રીન પર નીચેના વાકયમાં દેખાય છે.

```
cout << " marks = " ;
cout <, marks ;
```

આપણને એક કરતાં વધારે આઉટપુટ અથવા પુટ (Put) ઓપરેટર એક જ વાકય પર નીચેની રીતે બતાવી શકીએ. આ ક્રિયાને કેસકેડીંગ (cascading) કહે છે.

```
cout << 'Marks = " << marks;
```

વાકયનું આઉટપુટ marks = 87 આવશે.

7.12 C++ પ્રોગ્રામનું માળખું

C++ માં પ્રોગ્રામનું માળખું નીચે મુજબ છે.

```
# include < header file >
main ( )
{
_ _ _ _
_ _ _ _
}
```

C++ પ્રોગ્રામ main ( ) ફંકશનથી ચાલુ થાય છે. ફંકશનનું માળખું બે છગડીયા કૌંસ ({ }) ની વચ્ચે બંધાયેલ હોય છે. કૌંસ એ Pascal ના BEGIN અને END કી વર્ડને સમાન હોય છે. પ્રોગ્રામના વાકય કૌંસમાં લખાય છે. દરેક વાકય સેમીકોલનથી પૂરા થાય છે. (વાકયનું અંત) C++ ના પ્રોગ્રામમાં જરૂર પ્રમાણે ઘણા ફંકશન હોય છે જયારે પ્રોગ્રામ મેમરી પર લોડ થાય છે ત્યારે બધો કાર્યભાર main ( ) ફંકશન લઈ લે છે અને તે ફંકશન પહેલા અમલમાં મુકાય છે. ચાલો, આપણે આપણો પહેલો પ્રોગ્રામ લખીએ.

```
//this is my first program in c++
/* this program will illustrate different component of a simple
program in c ++ */
#  include,< iostream.h>
void main ()
{
cout << 'this is my first programm in c ++" ;
cout << "\n _ _ _ _ _" ;
}
```

જયારે ઉપરનો પ્રોગ્રામ કમ્પાઈલ, લીન્ક અને અમલમાં મૂકાય છે, ત્યારે VDU સ્ક્રીન પર નીચેનું આઉટપુટ દેખાય છે.

this is my first program in C++

"- - - - - "

આ પ્રોગ્રામના ઘણા ઘટકોની નીચે ચર્ચા કરીએ.

(i) Comments

પહેલી ત્રણ લાઈન આ પ્રોગ્રામની કોમેન્ટ છે. જે કમ્પાઈલર દ્વારા ધ્યાનમાં લેવાતી નથી. કોમેન્ટ પ્રોગ્રામમાં તેને વધારે વાંચવા લાયક બનાવવાં વપરાય છે. જો કોમેન્ટ નાની હોય અને એક વાકયમાં આવી જતી હોય તો તેની શરૂઆતમાં ડબલ (બે) સ્લેશ મૂકાય છે. જો વધારે લાઈનની કોમેન્ટ હોય તો તેને નિશાની /* અને */ વચ્ચે મૂકવામાં આવે છે. /* અને */ વચ્ચેનું બધું કમ્પાઈલર ધ્યાનમાં લેતું નથી.

(ii) Include <iostream. h >

ઉપરના પ્રોગ્રામમાં લાઈન કે જે નિશાની '#' થી શરૂ થતી હોય તેને ડ્રાઈરેકટીવ્સ કહે છે અને તે કમ્પાઈલરની સૂચનાઓ છે # ની સાથે જોડાયેલ શબ્દ include કમ્પાઈલરને ઉપરની પ્રોગ્રામમાં iostrea.h ફાઈલ ઉમેરવા જણાવે છે. iostream.h મથાળાની ફાઈલ છે. જે પ્રોગ્રામમાં ઈનપુટ/આઉટપુટની જરૂરિયાતની છે તેમાં આ ફાઈલ પ્રોગ્રામની ઉપર ઉમેરવામાં આવે છે.

(iii) Void main ()

main ( ) એ ફંકશનનું નામ છે. main ની સાથે જોડાયેલ કૌંસ બતાવે છે કે main () ફંકશન છે main ( ) ફંકશનની આગળનો શબ્દ void બતાવે છે કે ફંકશન main ( ) કોઈ કિંમત આપતું નથી. દરેક C++ પ્રોગ્રામ એક કરતાં વધારે ફંકશન ધરાવે છે. જયારે પ્રોગ્રામ મેમરી પર લોડ થાય છે. ત્યારે બધો કાર્યભાર main ( ) ફંકશનને આપે છે અને તે અમલમાં બનનારું પહેલું ફંકશન બને છે.

(iv) છગડીયા કૌંસ અને main ( ) ફંકશનનો ભાગ

C++ પ્રોગ્રામ main ( ) ફંકશનથી ચાલુ થાય છે અને તેનો ભાગ છગડીયા કૌંસથી બંધાયેલા હોય છે. આ કૌંસ Pascal ના BEGIN અને END કી વર્ડને સમાન હોય છે. પ્રોગ્રામના વાકય કૌંસમાં જ લખાય છે. દરેક વાકય સેમીકોલનથી જ પૂરા થવા જોઈએ. તેના વગર એરર મેસેજ લાવી શકે છે.

ઉદાહરણ - 11

C++ પ્રોગ્રામ લખો કે જે બે કિંમત X અને Y ને read કરી તેનો બદલાવી આઉટપુટ બતાવે.

ઉકેલ ધારો કે x અને y int ટાઈપ છે. વેરીએબલ temp int ટાઈપનું થોડા સમય માટેનું સ્થાન બનાવીએ. પ્રોગ્રામ નીચે પ્રમાણે બને.

```
# include < iostream.h>
```

નોંધ

```
main ()
{
        int x,y temp ;
        // _ _ Read the value of x and y
        cout << " Enter the values:";
        cin >> x >> y ;
        // _ exchange the contents
        temp = x ;
        x = y ;
        y = temp;
        // Display the output now
        cout << " the exchange values are:" << x <<y ;
        }
```

પાઠગત પ્રશ્નો 7.2

1. **ખાલી જગ્યા પૂરો.**

   (અ) ....ઓપરેટર બે કિંમત વચ્ચેનો સંબંધ જાણવા (ચેક કરવા) વપરાય છે.

   (બ) C++ બે ....ઓપરેટર આપે છે કે જેમાં એક વેરીએબલ જરૂરી છે.

   (ક) પ્રક્રિયા કે જેમાં એક પહેલાં જાહેર થયેલ સમીકરણ બીજા ટાઈપમાં ફેરવાય તેને .... સમીકરણ કહે છે.

   (ડ)............. કોમેન્ટ ની માહિતીને સમજતું નથી અને તેને ધ્યાનમાં લેતું નથી.

   (ઈ) ઓપરેટર ++ .... તરીકે ઓળખાય છે.

2. **નીચેનાની સામે ખરાં કે ખોટાની નિશાની કરો.**

   (અ) કોમ્પ્યુટરના એરેથમેટીક સમીકરણની કિંમત હંમેશા આંકડાકીય હોય છે.

   (બ) "\n" એ રીમાઈન્ડર ઓપરેટર છે.

   (ક) C++ માં એરેથમેટીક ઓપરેટર માન્ય છે.

   (ડ) AND ઓપરેશનમાં એક ચલ સાચો હોય તો આઉટપુટ સાચું આવે છે.

   (ઉ) + = એ કમ્પાઉન્ડ એસાઈનમેન્ટ ઓપરેટર છે.

3. **નીચેના સમીકરણ ઉકેલો.**
   (a) 5/3* 6
   (b) 6.0 * 5/3
   (c) 6 * 5/3

નોંધ

7.13 તમે જે શીખ્યા

આ પાઠમાં તમે ઓબ્જેક્ટ ઓરીએન્ટેડ પ્રોગ્રામની વિશેષતા ખાસ કરીને C++ ભાષા શીખ્યા આપણે કેરેકટર સેટ ટોકન કે જે વેરીએબલ, અચળ અને ઓપરેટર ધરાવે છે તે સમજયા. C++ માહિતીની જગ્યા બદલી પોતાની રીતે થાય છે, અથવા પ્રોગ્રામર દ્વારા થાય છે. (જેમ કે ઈમ્પ્લીસીટ અથવા એકસપ્લીસીટ કોલ) જે ટાઈપ કન્વરઝનમાં આપેલ છે. પ્રોગ્રામને વધારે વાંચવા લાયક બનાવવા માટે કોમેન્ટ વાકય પણ આપેલ છે. અંતમાં તમે C++ પ્રોગ્રામ લખી શકવા જોઈએ, જે માટે વપરાશકર્તા પાસેથી ઈનપુટ લઈ તેમાં પ્રક્રિયા કરી સ્ક્રીન પર પ્રિન્ટ થાય છે.

7.14 પાઠયાંત પ્રશ્નો

1. *કી વર્ડ અને આઈડીન્ટીફાયર વચ્ચે શું તફાવત છે ?*
2. *નીચેના શબ્દો સમજાવો.*

   (i) *લીટરલ* (ii) *ઈમ્પ્લીસીટી બદલી*
3. *ઈન્ટિઝર અચળને બતાવવા માટે C++ માં કેટલા રસ્તા છે ?*
4. *કોમેન્ટ વાકય માટે નિયમ શું છે ?*
5. *એસકેપ સીકવન્સ શું છે ? કોઈ પણ બે બેકસ્લેસ કેરેકટર અચળ જણાવો.*
6. *વિગતવાર રીતે C++ ઈનપુટ/આઉટપુટ સ્ટ્રીમ સમજાવો.*
7. *નીચેના બીજ સમીકરણને સમાન C++ માં સમીકરણ લખો.*

   (i) $\frac{2AB + 2\,BC + 2\,CA}{2A}$

   (ii) $^{4}/_{3}\ x^{2}$

   (iii) $\frac{b2 - 4ac}{2a}$
8. *નીચેના C++ સમીકરણ ઉકેલો.*

   int a, mb = 2, k =4 ;

   a = mb * 3/4 + K/4 + 8 - mb + 5/8 ;

નોંધ

9. num = 10 અને num = = 10 વચ્ચે શું તફાવત છે ?

10. નીચેના વાક્ય માટે C++ પ્રોગ્રામ લખો :

(i) a, b અને C ની કિંમત read કરવી.

(ii) a અને b ની કિંમત એક લાઈન પર લખવી અને C ની કિંમત બીજી લાઈન પર

(iii) a અને b ની કિંમતને એક લાઈન પર ખાલી જગ્યાથી અલગ પાડવી અને બીજી લાઈન બે ખાલી જગ્યા પછી C ની કિંમત લખવી.

11. આપેલ નંબરનો વર્ગ કરે તેવો પ્રોગ્રામ લખો.

7.15 પાઠગત પ્રશ્નોના જવાબ

1. (અ) આરક્ષિત (બ) સ્ટ્રકચર, ઓબજેકટ, ઓરીએન્ટલ

(ક) અંક (ડ) ઈન્ટિજર

(ઈ) સ્ટ્રીંગ

2. (અ) સાચું (બ) સાચુ (ક) ખોટુ (ડ) સાચું (ઈ) સાચુ

3. C++ માં વપરાતા વિવિધ ટોકનસ

(i) કીવર્ડ

(ii) આઈડેન્ટીફાયર

(iii) લીટરલ

(iv) પંકચ્યુએટર

(v) ઓપરેટર

4. ડેટાટાઈપ

(i) ઈન્ટિજર ટાઈપ (int)

(ii) Floating point ટાઈપ (float)

(iii) Character ટાઈપ (char)

7.2

1. (અ) રીલેશનલ (બ) યુનરી (ક) બદલી (ડ) કમ્પાઈલર (ઈ) વધારો

2. (અ) સાચુ (બ) ખોટુ (ક) સાચુ (ડ) ખોટું (ઈ) સાચુ

3. (અ) 6 (બ) 10.0 (ક) 10

8

OOP ના સામાન્ય ખ્યાલો

8.1 પ્રસ્તાવના

ઓબ્જેક્ટ (પદાર્થ) લક્ષી પ્રોગ્રામિંગ એ પ્રોગ્રામિંગનો અલગ અભિગમ છે. C++ દ્વારા સહાયરૂપ ઓબ્જેક્ટ લક્ષી ટેકનોલોજી સોફ્ટવેર ડેવલોપમેન્ટ માટે તાજેતરની ટેકનોલોજી ગણવામાં આવે છે. તે માહિતી મોડલિંગ માટે તેમજ ડેટા અથવા તર્ક માટે નમૂનારૂપ હોઈ શકે છે.

8.2 હેતુઓ

આ પાઠ ભણ્યા પછી તમે નીચેનાઓ માટે સમર્થ હશો :

- OOP *માટે ઉપયોગી મૂળભૂત ખ્યાલો જાણવા.*
- OOP *દ્વારા પૂરા પાડવામાં આવેલા વિવિધ ફાયદા.*
- OOP *દ્વારા પ્રોગ્રામિંગ એપ્લીકેશનને સમજાવવા.*

8.3 ઓબ્જેક્ટ લક્ષી પ્રોગ્રામિંગ (Object Oriented Programming)

ઓબ્જેક્ટ લક્ષી પ્રોગ્રામિંગ એ પ્રોગ્રામિંગ માટેનો અલગ અભિગમ છે. તે પ્રક્રિયાગત પ્રોગ્રામિંગ અભિગમ દ્વારા જોવા મળતી નબળાઈઓ દૂર કરીને પ્રોગ્રામરને ઉત્પાદકતા વધારવા માટે એક દૃશ્ય સાથે બનાવવામાં આવ્યું છે. ઘણા વર્ષો સુધી ઓબ્જેક્ટલક્ષી ભાષાઓ જેમ કે C++ અને સ્મોલટોક આવ્યા છે અને બજારમાં ખૂબ જ લોકપ્રિય બની રહી છે. આવી ભાષાઓ વિકસાવવા માટેની મુખ્ય જરૂરિયાત પ્રોગ્રામ માટે વધતી જતી માંગ અને જટિલતા હતી.

8.4 મૂળભૂત ખ્યાલો

નીચેના દર્શાવેલા ઓબ્જેક્ટલક્ષી પ્રોગ્રામિંગના ઉપયોગ માટેના મૂળભૂત ખ્યાલો છે

- *ઓબ્જેક્ટો*
- *વર્ગો (કલાસો)*
- *માહિતીમાં તાત્ત્વિક* (Data abstraction)
- *મોડયુલારીટી*
- *વારસો (ઈન્હેરીટન્સ)*
- *પોલિમોર્ફિઝમ*

8.4.1 ઓબ્જેક્ટો (Objects)

તે એક વ્યકિત, એક બેંક એકાઉન્ટ છે, અથવા કોઈ વસ્તુને દર્શાવી શકે છે. જ્યારે પ્રોગ્રામ ચલાવવામાં આવે છે ત્યારે ઓબ્જેક્ટો (વસ્તુઓ) એકબીજા પર સંદેશા મોકલી ક્રિયા - પ્રતિક્રિયા કરે છે. ઉદાહરણ તરીકે, ગ્રાહક અને એકાઉન્ટ બંને એક પ્રોગ્રામની વસ્તુઓ (ઓબ્જેક્ટ) હોય તો પછી ગ્રાહક ઓબ્જેક્ટ એક બેંક સિલક માટે એકાઉન્ટ ઓબ્જેક્ટને સંદેશો મોકલી શકે છે. દરેક ઓબ્જેક્ટમાં માહિતી અને કોડ હોય છે જે ડેટાને જાળવે છે. ઓબ્જેક્ટો અન્ય દરેક ડેટા અથવા કોડ વિગતો ખબર કર્યા વગર એકબીજાનો સંપર્ક કરી શકે છે. તે સંદેશાનો પ્રકાર સ્વીકારવામાં અને તેના દ્વારા પરત પ્રતિભાવ જાણવા માટે પૂરતો છે.

8.4.2 કલાસીસ (વર્ગો)(Classes)

આપણે હમાણાં જ ઉલ્લેખ કર્યો કે ઓબ્જેક્ટો માહિતી અને ફંકશન અથવા કોડ ધરાવે છે જેનાથી ડેટા જળવાય છે એક વર્ગની મદદથી વપરાશકર્તા દ્વારા એક ઓબ્જેક્ટની સમગ્ર માહિતી મેળવી શકાય છે. હકીકતમાં ઓબ્જેક્ટ એ કલાસ (વર્ગ) નો ચલ છે. જ્યારે એક વખત વર્ગ વ્યાખ્યાયિત કરવામાં આવે છે, ત્યાર પછી આપણને જોઈએ તેટલા વર્ગ સાથે સંકળાયેલ ઓબ્જેક્ટ બનાવી શકાય છે. ઉદાહરણ તરીકે, કેરી, સફરજન અને નારંગી એક કલાસ (વર્ગ) ફળના સભ્યો છે. જો ફળ એક કલાસ (વર્ગ) છે તો પછી તે નિવેદન ફળ કેરી માટે વ્યાખ્યાયિત કરવામાં આવે છે, તેથી ફળ વર્ગ સાથે જોડાયેલા એક ઓબ્જેક્ટ કેરી માટે પણ ઓબ્જેક્ટ બનાવશે.

8.4.3 માહિતીમાં તાત્ત્વિક (Data Abstraction)

પૃષ્ઠભૂમિ (બ્રેકગ્રાઉન્ડ) ની વિગતો વિના જરૂરી સુવિધાઓને રજૂઆત કરવાની પ્રકારને એબસ્ટ્રેકશન કહેવાય છે. વધુ, સ્પષ્ટ રીતે આ ખ્યાલ સમજવા માટે 'સ્વીચ બોર્ડ' નું ઉદાહરણ લો. તમે માત્ર તમારી જરૂરિયાત મુજબ ખાસ સ્વીચો દબાવો. તમે આ સ્વીચો નું આંતરિક કામ જાણવાની જરૂર નથી. સ્વીચની અંદર જે થઈ રહયું છે, તે તમારાથી છુપાયેલું છે. જયાં તમે આવશ્યક વસ્તુઓ પૃષ્ઠભૂમિની વિગતો જાણ્યા વગર સ્વીચબોર્ડ પર કામ કરો, તેને એબસ્ટ્રેકશન (માહિતીમાં તાત્વિક) કહેવાય છે.

8.4.4 માહિતી પ્રાવૃત (Data encapsulation)

માહિતી પ્રાવૃત OOP માટે એક મૂળભૂત ખ્યાલ છે. તે એક એકમ હેઠળ છે, જેથી બંને માહિતી તેમજ કાર્યોના સંયોજન માર્ગ બને છે. માહિતી જાણવા માટે એક જ રસ્તો છે, (માહિતી સાથે જોડવામાં આવે છે.) તેને વિધેયો દ્વારા પૂરી પાડવામાં આવે છે. આ વિધેયો C++ માં સભ્યો તરીકે ગણવામાં આવે છે. તે માહિતીને સીધી જ જાણવા શક્ય નથી. જો તમે એક ઓબ્જેકટમાં માહિતી સુધી પહોંચવા માંગો છો, તો તમે સભ્યના કાર્ય દ્વારા ઓબ્જેકટ સાથે જોડાઈ શકો છો. તે માહિતી વાંચશે અને તમને કિંમત પાછી મોકલશે. માહિતી ગુપ્ત છે, તેથી તે આકસ્મિક ફેરફારથી સુરક્ષિત ગણવામાં આવે છે. માહિતી અને તેના કાર્યો એક એન્ટિટીમાં સમાયેલી હોવાનું કહેવાય છે.

8.4.5 મોડયુલારીટી (Modularity)

વ્યક્તિગત ઘટકોના એક કાર્યક્રમ દ્વારા પ્રોગ્રામને વહેંચવાના અધિનિયમને મોડયુલારિટી કહેવામાં આવે છે. તે નીચેના લાભો આપે છે.

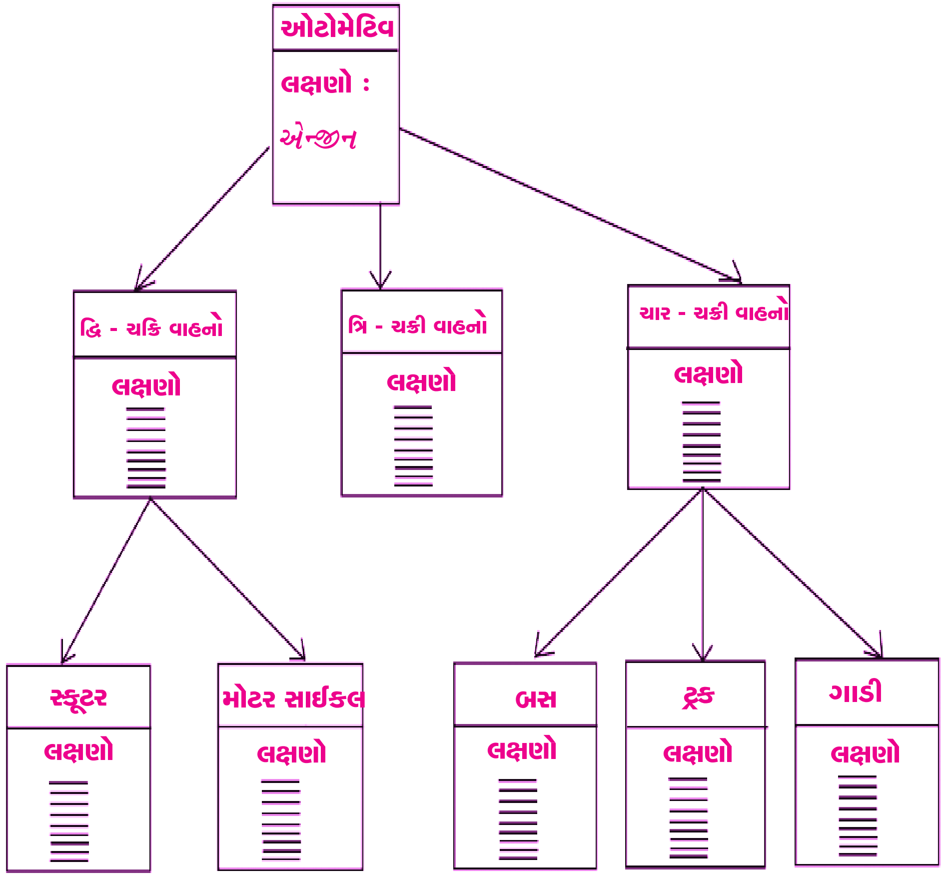
- *તે કેટલાક અંશે તેની જટિલતા ઘટાડે છે.*
- *તે પ્રોગ્રામની અંદર સારી રીતે વ્યાખ્યાયિત, દસ્તાવેજીકરણ સીમાઓ (સંખ્યા) બનાવે છે.*

મોડયુલ પોતે એક અલગ એકમ છે. તે અન્ય મોડયુલો સાથે સંકળાયેલો છે. છતાં સ્વતંત્ર સંકલિત કરી શકાય છે. મોડયુલો પ્રોગોમનો ધ્યેય હાંસલ કરવા માટે સાથે મળીને કામ કરે છે.

8.4.6 વારસો (Inheritance)

તે હાલના વર્ગનો ઉપયોગ કરી એક નવો વર્ગ વ્યાખ્યાયિત કરવા માટે સક્ષમ છે. હાલનો વર્ગ આધાર વર્ગ તરીકે ઓળખવામાં આવે છે અને નવો વર્ગ ઉતરી આવેલ વર્ગ તરીકે ઓળખાય છે. ઘણા બધા ઉદાહરણો આ પાસા પર આપી શકાય છે. ઉદાહરણ તરીકે, એક મોટર સાઈકલ પોતે એક વર્ગ છે. તે બે પૈડાવાળા વાહનોનો વર્ગ ઓટોમોટિવ વર્ગસભ્ય પણ છે. ઓટોમોટિવ આધાર વર્ગ એક ઉદાહરણ છે અને બે પૈડાવાળા વાહનનો વર્ગ ઓટોમોટિવ વર્ગસભ્ય પણ છે. જે આકૃતિ. ૮.૧ માં બતાવ્યું છે ઓટોમોટિવ આધાર વર્ગ છે જયારે બે પૈડાંવાળા વાહનો તેમાંથી તારવેલો વર્ગ છે. સરળ શબ્દોમાં આપણે કહી શકીએ કે મોટર સાઈકલ દ્વિચકી વાહન ઓટોમોટિવ છે.

C++ *અધિક્રમિક વર્ગીકરણ જેવા વર્ગોને આધાર આપે છે. વારસા* (Inheritance) *નો મુખ્ય લાભ એ છે કે તેના દ્વારા સામાન્ય આધાર વર્ગ મેળવી શકાય છે જમે, કે હાલના વર્ગ* (Class) *માટે કેટલીક નવી સુવિધાઓ ઉમેરી શકાય છે. બધા જ નવા વર્ગો તે રીતે વ્યાખ્યાયિત થયેલા છે કે જેમાં બંને વર્ગોના લક્ષણ સમાવેશ થાય છે. વારસા* (Inheritance) *દ્વારા હાલના વર્ગોનો નવી એપ્લિકેશનમાં જરૂરિયાત મુજબ ફેરફાર વગર સ્વીકારી શકાય છે.*



આકૃતિ. 8.1

8.4.7 પોલિમોર્ફિઝમ (Polymorphism)

પોલિમોર્ફિઝમ OPP ની શક્તિ પાછળનું મુખ્ય કારણ પોલિમોર્ફિઝમ છે. તે માહિતીના એક કરતાં વધુ ફોર્મ પ્રક્રિયા કરવા માટે આધાર આપે છે. ઉદાહરણ તરીકે, એક ક્રિયા જુદા - જુદા ઘટકોને અલગ - અલગ વર્તન પ્રદર્શન કરી શકે છે. આ વર્તન ઓપરેશનમાં ઉપયોગમાં લીધેલી માહિતી પ્રકારો પર આધારિત છે. બે નંબરોના સરવાળા માટેની ક્રિયા ધ્યાનમાં લઈએ. બે નંબરો દ્વારા સરવાળાની રકમ પેદા કરશે. જો આ સમીકરણો (Operands) શબ્દમાળાઓ (String) હોય તો પછી આ કામગીરી જોડાણ દ્વારા ત્રીજી શબ્દમાળા પેદા કરશે.

પાઠગત પ્રશ્નો : 8.1

---

1. *નીચેના કિસ્સામાં સાચું અથવા ખોટું જણાવો.*

   (અ) પ્રક્રિયાલક્ષી પ્રોગ્રામિંગમાં તમામ માહિતી બધા વિધેયો દ્વારા વહેંચાય છે.

   (બ) OPP નું એક ખાસ લક્ષણ છે કે તે વાસ્તવિક વિષયને ધ્યાનમાં રાખીને પ્રોગ્રામનું ઓબ્જેક્ટમાં વિભાજન કરે છે.

   (ક) રેંપીંગ ઉપર એક એકમના વિવિધ પ્રકારના ડેટાને પ્રાવૃત્ત (Encapsulation)

તરીકે ઓળખાય છે.

(ડ) ઓબ્જેક્ટલક્ષી કાર્યક્રમો પરંપરાગત કાર્યક્રમો કરતાં ખૂબ ઝડપથી ચલાવવામાં આવે છે.

(ઈ) C એ C++ નો ઉપગણ છે, કારણ કે C ના કાર્યક્રમો C++ ના કમ્પાઈલરોના હેઠળ ચાલશે.

2. એનકેપ્સ્યુલેટેડ (પ્રાવૃત્ત) શા માટે સલામત ગણવામાં આવે છે ?

3 માહિતીને છુપાવવું અને ઈનકેપ્સ્યુલેશન વચ્ચેનો તફાવત કરો.

8.5 OOP ના લાભો

OPP કાર્યક્રમ ડિઝાઈનર અને વપરાશકર્તા બંને માટે ઘણા લાભો પૂરા પાડે છે. ઓબ્જેક્ટલક્ષી અભિગમ સોફટવેર વિકાસ અને સોફટવેર ઉત્પાદન ગુણવતા સાથે સંબંધિત અનેક સમસ્યાઓ ઉકેલવા મદદ કરે છે. નવી ટેકનોલોજી વધારે પ્રોગ્રામર ઉત્પાદકતા, સોફ્ટવેરની સારી ગુણવત્તા અને ઓછી જાણવણી ખર્ચની ખાતરી આપે છે. તેના મુખ્ય લાભો નીચે મુજબ છે.

- સોફ્ટવેર જટિલતા સરળતાથી સંભાળી શકાય છે.
- ઓબ્જેક્ટલક્ષી સીસ્ટમો સરળતાથી સુધારી શકાય છે.
- તે વસ્તુઓ પર આધારિત પ્રોજેક્ટમાં કામની વહેંચણી કરવા માટે ખૂબ જ સરળ છે.

8.6 OOP ના પ્રોગ્રામિંગ કાર્યક્રમો (એપ્લિકેશનો)

OOP આજે પ્રોગ્રામિંગ માટે લોકપ્રિય બની ગયું છે. સોફ્ટવેર પ્રોગ્રામરો OOP વાપરતી ઉત્તેજના અને રસ અનુભવે છે OOP ના કાર્યક્રમોને (એપ્લિકેશનનો) ઘણા વિસ્તારોમાં મહત્વ પ્રાપ્ત થયું છે. OOP વિન્ડો અને શબ્દ આધારિત જેમ કે, એમ.એસ.વિન્ડોઝ, એક. - વિન્ડોઝ વગેરે. સીસ્ટમોના વિકાસ માટે વ્યાપક ઉપયોગ કરવામાં આવી રહ્યો છે. OOP માટેના આશાસ્પદ એપ્લીકેશન વિસ્તારો નીચેના છે.

1. **બહુવિધી માહિતી બંધારણ :** આ એક એપ્લીકેશન છે જયાં માહિતી માળખું ઘણી વખત ઉપયોગ કરવામાં આવે છે. ઉદાહરણ તરીકે એક કરતાં વધુ વિન્ડો માહિતી માળખાનો ઉપયોગ વિન્ડોને લગતી સીસ્ટમ માટે થાય છે.

2. **ડેટા બહુવિધ પ્રોગ્રામો :** આ કામગીરી વિવિધ પ્રોગ્રામો માટે માહિતી બંધારણ પર કરવામાં આવે છે કે જયાં એક એપ્લિકેશન છે. ઉદાહરણ, તરીકે એક એકાઉન્ટીંગ સિસ્ટમ માટેની રેકોર્ડ માન્યતા.

   OOP ની અન્ય એપ્લિકેશનોમાં સમાંતર પ્રોગ્રામિંગ, સિમ્યુલેશન અને મોડેલિંગ, કૃત્રિમ અને નિષ્ણાંત સિસ્ટમો, ન્યુરલ નેટવર્કસ અને CAD સિસ્ટમો છે.

પાઠગત પ્રશ્નો : 8.2

1. ખાલી જગ્યા પૂરો.

(અ) સ્મોલ ટોક એક .... ભાષા છે.

(બ) .... એક સ્વંય એકમ છે તેમજ .... તેનું એક ફંકશન (કાર્ય) છે.

(ક) .... તેના પોતાના પ્રકારની વસ્તુઓ બનાવવા માટે વાપરી શકાય છે.

(ડ) .... એક ઓબ્જેકટમાં વિધેયો સાથે માહિતી જોડવા માટેનો રસ્તો છે.

(ઈ) એક તારવેલો વર્ગ (Class) .... વર્ગમાંથી તારવી શકાય છે.

2. **નીચેનામાંથી ખરુ છે કે ખોટું તે ટીક કરો.**

(અ) C++ એ C પ્રોગ્રામિંગ ભાષા માટે એક એક્સટેન્શન છે.

(બ) C++ માં એક ઓબ્જેકટના કાર્યો (Functions) બીજા ઓબ્જેકટના કાર્યો એકસેસ કરી શકતાં નથી.

(ક) ઈનહેરીટન્સ (વારસો) હાલના Class (વર્ગ) માંથી એક નવો વર્ગ વ્યાખ્યાયિત કરવા માટેની ક્ષમતા છે.

(ડ) એબ્સ્ટ્રેકશન (ડેટાને છુપાવવો) જરૂરી સુવિધાઓ માટેનું પ્રતિનિધિત્વ ઉલ્લેખ પૃષ્ઠભૂમિ માહિતી વિના કરે છે.

(ઈ) એક ઓબ્જેકટ નવી માહિતીનો પ્રકાર છે.

3. **તફાવત રજૂ કરો.**

(અ) પ્રક્રિયાગત અને ઓબ્જેકટલક્ષી પ્રોગ્રામિંગ

(બ) પદાર્થ (Object) અને વર્ગ (Class)

8.7 તમે જે શીખ્યા

આ પાઠમાં તમે સોફ્ટવેર વિકાસમાં, ઓબ્જેકટ લક્ષી ટેકનોલોજી વિશે તાજેતરની ટેકનોલોજી શીખ્યા. OPP ની વિવિધ લાક્ષણીકતાઓ કોમ્પ્યુટર વ્યવસાયીઓના લાભ માટે સ્પષ્ટતા કરવામાં આવી છે. OPP ના આશાસ્પદ અરજી વિસ્તારોનું (એપ્લિકેશનમાં) પણ સંક્ષિપ્ત વર્ણન કરવામાં આવ્યું છે.

8.8 પાઠયાંત પ્રશ્નો

1. OPP ના લક્ષણો શું છે ?

2. નીચેના શબ્દો વિગતવાર સમજાવો.

(અ) માહિતીમાં તાત્વિક

(બ) ડેટા ઈનકેપ્સયુલેશન

(ક) પોલીમોર્ફિઝમ

(ડ) વારસો (Inheritance)

3. OOP ના વિવિધ લાભો સંક્ષિપ્તમાં વર્ણવો.

8.9. પાઠગત પ્રશ્નોના જવાબ

પાઠગત પ્રશ્નો - 8.1

1. (અ) F (બ) T (ક) F (ડ) F (ઈ) T

2. ફક્ત અધિકૃત કાર્યો સંબંધિત માહિતીને મેળવવાની ખાતરી કરવાનું કામ ઈનકેપ્સ્યુલેશન કરે છે. અનધિકૃત એકસેસ સામે માહિતી (Data) ની સલામતી સુનિશ્ચિત કરે છે.

3. જ્યારે આંતરિક માળખાનો કાર્યક્રમ બાકીનામાંથી છુપાયેલો હોય તો તે ડેટા હાઈડિંગ (ડેટાને છુપાવવો) ની લાક્ષણિકતા છે. બીજી બાજુ માહિતી ઈન્કેપ્સ્યુલેશન દ્વારા માહિતી અને તેથી સાથે સંકળાયેલી વિધેયો માત્ર સંબંધિત જાણકારી બહાર આવે છે. જેમ કે એક ઓબ્જેક્ટ વ્યાખ્યા સાથે બંધાયેલો છે કે જેના દ્વારા પધ્ધતિ અને બાકીના ઓબ્જેક્ટ છુપાયેલા રહે છે. આમ માહિતી ઈન્કેપ્સ્યુલેશન માહિતી છુપાવવાનો અમલ કરી રહ્યું છે.

પાઠગત પ્રશ્નો - 8.2

1. (અ) OOP (બ) ઓબ્જેક્ટ માહિતી
   (ક) વર્ગ (ડ) ઈનકેપ્સયુલેશન (ઈ) આધાર

2. (અ) T (બ) F (ક) T (ડ) T (ઈ) F

3.

(અ) એક પ્રક્રિયાગત ભાષામાં કાર્યક્રમ (પ્રોગ્રામ) કોમ્પ્યુટરને કંઈક કરવા માટે સૂચનો કરે છે. ઈચ્છિત ગણતરી કરવા માટે જરૂરી અલ્ગોરિધમની ધ્યાન પ્રક્રિયા ઉપર છે. પ્રક્રિયાગત ફેરફારમાં કામ કેવી રીતે કરવા પર ભાર હોય છે. ઓબ્જેક્ટલક્ષી અભિગમ સમસ્યા પ્રક્રિયા કરવાને બદલે સામેલ વસ્તુઓની દ્દષ્ટિએ જુએ છે.

(બ) એક ઓબ્જેક્ટ કેટલાક લક્ષણો તેમજ વર્તન દ્વારા ઓળખાતી એન્ટિટી છે. તે એક એન્ટિટે છે જેમાં ડેટા અને તેને લગતા કાર્યો સંકળાયેલા હોય છે.

એક કલાસ (વર્ગ) ઓબ્જેક્ટોનું જૂથ છે જે સામાન્ય ગુણધર્મો અને સંબંધો રજૂ કરે છે.

9

નિયંત્રણ નિવેદનો

9.1 પરિચય

ઊંચા સ્તરની ભાષામાં અમલ સામાન્ય રીતે ક્રમાંકિત હોય એટલે કે, દરેક વિધાન પ્રોગ્રામમાં તેના ક્રમમાં ચલાવવામાં આવે. જો કે, તેના પર આધાર રાખીને સમસ્યાની જરૂરિયાત પ્રમાણે તેનો ક્રમ સામાન્ય રીતે બદલી શકાય છે.એવા વિધાનને કે જેમાં પ્રશ્નોનો અમલનો ક્રમ સ્પષ્ટ જણાવ્યો હોય તેને નિયંત્રણ પ્રવાહના નિવેદનો કહેવામાં આવે છે. આ પાઠમાં આવા ઘણા નિયંત્રણો પ્રવાહની ચર્ચા કરવામાં આવી છે.

9.2 ઉદેશો

આ પાઠના વાંચન કર્યા પછી તમે સક્ષમ હશો :

- *એક નિવેદનમાં અને સંયોજન નિવેદન વચ્ચેનો તફાવત આપવા.*
- *શરતી નિવેદનો અંગેની સમજણ મેળવવા.*
- loop *નિવેદન સાથે પરિચિત થશો.*
- do-while *અને* do *વચ્ચેનો તફાવત આપવા.*
- Jump *નિવેદનો ઓળખવા*
- program *માંથી બહાર કઈ રીતે નીકળવું તે સમજાવવા.*

9.3 નિવેદનો

Statements *એ સૂચના છે, જે કોમ્પ્યુટરને આપવામાં આવે અને જેના કારણે કોઈપણ પ્રકારની ક્રિયા થઈ શકે છે.ક્રિયા એ* data *ના હલનચલન, અને નિર્ણય, વગેરે હોઈ શકે. પ્રોગ્રામ સ્ટેટમેન્ટ* C++ *માં નાનામાં નાનો એકમ છે.* Statements *હંમેશા સેમીકોલન* (;) *થી બંધ થાય છે.*

9.4 કંમ્પાઉન્ડ નિવેદન (Compound Statement)

એક સંયોજન નિવેદનમાં એક જૂથ છે કે જે દરેક વ્યક્તિગત નિવેદનમાં અર્ધવિરામ સાથે અંત થાય છે. નિવેદન જૂથ બ્લોક કહેવાય છે. Compound statements *બે કૌંસની વચ્ચે બંધ થાય છે* ({ }). *શરૂઆતનું* bracet ({) *શરૂઆત સૂચવે છે અને બંધ* bracet (}) *બંધ સૂચવે છે.*

9.5 ખાલી નિવેદન (Null Statement)

ફક્ત અર્ધવિરામ લખવાથી ખાલી (null) statement *સૂચવે છે. આમ,* ';' *'એક ખાલી વિધાન સૂચવે છે. આ ભાષાના સિન્ટેક્ષ સ્પષ્ટ કરવાના હોય ત્યારે ખૂબ ઉપયોગી નીવડે છે. પરંતુ કાર્યક્રમના તર્ક માટે કોઈ નિવેદન જરૂરી નથી. આ નિવેદન મોટા ભાગે* for loop *માં ઉપયોગી થાય છે.*

9.6 શરતી નિવેદન (Conditional Statements)

ક્યારેક પ્રોગ્રામ કોઈ ચોક્કસ સ્થિતિ પર આધાર રાખીને ચલાવવાની જરૂર પડે છે. C++ *નિયંત્રણ અમલીકરણ માટે નીચેના નિવેદનો પૂરા પાડે છે.*

(i) 'if ' *વિધાન*

(ii) 'if else' *વિધાન*

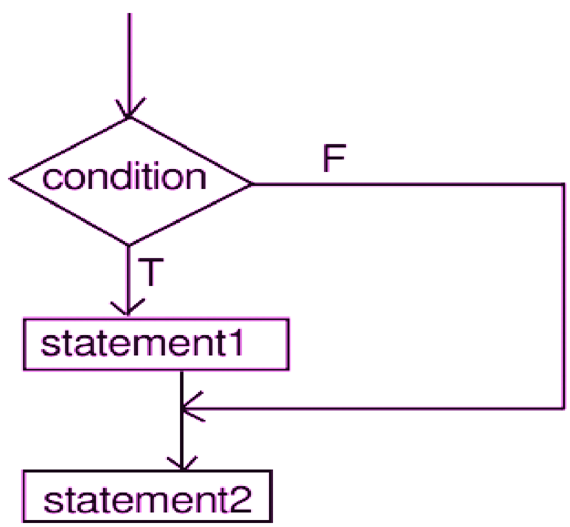
(iii) 'nested if' *વિધાન*

(iv) 'switch' *વિધાન*

'if' statement (વિધાન)

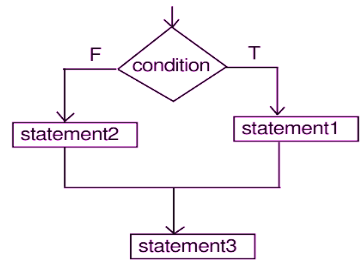
'if' *ની વાક્ય રચના*

```
if (condition)
{
statement(s);
}
```



આકૃતિ. ***9.1***

'આ ફલોચાર્ટ પરથી કહી શકાય કે જો If condition' 'સાચી હોય છે, તો statement 1 ચલાવવામાં આવે છે. અન્યથા તે છૂટી જાય છે. statement 1 એક અથવા વધારે નિવેદન હોઈ શકે છે.



આકૃતિ. 9.2

ઉદાહરણ 1

'if else' *વિધાન*

if- else વચ્ચેની વાકયરચના

```
if (condition)
statement 1;
else
statement 2;
```

ઉપરના ફલોચાર્ટ ઉપરથી આપેલ condition નું પ્રથમ મૂલ્યાંકન કરવામાં આવે છે જે સ્પષ્ટ છે. જો condition સાચી હશે તો statement1 ચલાવવામાં આવશે અને ત્યારબાદ statement3. જો condition ખોટી હશે તો statement2 ચલાવવામાં આવશે અને ત્યારબાદ statement 3. એક વસ્તુનું ધ્યાન રાખવું જરૂરી છે કે statement1 અને statement2 એક અથવા સંયોજન નિવેદન બની શકે છે.

ઉદાહરણ 2

એવા પ્રોગ્રામ લખો કે જે બે નંબરમાંથી મોટો નંબર છાપશે.

```
# include <iostream.h>
void main ()
{
int x, y;
cout << "Enter two numbers" <<"\n";
cin>> x >> y ;
if ( x > y)
cout << "The bigger number is " << X;
else
cout << "The bigger number is" <<y;
}
```

ઉદાહરણ 3

એવો program લખો કે જે 1 થી 10 નંબરની વચ્ચેનો એકી અને બેકી સંખ્યાનો સરવાળો કરી ગણતરી દર્શાવે.

```
# include <iostream.h>
void main ()
{
int sum1, sum2;
```

```
sum1 = sum2 = 0;
for (int i =1; i < = 10; i++)
if (i % 2 = = 0)
sum 1 + = i ;
else
sum2+ = i ;
cout << "sum of odd number =" << sum2 << "\n";
cout << "sum of even number = "<< sum1 << "\n";
}
```

ઉદાહરણ 4

એવો પ્રોગ્રામ લખો કે જે ત્રણ નંબરો માંથી સૌથી મોટો નંબર દર્શાવે.

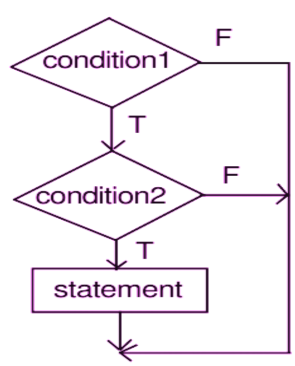
```
# include <iostream.h>
void main ()
{
int x, y, z, l;
cout << "Enter three numbers" <<"\n";
cin >> x >> y>>z;
if (x > y )
l = x ;
else
l = y ;
if (l > z)
cout << "The greatest number is " << l;
else
cout << "The greatest number is "<<z;
}
```

'Nested if' વિધાન

nested if *વિધાનની વાકયરચના*

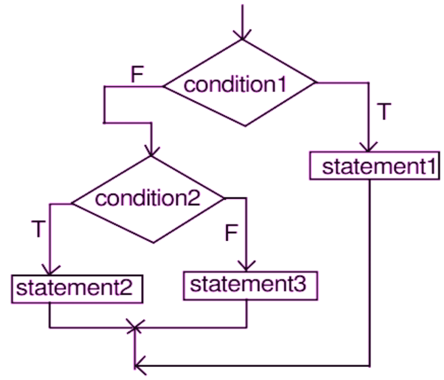
(i) if *બ્લોક બીજા અથવા અન્ય બ્લોકમાં* nested *કરી શકાય છે આને* if *અથવા* else *બ્લોકનું* nesting *કહે છે.*

```
If (condition1)
{
if (condition2)
{
statement(s);
}
}
```



આકૃતિ.9.3

(ii) if (condition 1)
statement1;
else if (condition2)
statement2;
else statement 3;



આકૃતિ.9.4

ઉપરના ફ્લોચાર્ટ ઉપરથી સ્પષ્ટ થાય છે કે condition 1 અને condition2 ક્રમ પ્રમાણે nested છે કે જ્યાં સુધી condition નું મૂલ્યાંકન સાચુ થાય. ફક્ત એક જ statement ચલાવવામાં આવશે. જો એક પણ condition સાચી ના પડે તો statement 3 ને ચલાવવામાં (execute) આવશે.કંન્ટ્રોલને statement 3 પછી આવતા statement પર પાસ કરવામાં આવશે. અંતમાં દરેક વિધાન એક અથવા સંયોજક વિધાન છે.

ઉદાહરણ 5

એવો પ્રોગ્રામ લખો કે જે 3 નંબરમાંથી મોટો નંબર શોધી કાઢે.

```
    # include < iostream.h>
    void main ( )
    {
    int x, y, z;
    cout << "Enter three numbers" << "\n";
    cin >> x >> y >> z ;
    if ( x > y )
    {
    if (x >z)
    cout << "The greatest number is " << x;
    else
    cout << "The greatest number is "<<z;
    }
    else
    {
    if (y > z)
    cout << "The greatest number is "<< y;
    else
    cout "The greatest number is " << z;
    }
}
```

ઉદાહરણ 6

એવો પ્રોગ્રામ લખો કે જેમાં nested if *ની મદદથી માર્ક્સ નાંખો અને તેના આધારે નીચેના પ્રમાણે* grade *મળે.*

Mark	**Grade**
> = 90	A
> = 80 and < 90	B
> = 70 and < 80	C
> = 60 and < 70	D
< 60	Fail

```
# include <iostream.h>
void main ( )
{
int marks;
cout << "Enter total marks" << "\n";
cin >> marks;
if (marks > = 90)
cout << "A Grade";
else
if (marks > = 80)
cout << "B Grade";
else
if (marks > = 70)
cout << "C Grade";
else
if (marks > = 60)
cout << "D Grade";
else
cout << "Failed";
}
```

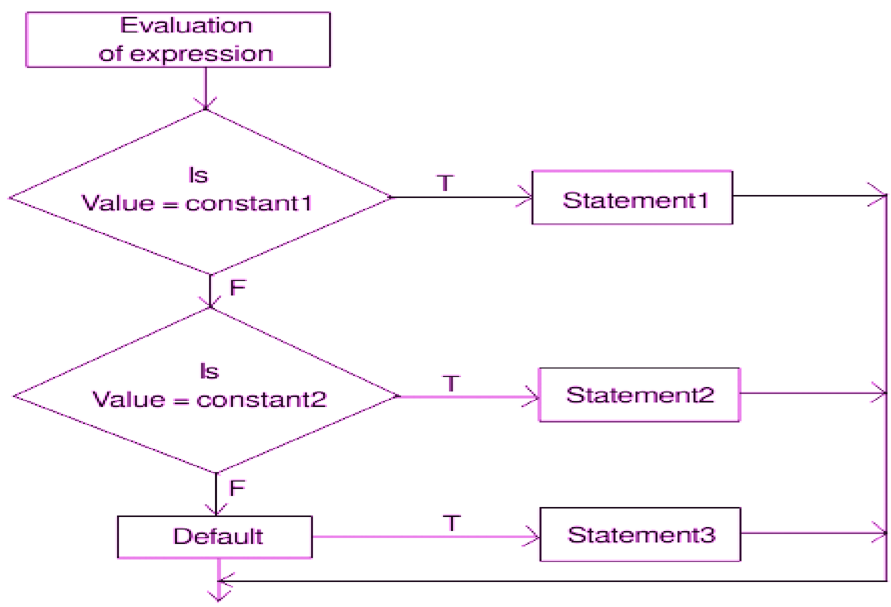
switch વિધાન

if *અને* if - else *ફક્ત* 2 *રીતે* branching *ની પરવાનગી આપે છે, જ્યારે* switch *વિધાનને ઘણી બધી* (multiple) *રીતે* branching *ની પરવાનગી આપે છે.*

switch વિધાનની વાક્યરચના

```
switch (var / expression)
{
case constant1 : statement1;
                   |
                   |
                   |
                   break;
case constant2 : statement2;
                   |
                   |
                   |
                   break;
default : statement3;
}
```

switch *વિધાનનો અમલ* expression *ના મૂલ્યાંકન સાથે શરૂ થાય છે. જો* expression *ની કિંમત* constant *સાથે મેળ ખાય તો તે વિધાન ચલાવવામાં આવે છે અને આ પ્રમાણે* break *નો અમલ* (execute) *થાય ત્યાં સુધી અનુક્રમે ચલાવો.* Break *વિધાનને કારણે* Control, Switch *વિધાનની બહાર આવી. જો* expression *ની* value *કોઈ પણ* constant *ને નહી મળે તો* default *ના વિધાનને ચલાવવામાં આવે છે.*



આકૃતિ .9.5

ઉદાહરણ *7*

```
# include < iostream.h>
void main ( )
{
char ch;
```

```
    cout << "Enter a character";
    cin >> ch;
    switch (ch)
    {
    case 'a' : cout << "vowel a";
                break ;
    case 'e' : cout << "vowel e";
                break ;
    case 'o' : cout << "vowel o";
                break ;
    case 'i' : cout << "vowel i"
                break ;
    case 'u' : cout << "vowel u";
                break;
    default : cout <<"not a vowel";
    }
    }
```

program*માં* break *વિધાનની જરૂરીયાત એ* switch *વિધાનના ચોક્કસ ભાગને ચલાવવા માટે થાય છે. જો ઉપરના* program *માંથી* break *વિધાન નાબુદ કરવામાં આવે તો* control, case *વિધાન સાથે દરેક* statement *ને ચલાવશે અને આ ક્રમ છેલ્લા* case *તથા* default *સુધી ચાલશે. ઉપરના* switch *વિધાનને નીચે પ્રમાણે પણ લખી શકાય છે :*

```
switch (ch)
{
    case 'a' :
    case 'A' : cout << "vowel a";
             break;
    case 'e' :
    case 'E' : cout << "vowel e";
             break;
    case 'i' :
    case 'I' : cout << "vowel i";
             break;
    case 'o' :
    case 'O' : cout << "vowel o";
             break;
    case 'u' :
    case 'U' : cout << "vowel o";
```

```
                break;
    default : cout << "not a vowel";
    }
```

અહીં નાના (small) *અથવા મોટા* (capital) *અક્ષરને* input *તરીકે સ્વીકારાશે.*

નોંધ

ઉદાહરણ 8

એવો પ્રોગ્રામ લખો કે જે એક અક્ષર સ્વીકારે અને તેના આધારે નીચે જણાવ્યા પ્રમાણે Basic salary *ને દર્શાવે.*

Grade	Basic salary
A	15000
B	12000
C	10000
D	9000

```
# include < iostream.h>
void main ( )
{
char grade;
cout << "Enter Grade" << "\n";
cin >> grade;
switch (grade)
{
case 'A' : cout << "Basic salary is 15000";
            break;
case 'B' : cout << "Basic salary is 12000;"
            break;
case 'C' : cout << "Basic salary is 10000";
            break;
case 'D' : cout << "Basic salary is 9000";
            break;
default : cout << "Invalid grade code";
}
}
```

પાઠગત પ્રશ્નો 9.1

1. *સંયોજન વિધાનનો શું મતલબ છે?*
2. *સિલેકશન વિધાન શું છે ?* C++ *દ્વારા વર્ણવેલ સિલેકશન વિધાનોના નામ લખો.*
3. switch *વિધાનમાં* default *નું શું મહત્વ છે?*

4. *નીચેના પ્રોગ્રામમાં જો કોઈ ભૂલો હોય તેને શોધી, સાચી રીતે લખો.*

a)
```
# include <iostream.h>
void main ( )
int a, b;
cin << b; <<a;
if (a > b) Max = a
}
```

b)
```
# include <iostream.h>
void main( )
{
int x, y;
cin >> x ;
for (y = 0 ; y < 10, y ++)
if x = = y
cout << y + x ;
else
cout >> y;
}
```

5. *નીચેના પ્રોગ્રામનો આઉટપુટ શું હશે?*

a)
```
cin >> x;
if ( x = = 10 )
cout << "x is 10" ;
else
cout << " x is not 10" ;
```

જો ઈનપુટ આપવામાં આવે :

(i) 11

(ii) 10

b)
```
int year ;
cin >> year ;
if (year % 100 = = 0)
if (year % 400 = = 0)
cout << "Leap" ;
else
cout << "Not century year" ;
```

જો ઈનપુટ છે :

(i) 2000

(ii) 1971

c)
```
int year
cin >> year ;
if (year % 100 = = 0)
{
if (year % 400 = = 0)
cout << "Leap" ;
}
else
cout << "Not century year" ;
```
જો ઈનપુટ આપવામાં આવે :

(i) 2000

(ii) 1971

6. *નીચેના વિધાનોના જેવુ જ* switch *વિધાન લખો.*
```
if (code = = 'a')
cout << "season is summer" ;
else
if (code = = 'r')
cout << "season is rainy" ;
else
if (code = = 'w')
cout << "season is winter";
else
cout << wrong code" ;
```
7. c *ની કિંમત શું હશે?*
```
{
int c;
float a, b;
a = 245.05;
b = 40.02 ;
c = a + b;
}
```
a) 285.09 b) 2850 c) 285 d) 285.0

8. i *અને* k *ની કિંમત શું હશે?*
```
{
int i, j, k ;
j = 5 ;
i = 2 * j / 2 ;
```

```
k = 2 * (j/2) ;
}
```

a) i=4, k=4 b) i=4, k=5 c) i=5, k=4 d) i=5, k=5

9. *નીચેના પ્રોગ્રામનો આઉટપુટ શું હશે?*

```
void main ( )
{
int x = 5, y = 6, z = 7;
if (x < y + z)
cout << "Hello \n" ;
else
{
cout << "How \n" ;
cout << "Are you \n";
}
}
```

a) Hello b) How c) Are you d) None of the above

10. *નીચે આપેલ પ્રોગ્રામનો ભાગ જુઓ.*

```
switch (choice)
{
case 'W' : cout << "WHITE" ;
case 'R' : cout << "RED" ;
case 'B' : cout << "BLUE" ;
default : cout << "error" ;
    break ;
}
```

જો choice = 'R' *હોય તો આઉટપુટ શું હશે?*

a) RED b) RED BLUE error c) RED error

d) WHITE RED BLUE

9.7 લૂપ રચના (Loop Construct)

આ પણ એક પૂનરાવર્તિત નિયંત્રણ માળખુ કહેવાય છે. ક્યારેક આપણને એક કરતા વધારે વિધાનની સંખ્યાબંધ વખત ચલાવવાની જરૂર પડે છે અને દરેક વખતે એક અથવા વધારે variable *(ચલ) ની જરૂર પડે છે. આ પ્રકારના* program *ને* looping *કહેવાય છે.* C++ *નીચેની રચના પૂરી પાડે છે.*

(i) while loop

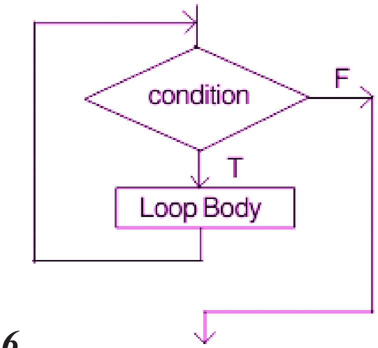
(ii) do - while loop

(iii) for loop

While loop

while loop *ની વાક્ય રચના*

```
while (condition)
{
statement(s);
}
```



આકૃતિ *9.6*

પ્રોગ્રામનો ફલો જોતા એવુ લાગે છે કે condition *નું અમલીકરણ પહેલા થશે. જો* condition *સાચી હશે તો* loop*ની અંદર* (body) *ના વિધાનોનો અમલ થશે અને* condition *ફરીથી મૂલ્યાંકન કરવામાં આવશે. જેથી,* loop body *ના વિધાનો વારંવાર ચલાવવામાં આવશે જ્યાં સુધી* condition *સાચી ઠરશે. જેવું* condition *નું મૂલ્યાંકન ખોટું પડશે તરત* control, loop body *માંથી બહાર આવી જશે અને તરત* 'while'loop *પછીના વિધાનોને અમલમાં* (execute) *મૂકશે. આને વધુ સમજવા માટે આપણે એક ઉદાહરણ લઈએ.*

ઉદાહરણ *9*

પ્રથમ દસ નેચરલ (natural) *નંબરોનો સરવાળો શોધો.* i.e. 1, 2, 3,..................10.

```
# include <iostream.h>
void main ( )
{ int n, total = 0 ;
n = 1 ;
while (n < = 10)
{
   total + = n ;
   n + + ;
}
cout << "sum of first ten natural number" << total ;
}
```

variable *(ચલ)* n *ની કિંમત* loop *નિયંત્રિત કરવા માટે વપરાય છે. જેથી ચલ* n *ને* loop *નિયંત્રણ ચલ કહેવાય છે.સામાન્ય રીતે નીચે દર્શાવેલ ત્રણ કામગીરી,* loop *ચલ ઉપર અનુસરવામાં આવે છે.*

(i) Initialize the loop control variable

(ii) Test the loop control variable

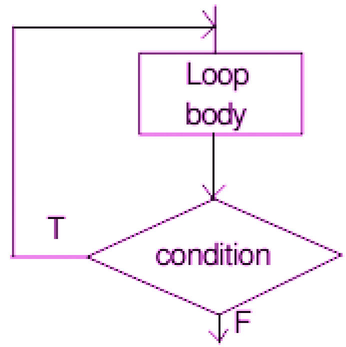
(iii) Update the loop control variable

loop માં પ્રથમ (i) Operation અનુસરવામાં આવવું જોઈએ. બીજુ (ii) Operation પણ દરેક વખત અમલમાં આવવુ જોઈએ જ્યારે loop body ને અમલમાં લાવવામાં આવે ; આ પરિણામ પર આધાર રાખીને loop ક્યાંક તો વારંવાર ચાલશે ક્યાં તો exit થઈ જશે. ત્રીજુ (iii) Operation ફરજીયાત loop body માં સમાવેલુ હોવું જોઈએ. જ્યાં સુધી loop નિયંત્રણ ચલને loop body માં update કરવામાં આવશે ત્યાં સુધી loop નિયંત્રણ ચલ બદલાય નહિં અને loop ફરીથી ચાલશે નહિં.

do-while loop

Syntax of do-while loop

```
do
{
} while (condition);
```



આકૃતિ 9.7

ફલો ચાર્ટ ઉપરથી જાણી શકાય છે કે loop ના દરેક અમલીકરણ પછી condition નું મુલ્યાંકન થશે જો તે સાચુ હશે તો જ loop નું અમલીકરણ પાછું થશે. જો condition ચકાસણી કરતા તે ખોટી હશે તો loop માંથી control બહાર આવી જશે અને loop પછીના વિધાનો ચલાવવામાં આવશે.

Note : do-while, loop ઓછામાં ઓછું એક વાર તો અમલમાં (execute) આવશે જ while અને do-while loop નો એક અગત્યનો તફાવત એ છે કે સંબંધિત condtion નું મુલ્યાંકન અને loop નું અમલીકરણ (execution). While loop માં કોઈ પણ નિવેદનોનો અમલ કરતા પહેલા condition નું મુલ્યાંકન કરવામાં આવે છે. જ્યારે do-while loop માં condition નું મુલ્યાંકન loop નું એક વખત અમલીકરણ થયા પછી કરવામાં આવે છે. જેથી loop body એક વાર તો અમલમાં આવશે જ.

ચાલો આને વધુ સમજવા માટે એક ઉદાહરણ લઈએ.

ઉદાહરણ 10

પ્રથમ દસ નેચરલ (natural) નંબરોનો સરવાળો શોધવો.

```
# include < iostream.h>
void main ( )
{
int N, number, sum;
cin >> N;
sum = 0;
number = 1;
do
{
sum + = number;
number + + ;
```

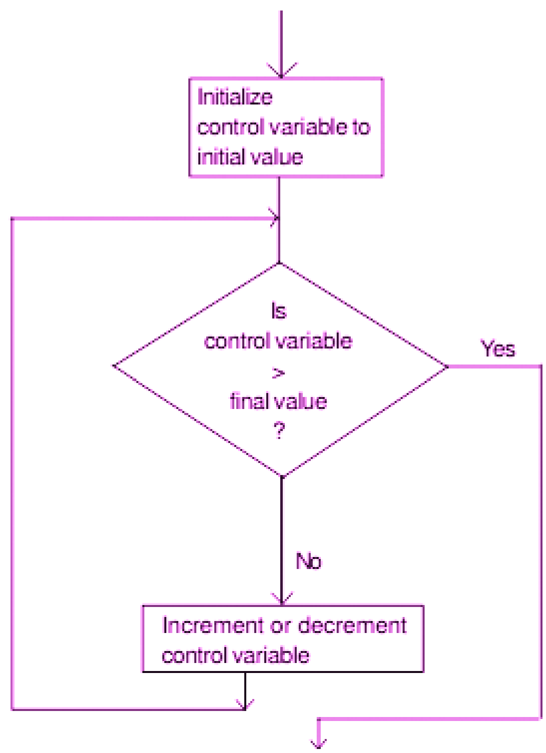
```
}
while (number <= N) ;
cout << sum;
}
```

For loop

તેને ગણતરી નિયંત્રક (count controlled) *લૂપ કહેવાય છે કારણ કે પ્રોગ્રામને પહેલેથી ખબર હોય છે કે કેટલી વખત લૂપને ચલાવવી* (execute) *છે.*

syntax of for loop

```
for (intialization; decision; increment/decrement)
{
statement(s);
}
```



આકૃતિ *9.8*

ફ્લો ચાર્ટ પરથી જાણી શકાય છે કે,

for loop *માં ત્રણ પ્રકારના* operation *નો સમાવેશ થાય છે.*

(i) Initialization of loop control variable *(ચલ)*

(ii) Testing of loop control variable *(ચલ)*

(iii) Update the loop control variable either,by incrementing or decrementing.

Operation (i) *નું અમલીકરણ થયા બાદ* loop control variable *ને પ્રારંભિત* value

મળે છે. બીજી બાજુ operation (ii) નું અમલીકરણ થવા માટેનું કારણ ફક્ત condition સાચી છે કે ખોટી એ જાણવા માટે થાય છે. જો condition સાચી હશે તો program, for loop ની body અમલમાં (execute) મૂકશે અને ત્યાર બાદ loop control variable ની value વધશે કે ઘટશે. ફરીથી condition ચકાસવામાં આવશે અને આ રીતે loop આગળ execute થયા કરશે. જ્યારે condition ખોટી પડશે તરત control loop માંથી બહાર આવી જશે.

ઉદાહરણ 11

```
for (int i = 0 ; i < 5 ; i + + )
        cout << i ;
```

ઉપરના program *નું* output *આ પ્રમાણે આવશે*

0 1 2 3 4

```
for (char i = 'A'; i < 'E' ; i + + )
        cout < < i ;
```

ઉપરના program *નું* output *આ પ્રમાણે આવશે*

A B C D

પાઠગત પ્રશ્નો 9.2

1. while અને do-while લૂપ વચ્ચે તફાવત શું છે?
2. એવી while લૂપ લખો જે નંબર 2, 4, 6, 8 ...... 12 બતાવે.
3. એવી do-while લૂપ લખો જે નંબર 1, 3, 5, 7 ..... 11 બતાવે.
4. નંબર 10 થી 20 બતાવે તેવી for લૂપ લખો.
5. નીચેના લૂપ કેટલી વખત execute થશે?

a)
```
int s = 0, i = 0;
while ( i + + < 5)
        s + = i;
```
b)
```
int s = 0 ; i = 0;
do
        s + = i ;
while ( i < 5 ) ;
```

6. નીચેના પ્રોગ્રામનો આઉટપુટ શું હશે?

a)
```
# include <iostream.h>
void main ( )
{
long number = 7866372, result = 0 ;
do
{
result * = 10 ;        int digit = number % 10 ;
result + = digit ;     number / = 10 ;
}
```

નોંધ

```
while (number) ;
cout << "output =" << result << endl;
}
```

b)
```
void main ( )
{
int i = 0, a = 0, b = 0, c = 0, f = 0 ;
while ( i < = 5)
{
switch (i + +)
{
    case 1 :
    case 2 : + + a ;
    case 3 :
    case 4 : + + b ;
    case 5 : + + c ;
default : + + f ;
}
cout << a << "\n" ;
cout << b << "\n" ;
cout << c << "\n" ;
cout << f << "\n" ;
}
```

7. *નીચેનો પ્રોગ્રામ રન કર્યા પછી* counter *ની કિંમત શું હશે?*

```
void main ( )
{
int counter ;
int digit = 0 ;
counter = 1 ;
while (digit < = 10)
{
+ + counter ;
+ + digit ;
}
cout <<counter ;
}
```

a) 9   b) 10   c) 11   d) 12

8. *નીચેનો પ્રોગ્રામ રન કર્યા પછી* sum *ની કિંમત શું હશે?*

```
void main ( )
{
int sum, i ;
sum = 1 ;
i = 9 ;
do
{
i = i - 1 ;
sum = 2 * sum ;
}
while (i > 9) ;
cout << sum ;
}
```

a) 1 b) 2 c) 9 d) 0.5

9. *નીચેનો પ્રોગ્રામ શું કરે છે?*

```
void main ( )
{
int digit = 0 ;
do
{
cout << digit + + << "\n" ;
}
while (digit < = 10);
}
```

a) 1 to 10 *પૂર્ણાંક નંબર બતાવે.*
b) 0 to 10 *પૂર્ણાંક નંબર બતાવે.*
c) 10 *પૂર્ણાંક ઉમેરો.*
d) *ઉપરનામાંથી કોઈ નહીં.*

10. digit *ની કિંમત શું હશે?*

```
void main ( )
{
int digit ;
for (digit = 0 ; digit < = 9 ; digit + + )
cout << digit << "\n" ;
digit = 5 * digit ;
- - digit ;
}
```

a) 49 b) -1 c) 47 d) 46

નોંધ

9.8 જંપ નિવેદન (Jump Statements)

jump *નિવેદનો બિનશરતી હોય છે.* program control *ને એક જગ્યાએથી બીજી જગ્યાએ* (function*ની અંદર*) transfer *કરે છે.*

a) goto *નિવેદન*

b) break *નિવેદન*

c) continue *નિવેદન*

goto નિવેદન (statement)

goto *નિવેદનની વાકયરચના*

```
goto pgr;
    |
    |
    |
    |
pgr :
```

pgr *એક લેબલ તરીકે લેવું છે. એ વપરાશકર્તા દ્વારા દર્શાવેલુ* identifier *છે.* goto *નિવેદનના અમલીકરણ પછી,* control *ટ્રાન્સફર થઈને* pgr *પછીની લાઈન (નિવેદનો) ને અમલમાં મૂકે છે.*

નોંધ : પ્રોગ્રામની અંદર go to *નિવેદનનો ઉપયોગ એ પ્રોગ્રામીંગની સારી આદત નથી.*

Break *નિવેદન* (statement)

break **નિવેદનની વાક્ય રચના**

break *નિવેદન* switch *નિવેદનની અંદર પણ ઉપયોગમાં લઈ શકાય છે. બીજા બધા* loops *માં પણ તેનો ઉપયોગ થઈ શકે છે.* break *નિવેદન થી* program *નું અમલીકરણ એક જગ્યાએથી બીજી જગ્યાએ નિવેદનો* execute *કરે છે.*

```
while (condition)
{
statement 1;
if (condition)
break ;
statement 2;
}
statement 3;
```

break statement *(નિવેદનો) એક* loop *ને અવગણીને* loop *ની બહાર* control *ને લઈ જાય છે.*

continue statement

continue *નિવેદનનો ઉપયોગ* loop *માં થાય છે, અને* loop body *માં* continue *પછીના નિવેદનોની અવગણના કરીને ફરીથી* loop *અમલમાં મૂકે છે.*

```
while (condition)
{
statement 1;
If (condition)
continue ;
statement 2;
}
statement 3;
```

continue *નિવેદન* loop *ની* body *માં ના બાકીના નિવેદનો છોડીને* loop *નો અમલ* (execution) *ફરીથી કરે છે.*

9.9 exit ( ) function

program *ના અમલીકરણ* (execution) *ને કોઈ પણ સમયે* stop *કરવા* exit ( ) *નો ઉપયોગ થાય છે. અને એના* status *ની જાણકારી* calling program *ને જણાવી દેવાય છે. જેને સામાન્ય રીતે*

exit (code) ; *રીતે લખાય છે.*

જ્યાં code *એ એક* integer value *છે.* code *પાસે સાચા અમલીકરણ માટે શુન્ય કિંમત હોય છે.* code *ની કિંમત* operating system *પ્રમાણે બદલાતી રહે છે.જેનો ઉપયોગ કરવા માટે* process.h *નામની* header file *ની જરૂર પડે છે.*

પાઠગત પ્રશ્નો 9.3

---

1. *એક સંયોજન વિધાન () શું છે ?*
   (i) an operation
   (ii) a label
   (iii) a variable
   (iv) a function
2. break *નિવેદનને બહાર કાઢે છે :*
   i) only from the innermost loop
   ii) only from the innermost switch
   iii) from all loops and switches
   iv) from the innermost loop or switch
3. *લૂપની અંદરનું* continue *નિવેદન* control *ને .... એ લઈ જાય છે.*
4. exit ( ) *ફંકશન વાપરવા માટે કઈ* header file *પ્રોગ્રામમાં વાપરવી જોઈએ?*
5. goto *નિવેદન* i = 100 ; *નો ઉપયોગ કરીને નીચેનો પ્રોગ્રામ ફરી લખો.*

```
while (i)
cout << i — ;
cout << "\n Thank you" ;
```

6. *નીચેના પ્રોગ્રામ* while *લૂપનો ઉપયોગ કરીને ફરીથી લખો.*

```
i = 2 ;
start :
cout << i ;
i + = 2 ;
if ( i < 51) goto start ;
cout << "\n Thank you" ;
```

7. *નીચેના પ્રોગ્રામનો આઉટપુટ શું હશે?*

```
# include < iostream.h>
void main ( )
int x = 10, count = 0 ;
while (x)
{
x - - ;
if (x = = 4)
continue ;
count + + ;
}
cout << "\n" <<count ;
}
```

8.

```
# include < iostream.h>
void main ( )
{
char ch = 'A' ;
while (ch < = 'F' )
{
switch (ch)
{
case 'A' :
case 'B' :
case 'C' :
case 'D' : ch + + ; continue ;
case 'E' :
case 'F' : ch + + ;
}
putchar (ch) ;
}
```

}

What will be the output of the above program ?

a) ABCDEF b) FG c) EFG d) None of the above

9.10 તમે જે શીખ્યા

આ પાઠમાં તમે એવા નિવેદનો શીખ્યા જે પ્રોગ્રામના execution ને નિયંત્રિત કરે છે. હવે તમે control નિવેદનનો ઉપયોગ કરીને નાના પ્રોગ્રામ લખવા માટે સક્ષમ હશો. '{' (opening) કૌંસ અને '}'. (closing) કૌંસ વિશે પણ શીખ્યા.

9.11 પાઠયાંત પ્રશ્નો

1. એવા પ્રોગ્રામ લખો જે નીચે પ્રમાણ મેનુ બતાવે અને choice પૂછે. જો દાખલ કરેલ choice ખોટી હોય તો યોગ્ય મેસેજ બતાવે, નહીં તો દાખલ કરેલો choice નંબર બતાવે.

MENU

1. Create a data file
2. Display a data file
3. Edit a data file
4. Exit

   Choice :

2. એવો પ્રોગ્રામ લખો જે mark સ્વીકારે અને નીચે આપેલા કોષ્ટક મુજબ grade આપે.

Grade	Mark
A	> = 90
B	> = 80 but < 90
C	> = 70 but < 80
D	> = 60 but < 70
F	< 60

3. એવો પ્રોગ્રામ લખો જે વપરાશકર્તા દ્વારા દાખલ કરેલા પૂર્ણાંક સંખ્યાનો સરવાળો બતાવે
4. 500 સુધી fibonacci series ના સભ્યો પ્રિન્ટ કરવા માટેનો પ્રોગ્રામ લખો.
5. આપેલા નંબરને ઉલટો (reverse) કરવા માટેનો પ્રોગ્રામ લખો.
6. એવો પ્રોગ્રામ લખો જે કોઈ પણ નંબરથી કોઈ પણ નંબર સુધીનાં ઘડીયા (table) લખે.

input :

Enter the table of which number : 4

Enter upto which number : 5

Output

4 x 1 = 4

4 x 2 = 8

4 x 3 = 12

4 x 4 = 16

4 x 5 = 20

Continue (Y/N) ?

નોંધ

7. એવો પ્રોગ્રામ લખો જે આપેલ નંબર () નંબર શોધે.
   input : 5
   Output : 5 x 4 x 3 x 2 x 1 = 120
8. 2 અને 20 વચ્ચે ઘડીયા (table) 10 સુધી પ્રિન્ટ કરવા માટેનો પ્રોગ્રામ લખો.
9. 1 થી 100 વચ્ચે Pythagorean triplets એટલે એક નંબરનો વર્ગ એ બીજા બે નંબરના વર્ગોના સરવાળા બરાબર હોય.)
   ઉદાહરણ : 3, 4, 5
   52 = 32 + 42
   25 = 9 + 16 = 25
10. $ax^2 + bx + c = 0$ quadratic સમીકરણનું root નીચેના સૂત્રનો ઉપયોગ કરી શોધો.

$$x = -b \pm \sqrt{\frac{b^2 - 4ac}{2a}}$$

11. નીચેની શ્રેણીનો સરવાળો શોધવા માટેનો પ્રોગ્રામ લખો.
    $1 + x + x^2 + x^2$ .... x^n
12. એવા પ્રોગ્રામ લખો જે નંબર N વાંચે અને 1 થી K વચ્ચેનાં બધા નંબરનો Factorial બતાવે :

Number	Factorial
1	1
2	2
3	6
|	|
|	|
|	|
N	N1

13. આપેલો નંબર એકી (odd) છે કે બેકી (even) તે શોધવા માટેના પ્રોગ્રામ લખો. જો એકી નંબર હોય હોય તો પ્રાઈમ છે કે નહી તે શોધો.
14. એવો પ્રોગ્રામ લખો જે વપરાશકર્તાને N પૂર્ણાંક (integers) દાખલ કરવાનું કહે. નીચેનું શોધો :

(i) positive integer ની સંખ્યા
(ii) negative integer ની સંખ્યા
(iii) zero હોય તેવા integer ની કુલ સંખ્યા

9.12 પાઠગત પ્રશ્નોના જવાબ

પાઠગત પ્રશ્નો 9.1

1. એક સંયોજન વિધાન (statement) ({ }) થી બંધાયેલ હોય છે.
2. C++ દ્વારા પૂરા પાડવામાં આવતા પસંદગી નિવેદનોના ઉપયોગથી કઈ સૂચના (નિવેદનો) (excute) કરવા છે તે પસંદ કરી શકાય છે. ઉદાહરણ. if, if - else,

switch *નામ લખો.*

3. *જો* case *લેબલમાં રહેલ* constant *ની કિંમત,* switch *નિવેદનના ચલની કિંમત અથવા* expression *સાથે મળે નહી ત્યારે* default *લેબલ* execute *થાય છે.*

4. a)
```
# include <iostream.h>
void main ( )
{
int a, b, max;
cin >> b >>a;
if (a > b) Max = a;
}
```

b)
```
# include <iostream.h>
void main ( )
{
int x, y;
cin >> x;
for (y = o ; y < 10 ; y + + )
if (x = = y)
cout << y + x;
else
cout << y;
}
```

5.
   a) (i) x is not 10
      (ii) x is 10
   b) (i) Leap
      (ii) No output
   c) (i) Leap
      (ii) Not century year

6. **નીચેના માટે સમકક્ષ Switch (code)**
```
{
case 'a' : cout << "season is summer" ; break;
case 'r' : cout << "season is rainy"; break;
case 'w' : cout << "season is winter"; break ;
default : cout << "wrong code";
}
```

7. c)
8. c)
9. Hello

નોંધ

10. b)

પાઠગત પ્રશ્નો 9.2

1. while લૂપમાં condition પહેલાં તપાસવામાં આવશે પછી લૂપના નિવેદન execute કરાશે જ્યારે do - while માં લૂપના નિવેદનો પહેલાં execute થશે પછી condition તપાસવામાં આવશે, તેથી લૂપના નિવેદનો એક વખત ચોક્કસ execute થશે પછી ભલે condition ખોટી હોય.

2.
```
int i = 2 ;
while (i < = 12)
{
cout << i ;
i + = 2 ;
}
```

3.
```
int i = 1 ;
do
{
cout << i ;
i + = 2 ;
} while (i < = 11);
```

4.
```
for (int i = 10; i < = 20 ; i + + )
cout << i ;
```

5.
 a) 5 times
 b) endless

6.
 a) output = 2 7 3 6 6 8 7
 b) 2
 4
 5
 6

7.
 d)

8.
 b)

9.
 b)

10.
 a)

પાઠગત પ્રશ્નો 9.3

1. (ii)   2. (iv)

3. the decision statement of loop
4. process.h
5. i = 100 ;
   xyz : i - - ;
   if i > 0 goto xyz ;
   cout << "\n Thank you" ;
6. i = 2 ;
   while (i < 51)
   {
   cout << i ;
   i + = 2, ;
   }
   out << "\n Thank you" ;
7. 9
8. (b)

10

કાર્યો

10.1 પ્રસ્તાવના

C++ ભાષામાં, કાર્યો, C++ લાઈબ્રેરી હેડર ફાઈલો હેઠળ ઉપલબ્ધ છે. તે કાર્યની ઘોષણા, ક્લાસની વ્યાખ્યા વિગેરે પુરૂ પાડે છે. દાખલા તરીકે,

```
double sqrt (float) ;
```

ઉપરોક્ત ઘોષણા ઉલ્લેખ કરે છે કે આ sqrt કાર્ય એક દલીલ (argument) float પ્રકારની લે છે અને double પ્રકાર પરત કરે છે. પૂર્વ વ્યાખ્યાયિત કાર્યો (વિધેયો) વાપરવા માટે, હેડર ફાઈલ (જે કાર્યોના નમૂના ધરાવે છે) નો પ્રોગ્રામ માં સમાવેશ કરવો જ જોઈએ તેના માટે Pre-processor માર્ગદર્શક સૂચના(directive) #include નો ઉપયોગ કરવો.

10.2 હેતુઓ

આ પાઠ પસાર કર્યા પછી તમે નીચેના કાર્યો કરવા સક્ષમ હશો :

- #include ના ઉપયોગ અને કાર્યક્રમમાં તેની જરૂરિયાત વિશે પરિચિત હશો.
- math.h, ctype.h અને string.h હેડર ફાઈલમાં સમાવિષ્ટ કાર્યોની વ્યાખ્યા કરી શકશો.
- character (અક્ષર) હેન્ડલીંગ કાર્યો જે ઈનપુટ/આઉટપુટ માં મદદ કરે છે તે સમજાવી શકશો.
- inline કાર્ય અને મૂળભૂત કિંમત સાથેના કાર્યને વ્યાખ્યાયિત કરી શકશો.

10.3 # include directive (માર્ગદર્શક સૂચના)

include directive, *સંકલનકર્તા* (compiler) *ને બીજી ફાઈલ વાંચવા અને તેનો હાલની ફાઈલમાં સમાવેશ કરવાની સૂચના આપે છે. કંમ્પાઈલર સંપૂર્ણ કોડને કમ્પાંઈલ કરે છે.*

શીર્ષક (Header) *ફાઈલ બેમાંથી એક રીતે સમાવિષ્ટ કરી શકાય.*

```
# include < iostream. h>
```

અથવા

```
# include "iostream.h"
```

કોણ કૌંસે (Angle brackets) *માં શીર્ષક ફાઈલના સમાવેશનો મતલબ છે કે તે ફાઈલ પ્રમાણભૂત ડીરેકટરીમાં સમાવિષ્ટ છે. શીર્ષક ફાઈલનો સમાવેશ* double quotes *નો મતલબ છે કે તે ફાઈલ હાલની ડિરેકટરીમાં છે.*

10.4 Library Functions (લાઈબ્રેરીના કાર્યો)

C++ *ઘણા પૂર્વ નિમિત કાર્યો આપે છે. જે પ્રોગ્રામીંગ સમય બચાવે છે.*

ગાણિતિક કાર્યો (Mathematical Functions)

ગાણિતિક કાર્યો જેવા કે sin( ), cos ( ) *વિગેરે* math.h. *શીર્ષક ફાઈલમાં વ્યાખ્યાયિત કરેલ છે. થોડા મહત્વના ગાણિતિક કાર્યો નીચેના ટેબલમાં આપેલ છે.*

કાર્ય	અર્થ
sin(x)	x *ખૂણાનું* Sine *મૂલ્ય (રેડિયનમાં માપેલ)*
cos(x)	x *ખૂણાનું* Cosine *મૂલ્ય (રેડિયનમાં માપેલ)*
tan(x)	x *ખૂણાનું* Tangent *મૂલ્ય (રેડિયનમાં માપેલ)*
asin(x)	$\sin^{-1}$ x (x *નું મૂલ્ય રેડિયનમાં માપેલ)*
acos(x)	$\cos^{-1}$ x *(Xનું મૂલ્ય રેડિયનમાં માપેલ)*
exp(x)	x *નું ઘાતાકીય કાર્ય*
log(x)	x *નું ઘાતાંક (*legarithim of x *)*
log10 (x)	x *નું ઘાતાંક (*base 10*)*
sqrt (x)	x *નો વર્ગમૂળ*
pow (x,y)	*પૂર્ણાંક* x *ની સંપૂણ કિંમત*
abs (x)	*વાસ્તવીક સંખ્યા* x *ની સંપૂર્ણ કિંમત*

Charater Functions મૂળાક્ષર કાર્યો.

તમામ મૂળાક્ષર કાર્યોને ctype.h શીર્ષક ફાઈલની જરૂર પડે છે. નીચેનું કોષ્ટક તે કાર્યોની યાદી આપે છે.

String (શબ્દમાળા)	Functions (કાર્યો)
isalpha (c)	જો c નું મૂલ્ય A to Z અથવા a to z હશે તો તે સાચું પરત કરશે નહીતર ખોટું પરત કરશે.
isdigit (c)	જો C નું મૂલ્ય આંકડો (0 થી 9) હશે તો કાર્ય સાચું પરત કરશે નહીતર ખોટું પરત કરશે.
isalnum (c)	જો C નું મૂલ્ય 0 થી 9 અથવા મૂળાક્ષરના અક્ષર હશે તો સાચું પરત કરશે.
islower (c)	તે સાચુ પરત કરશે જો C નું મૂલ્ય Uppercase અક્ષર હશે નહીતર ખોટું
is- upper(c)	તે સાચું પરત કરશે જો c નું મૂલ્ય Uppercase અક્ષર હશે નહીતર ખોટું.
toupper (c)	તે c ને Uppercase માં રૂપાંતરિત કરશે.
to lower (c)	તે c ને lowercase માં રૂપાંતરણ કરશે.

શબ્દમાળા કાર્યો (String Functions)

શબ્દમાળા કાર્યો string.h. શીર્ષક ફાઈલમાં હાજર છે. કેટલાક શબ્દમાળા કાર્યો નીચેના કોષ્ટકમાં આપેલ છે.

strlen (S)	આ કાર્ય શબ્દમાળામાં હાજર જગ્યાઓ સહિતના અક્ષરોની સંખ્યા આપે છે.
strcat (S_1, S_2)	આ કાર્ય શબ્દમાળા S_2 ને શબ્દમાળા S_1 ના પાછળ જોડે છે. આના માટે S_1 શબ્દમાળા પાસે S_2 શબ્દમાળાને સમાવિષ્ટ કરવા પૂરતી જગ્યા હોવી ફરજિયાત છે.
strcpy (S_1, S_2)	આ કાર્ય શબ્દમાળા S_2 ની નકલ શબ્દમાળા S_1 માં કરે છે. શબ્દમાળા S_1 માં ફરજિયાત પૂરતું સંગ્રહસ્થાન હોવું જરૂરી છે.
strcmp ((S_1,S_2) = = 0) strcmp ((S_1,S_2) < 0)) strcmp ((S_1, S_2) > 0)	આ કાર્ય શબ્દમાળા S_2 ને S_1 સાથે સરખાવીને ખાતરી કરે છે કે S_1 અને S_2 સરખા છે, S_1 શબ્દમાળા S_2 કરતા મોટી છે કે નાની છે.

console I/O functions : *નીચે કાર્યોની યાદી આપી છે.*

(i) getchar ( )

(ii) putchar ( )

(iii) gets ( )

(iv) puts ( )

ઉપરોક્ત કાર્યો માટે શીર્ષક ફાઈલ stdio.h *છે. પ્રથમ બે કાર્યો એક અક્ષર સાથે કામ કરે છે અને છેલ્લા બે કાર્યો શબ્દમાળા સાથે કાર્ય કરે છે.*

Getchar( ) કાર્ય

getchar( ) *કાર્ય પ્રમાણભૂત ઈનપુટ ઉપકરણમાંથી એક અક્ષર પરત કરે છે. આ કાર્ય કોઈ પરિમાણ* (Parameter) *લેતું નથી. અને પરત કરેલ મૂલ્ય ઈનપુટ અક્ષર છે.* getchar ( )*; નું સામાન્ય સ્વરૂપ છે.*

```
A = getchar( );
```

ચલ A *અક્ષર* (character) *પ્રકારનો છે. તે ઈનપુટ ઉપકરણમાંથી અક્ષર ઈનપુટ કરે છે અને તે* A *ને આપે છે.*

Putchar( ) કાર્ય

putchar ( ) *કાર્ય એક દલીલ* (argument) *લે છે. જે એ અક્ષર છે, જેને આઉટપુટ ઉપકરણ પર મોકલવાના હોય છે. તે આ અક્ષરને પરિણામ તરીકે પરત કરે છે.* putchar ( ) *નું સામાન્ય સ્વરૂપ છે.*

```
putchar (ch) ;
```

જ્યાં ch *એ અક્ષર* (Character) *પ્રકારનો ચલ છે.*

ઉદાહરણ - 1.

```
# include < iostream.h>
# include < stdio. h>
void main ()
{
        char ch;
        ch = getchar ( ) ;
        putchar (ch);
}
```

ઉપરોક્ત પ્રોગ્રામ કી - બોર્ડ પરથી એક અક્ષર લે છે, અને તેને સ્ક્રીન પર છાપે છે.

gets ( ) function gets ( ) કાર્યને પ્રમાણભૂત ઈનપુટ સ્ટ્રીમ stdin માંથી ન્યુલાઈન અક્ષરથી પૂરી થયેલ શબ્દમાળા મળે છે. gets ( ) ન્યુલાઈનને Null character (0) થી બદલે છે, અને તે અંદર નાંખેલી શબ્દમાળામાં સફેદ અક્ષરો (જગ્યાઓ, ટેબ) સમાવી લેવાની પરવાનગી આપે છે. gets ( )ને જ્યારે New line અક્ષર મળે છે ત્યારે એ Newline (નવી લાઈન) સુધીનું બધું પરત કરે છે અને તેને A માં નકલ કરે છે.

```
gets (A) ;
```

ઉદાહરણ - 2

```
# include <stdio.h>
# include < iostream.h>
void main ()
{
char A [ 100] ;
count << "input a string";
gets (A) ;
puts (A) ;
}
```

બીજા કેટલાક કાર્યો (Some more functions)

The getch ( ) અને getche ( ) કાર્યો.

getch ( ) અને getche ( ) ના સામાન્ય સ્વરૂપ નીચે મુજબ છે.

```
ch = getche ( ) ;
ch1 = getch ( ) ;
```

ch અને ch 1 અક્ષર (character) પ્રકારના ચલ છે. તેઓ કોઈ પ્રકારની દલીલ લેતાં નથી અને તેને conio.h શીર્ષક ફાઈલની જરૂર પડે છે. આનો અમલ (execution) કરતી વખતે cursor બ્લીન્ક થાય છે ત્યારે વપરાશકર્તા એ ફરજિયાત અક્ષર ટાઈપ કરવો પડે છે. getche ( ) એ પરત કરેલ અક્ષરનું મૂલ્ય ch ને આપવામાં (assign) આવે છે. getche ( ) કાર્યનો Screen ઉપર પડઘો પડે છે. તેથી getche માં "e" છે. બીજું કાર્ય getch ( ) છે જે getche () જેવું જ છે. પરંતુ તેમાં અક્ષરનો પડઘો Screen પર નથી પડતો.

ઉદાહરણ - 3

વપરાશકર્તા (user) એ નાંખેલ વાકયમાં અક્ષરો અને શબ્દોની ગણતરી

```
# include < iostream.h>
# include < conio.h>
void main ()
{
char ch ;
```

```
int chcnt = 0, wdcnt = 1;
while (( ch = getche ( )) ! = ‘\r’)
{
   if (ch= = ‘ ‘)
   wdcnt ++ ;
   else
   chcnt ++ ;
}
cout << “ No. of character” << chcnt <<“\n”;
cout << “ No. of words” << wdcnt;
}
```

ઉપરોક્ત પ્રોગ્રામ વાકયમાં કુલ અક્ષર અને શબ્દો ગણે છે. જે વાકય રીર્ટન કી થી પુરું થયું હોય.

10.1 પાઠગત પ્રશ્નો

1. *શીર્ષક* (header) *ફાઈલ શું છે ? તેનો ઉદેશ્ય શું છે ?*
2. *શીર્ષક ફાઈલને પ્રોગ્રામમાં સમાવિષ્ટ કરવાની શું જરૂરિયાત છે ?*
3. *નીચેના માટે જરૂરી શીર્ષક ફાઈલનું નામ આપો.*
   (i) abs ( )
   (ii) strcmp ( )
   (iii) clrscr ()
   (ix) tolower ( )
4. *લાઈબ્રેરી કાર્ય શું છે ?*
5. getch ( ) *અને* getche ( ) *વચ્ચેનો તફાવત જણાવો.*
6. *નીચેનાનો આઉટપુટ શું હશે.*

   (a)
```
# include <iostream.h>
# include < stdio.h>
void main ()
{
char name [100] ;
puts ( “enter your name”);
gets (name) ;
puts (“Hello”);
puts (name) ;
}
```

Input : MOHIT

(b)
```
# include < iostream.h >
# include < stdio.h >
void main ( )
{
char c ;
c = getchar ( ) ;
while  (c!= '.')
{
putchar (c);
c = getchar ( ) ;
}
Input : RUBY
```

(7) ખાલી જગ્યા પૂરો.

(1) બધી શીર્ષક ફાઈલને .... લખાણ (extension) હોય છે.

(2) અક્ષર કાર્યો .... માં ઉપલબ્ધ છે.

(3) toupper (C) કાર્ય C ને ....માં રૂપાંતરીત કરે છે.

(4) strcpy (s_1,s_2) માં શબ્દસમૂહ ....ની નકલ શબ્દસમૂહ ....માં થાય છે.

(5) ફંકશન જે શબ્દસમૂહની લંબાઈ ગણે છે. તે .... ફંકશન છે.

(6) sin () કાર્ય .... માં ઉપલબ્ધ છે.

(8) નીચેના ખરાં છે કે ખોટાં તે જણાવો.

(a) isdigit () કાર્ય ચેક કરે છે કે આપેલ અક્ષર એક નંબર છે કે નહી.

(b) બધા શબ્દસમૂહ કાર્યો stdio.h માં ઉપલબ્ધ છે.

(c) iostream.h બધા ઈનપુટ/આઉટપુટ કાર્યો ધરાવે છે.

(d) tolower (c) , c ને લોઅરકેશ(નાના અક્ષરો) માં રૂપાંતરિત કરે છે.

(e) strcat (s1, s2) બે શબ્દસમૂહોને ત્રણમાં વિભાજીત કરે છે.

(f) pow (x,y), y ના x powerની ગણતરી કરે છે.

(g) cos (x), x કોણનો Cosine પરત કરે છે.

(h) tolower () કાર્ય ctype.h. માં ઉપલબ્ધ છે.

(i) log (x) એ string.h. માં ઉપલબ્ધ છે.

10.5 વપરાશકર્તા દ્વારા બનાવેલ C++ કાર્યો (User defined C++ function)

C++ *કાર્ય એ એક એકમમાં પ્રોગ્રામના નિવેદનોનું જૂથ છે.* main ( ) *એ બધા* C++ *પ્રોગ્રામમાં આવશ્યક કાર્ય છે. જો કોઈ પ્રોગ્રામમાં એક જ કાર્ય હોય તો તેનું નામ* main ( ) *હોવું જોઈએ.* main ( ) *કાર્ય એ પ્રોગ્રામના અમલ માટેનું શરૂઆતનું બિંદુ છે.* main () *ની વ્યાખ્યા નીચેના જેવી લાગે છે.*

```
main ( )
{
// Main program statements
}
```

C++ *માં* main () *કાર્ય* , int *પ્રકાર નું મૂલ્ય* operating system *ને પરત કરે છે. જે કાર્ય ને* return value *(પરત મૂલ્ય) હોય તેનો અંત* return statement *થી કરવો જોઈએ. તેથી* main () *કાર્ય, નીચેની રીતે વ્યાખ્યાયિત થાય છે.*

```
int main ()
{
return (0) ;
}
```

મૂળભૂત રીતે કાર્ય પરત પ્રકાર પૂર્ણાંક (int) *છે. તેથી પૂર્ણાંક* (int) *કી વર્ડ,* main ( ) *શીર્ષકમાં વૈકલ્પિક છે.*

નીચે આપેલો પ્રોગ્રામ ધ્યાનમાં લો.

```
# include <iostream.h>
void main ()
{ z
cout << " computer science", "/n";
}
```

ઉપર આપેલ પ્રોગ્રામ સ્ક્રીન ઉપર "Computer Science" *પ્રિન્ટ કરશે. આપણે ઉપરનો પ્રોગ્રામ નીચેની રીતે પણ લખી શકીએ.*

```
# include < iostream.h>
void message () ;
void main ()
{
message ( ) ;
}
```

```
void message ()
{
cout <<" computer science \n" ;
}
```

ઉપરનો કાર્યક્રમ પણ Computer Science *પ્રિન્ટ કરશે પરંતુ તે પ્રિન્ટીંગ માટે* message () *કાર્યનો ઉપયોગ કરશે.*main () *અને* message () *કાર્ય ઉપરાંત,* # include preprocessor directive *(માર્ગદર્શક સૂચના) પછી એક વધારાની લાઈનમાં* void message () *તે કાર્યનો નમૂનો* (prototype) *તરીકે ઓળખાય છે.*

10.5.1 Function Prototype (કાર્ય નમૂનો)

કાર્યનો નમૂનો, કંમ્પાઈલરને જે કાર્ય પ્રોગ્રામમાં વાપરવાના હોય તે, દલીલ જે તે કાર્ય લેશે તે અને પરત મૂલ્યના પ્રકાશ વિશે માહિતગાર કરે છે. જે કાર્ય કોઈ મૂલ્ય પરત નથી કરતા તેવા કાર્યો void *કાર્ય તરીકે ઓળખાય છે.*

10.5.2 કાર્યની દલીલો (Arguments to a Function)

કોઈકવાર Calling *કાર્ય,* Called *કાર્યને અમુક કિંમત આપે છે તે પરિમાણ તરીકે ઓળખાય છે. જે ચલ* (variables) calling *કાર્યને કિંમતો પૂરી પાડે છે, તે વાસ્તવિક પરિમાણો કહેવાય છે. ચલ જે* Called *નિવેદનથી કિંમત મેળવે છે તે ઔપચારિક પરિમાણો તરીકે ઓળખાય છે.*

નીચેના પ્રોગ્રામ ધ્યાનમાં લો જે વર્તુળના વિસ્તારનું મૂલ્યાંકન કરે છે.

```
# include <iosteram.h>
void area (float);
void main ()
{
float radius ;
cin >> radius;
area (radius) ;
}
void area (float r )
{
cout << ( "The area of circle is" << 3.1416*r * r << "\n" ;
}
```

વાકય area (radius) *, કાર્યને આવાહન કરે છે જે* area (6.0) ; *તરીકે લખી શકાય.*
આ કાર્યનું મૂલ્યાંકન અને ત્રિજયાનું મૂલ્ય 6.0 *લઈ વર્તુળનો વિસ્તાર દર્શાવશે.*
અહીં ત્રિજયાને (radius) *વાસ્તવીક પરિમાણ કહેવામાં આવે છે અને* r *ઔપચારિક પરિમાણ કહેવાય છે.*

10.5.3 કાર્યનો પરત પ્રકાર (Return Type of a function)

ઉપરના કાર્યક્રમમાં વર્તુળનો વિસ્તાર, area ( ) *કાર્યની અંદર જ બતાવવામાં આવે છે. જો વર્તુળનો વિસ્તાર મુખ્ય કાર્યમાં જોઈએ તો પરત વાક્ય* (Return Statement) *થી શક્ય છે. આવા પ્રોગ્રામને નીચેની રીતે લખી શકાય.*

```
# include <iostream.h>
float area (float) ;
void main ()
{
float radius, y;
cin >> radius;
y = area (radius) ;
cout << "The area of the circle is" <<y ;
}
float area (float r)
{
return (3.1416*r*r);
}
```

કાર્ય નમૂનામાં void *ની બદલી* float *થી થાય છે. તે સૂચવે છે કે કાર્ય* area ( ), main ( ) *કાર્યને* float *મૂલ્ય પરત કરે છે.* calling *કાર્યમાં ચલ* y *ને શોધવા* area *કાર્ય એ પરત કરેલ વેલ્યુ (કિંમત) આપવામાં આવે છે. આમ, ચલ* y *ને વર્તુળનો વિસ્તાર સોંપવામાં* (assigned) *આવે છે.*

10.5.4 વૈશ્વિક અને સ્થાનિક ચલો (Global and Local Variables)

પ્રોગ્રામમાં જાહેર ચલ વ્યાપક રીતે નીચેની રીતે વર્ગીકૃત કરી શકાય.

(1) સ્થાનિક ચલ

(2) વૈશ્વિક ચલ

સ્થાનિક ચલ (Local Variable)

ચલ (variable) *જે કાર્યની અંદર જાહરે* (declare) *કરવામાં આવેલ હોય તેનું મૂલ્યાંકન એ કાર્યની અંદર જ કરવામાં આવશે. કાર્યક્રમના જે ભાગની મેમરીમાં ચલ ને જાળવી રાખેલ હોય તે તેના અવકાશ* (Scope) *તરીકે ઓળખાય છે. સ્થાનિક ચલનો અવકાશ તે જયાં વ્યાખ્યાયિત કરેલ હોય તે કાર્ય જ હોય છે. એક ચલ કાર્ય અથવા સંયોજન નિવેદન* (compound statement) *માટે સ્થાનિક હોઈ શકે છે.*

વૈશ્વિક ચલ (Global variable)

કોઈ પણ ચલ જે કાર્યની બહાર જાહેર કરવામાં આવે છે તે ચલ વૈશ્વિક ચલ તરીકે ઓળખાય છે. આવા ચલનો અવકાશ કાર્યક્રમ (Programme) *ના અંત સુધી હોય છે. આ ચલ વ્યાખ્યાયિત કર્યા પછી જે પણ કાર્ય વ્યાખ્યાયિત થાય તે બધા કાર્યોમાં તે ચલ ઉપલબ્ધ હોય છે. તેથી વૈશ્વિક*

ચલને કાર્યક્રમની શરૂઆતમાં વ્યાખ્યાયિત કરવા જોઈએ. જો પ્રોગ્રામમાં ચલ "a" સ્થાનિક અને વૈશ્વિક તરીકે જાહેર કરેલ હોય તો "a" સ્થાનિક ચલ રજૂ કરે છે. જ્યારે : : a એક વૈશ્વિક ચલ રજૂ કરે છે. :: પ્રતીક અવકાશ પ્રસ્તાવ (scope resduction) ઓપરેટર કહેવાય છે.

ઉદાહરણ 4

```
# include < iostream.h>
int  m = 2;
void main ( )
{
int m = 15
{
    int m = 10* : : m ;
    cout << 'm = " << m << "\n" ;
    cout << " :: m =" << :: m << "\n" ;
}
cout << " m=" << m << "\n" ;
cout << " : : m = " < < "\n";
}
```

ઉપરના પ્રોગ્રામની નીપજ (Output)

```
m = 20
:: m = 2
m = 15
:: m =2
```

ચલો અને સંગ્રહ વર્ગ (varibles and storage class)

ચલનો સંગ્રહ વર્ગ નક્કી કરે છે કે પ્રોગ્રામનો કયો ભાગ તેને વાપરી શકે છે અને તે કેટલા સમય સુધી અસ્તિત્વમાં રહે છે. સંગ્રહવર્ગ નીચેની રીતે વર્ગીકૃત કરી શકાય છે.

(1) સ્વંયસંચાલિત (Automatic)

(2) પત્રક (Register), નોંધણી કરાવવી.

(3) સ્થિર (static)

(4) બહારનું (External)

સ્વયંસંચાલિત ચલ (Automatic varible)

બધા ચલો (Variable) મૂળભૂત રીતે સ્વયંસંચાલિત હોય છે. દા.ત. જાહેરાતો int a અને auto int a બંને સમકક્ષ છે. સ્વયંસંચાલિત ચલો તેનો અવકાશ કાર્યક્રમના અંત સુધી જાળવી રાખે છે. જેમાં એની જાહેરાત કરવામાં આવી હોય છે. સ્વંયસંચાલિત ચલ બનાવવામાં આવતા નથી. જ્યાં સુધી તેની જાહેરાત જે કાર્યમાં કરી છે, તેને બોલવવામાં ન આવે. (Called function) જ્યારે

નિયંત્રણની કાર્યની બહાર નીકળે છે અને તે કોલિંગ કાર્ય/પ્રોગ્રામ માં આવે છે. ત્યારે ચલો નાશ પામે છે અને તેમના મૂલ્યો (values) *ખોવાઈ જાય છે. નામ સ્વંયસંચાલિત* (Automatic) *નો ઉપયોગ કરવામાં આવે છે કારણ કે ચલો આપમેળે બનાવવામાં આવે છે જયારે કાર્યને બોલાવવામાં આવે છે અને તેનો નાશ થાય છે, જયારે કાર્ય પરત કરે છે.*

ઉદાહરણ - 5

```
# include <iostream.h>
void sum (int) ;
void main ()
    {
int i ;
for ( i =1; i< =5; i ++ )
sum (i) ;
}
void sum ( int n)
{
    auto int s = 0 ;
    s = s+ n ;
    cout << s << "\n" ;
}
```

પ્રોગ્રામની નીપજ (out put) *છે.*

આઉટપુટ : 1
2
3
4
5

પત્રક ચલ/નોંધણી ચલ (Register Variables)

એક રજીસ્ટર ઘોષણાએ સ્વંયસંચાલિત (auto) *ઘોષણા છે. રજીસ્ટર ચલમાં સ્વયંસંચાલિત ચલની બધી લાક્ષણિકતાઓ હોય છે. બંનેમાં તફાવત એ છે કે રજીસ્ટર ચલ ઝડપથી ઉપયોગ કરી શકાય છે, કારણ કે તે* CPU *માં સંગ્રહ થાય છે, નહીં કે મેમરીમાં. રજીસ્ટર અને ઓટો સ્થાનિક ચલને લાગુ કરી શકાય છે.*

સ્થિર ચલ (Static Varible)

ચલ, સ્વંયસંચાલિત સ્થિર ચલ અથવા બ્રાહ્ય ચલ હોઈ શકે છે. સ્થિર બ્રાહ્ય ચલ છે. વિવિધ ફાઈલ પ્રોગ્રામમાં જ અર્થપૂર્ણ છે. સ્થિર સ્વયંસંચાલિત ચલની દશ્યતા સ્થાનિકચલ જેવી પરંતુ જીવનકાળ બ્રાહ્ય ચલનો હોય છે. તેથી તે દેખાય છે, માત્ર કાર્યમાં જયાં તે વ્યાખ્યાયિત કરેલ છે. પરંતુ તેનું અસ્તિત્વ પ્રોગ્રામના જીવન સુધી રહે છે. જો પાછળના પ્રોગ્રામમાં int s = 0 *ને બદલી નાંખવામાં આવે* Static int s = 0 *થી તો પ્રોગ્રામની નીપજ નીચે મુજબ હશે.*

આઉટપુટ :

```
1
3
6
10
15
```

બાહ્યચલ (External variable)

મોટા પ્રોગ્રામ, એક કરતાં વધારે વ્યકિતઓ દ્વારા અલગ - અલગ ફાઈલમાં લખી શકાય છે. એક ચલને એક ફાઈલમાં વૈશ્વિક જાહેર કરવામાં આવે તો બીજી ફાઈલના કાર્ય માટે ઉપલબ્ધ રહેશે નહી. આવા ચલ જે બંને ફાઈલોના કાર્યો માટે જરૂરી છે, તેને એક ફાઈલમાં વૈશ્વિક ચલ જાહેર કરવો જોઈએ, અને બીજી ફાઈલમાં બ્રાહય ચલ જાહેર કરવો જોઈએ.

દા.ત. extern int a ;

જયાં extern, *બ્રાહ્ય માટે મુખ્ય શબ્દ છે.*

10.5.5 કાર્યને બોલાવવું (calling of Function)

કાર્યને નીચેની પધ્ધતિઓથી બોલાવી શકાય.

(1) કિમંતથી બોલાવવું. (call - by - value)

(૨) સંદર્ભથી બોલાવવું. (call - by- refernence)

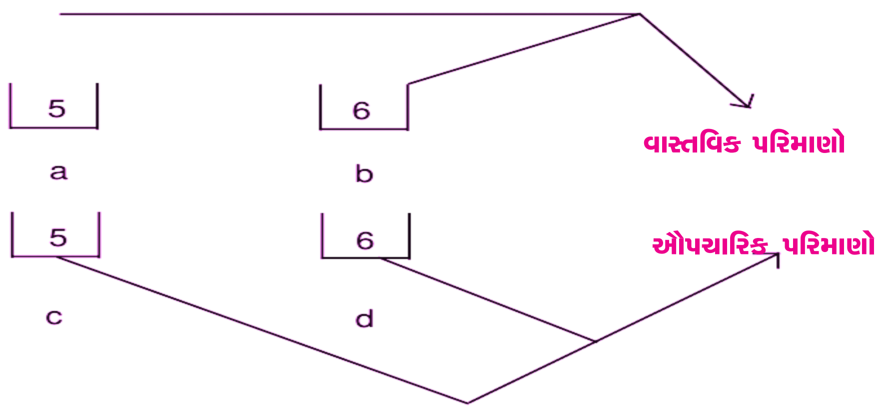
કિમંતથી બોલાવવું (Call - by - value)

નીચેના પ્રોગ્રામને ધ્યાનમાં લો તે બે ચલની કિંમત call - by - value *પધ્ધતિથી અદલાબદલી કરશે.*

```
# include < iostream.h.>
void swap (int, int) ;
void main ()
{
    int a,b ;
    cin >> a >> b;
    swap (a,b) ;
    cout << a << b << "\n" ;
}
void swap (int c, int d)
{
    int t ;
    t = c;
    c = d ;
    d = t ;
    cout << c << d << "\n" ;
```

}

ચલ a,b, *વાસ્તવિક પરિમાણો કહેવાય છે જ્યારે* swap () *કાર્યને બોલાવવામાં આવે છે ત્યારે* a *ચલ નું મૂલ્ય* c *ચલને પસાર થાય છે અને* c *નું મૂલ્ય* d *ને પસાર થાય છે. અહીંયા* c *અને* d *ઔપચારિક પરિમાણો કહેવામાં આવે છે.*



કાર્ય swap ( ) *માં મૂલ્યો - અદલાબદલી થાય છે. પરંતુ આ બદલાવ વાસ્તવિક પરિમાણોમાં અમલ નથી થતો.*

Input	5	6
Output	6	5
	5	6

સંદર્ભથી બોલાવવું (call by reference)

નીચેનો પ્રોગ્રામ સંદર્ભથી બોલાવવું (call - by - reference) *ને ગણતરીમાં લઈ બે ચલની કિંમત* (value) *ની અદલાબદલી કરશે.*

```
# include <iostream. h>
void swap (int &, int &) ;
void main ( )
{
    int a, b;
    cin >> a >> b ;
    swap (a,b) ;
    cout << a << b << "\n") ;
}
```

```
void swap (int & c, int & d )

{

    int t ;

    t = c ;

    c = d ;

    d = t;

    cout << c << d <<“ \n” ;

}
```

આ પધ્ધતીમાં જયારે કાર્યને બોલાવવામાં આવે ત્યારે સરનામા (address) વાસ્તવિક પરીમાણોથી ઔપચારિક પરિમાણોમાં બદલી કરવામાં આવે છે. એનો મતલબ a અને c સમાન સરનામા શેર (share) કરી રહયા છે. અને b અને d સમાન સરનામા શેર કરે છે.

6	5
5	6
a/c	b/d

ઈનપુટ	5	6
આઉટપુટ	6	5

એનો મતલબ ઔપચારિક ચલમાં કોઈ બદલાવ વાસ્તવીક રૂપમાં અમલ થાય છે.

10.6 ઈનલાઈન કાર્ય (inline Function)

કાર્ય મેમરીનો બચાવ કરે છે, કારણ કે, કાર્યને કોલ દરેક વખતે એક જ કોડનો અમલ કરે છે . જયારે કંમ્પાઈલર કાર્યનો કોલ (call) જોવે છે, ત્યારે સામાન્ય રીતે એ કાર્ય પર કુદકો મારે છે. કાર્યના અંતમાં તે પાછો કૂદકો કાર્ય કોલ પછીના વાકય પર મારે છે.

ઘટનાઓ (events) નો ક્રમ મેમરીમાં જગ્યા બચાવે છે. પરંતુ એ અમલમાં વધારાનો સમય લે છે. નાના કાર્યોમાં અમલનો સમય બચાવવાં ઈન લાઈન કાર્યનો ઉપયોગ થાય છે. દરેક વખતે જયારે કાર્યને બોલાવવામાં આવે છે. ત્યારે કાર્ય પર કૂદકો મારવાના બદલે વાસ્તવિક કોડ દાખલ કરવામાં આવે છે.

ઉદાહરણ - 6

```
# include < iostream.h>

inline int cube (int r)

{

        return (r * r * r) ;
```

```
}
void main ()
{
int number, y ;
cout << "Enter Number";
cin >> number ;
y = cube (number) ;
cout << 'The cube of the number is" << y;
}
```

નોંધવા જેવા મહત્વના મુદ્દાઓ

1. *કાર્યના નામ પહેલા* inline *શબ્દ મૂકીને કાર્યને ઈનલાઈન બનાવવામાં આવે છે.*
2. *ઈનલાઈન કાર્યની જાહેરાત* main () *પહેલા કરવી જોઈએ.*
3. *એના કાર્યનો નમૂનો નથી.*
4. *માત્ર ટૂંકા કોડ ઈનલાઈન કાર્યમાં ઉપયોગ થાય છે.*
5. *જો લાંબો કોડ હોય તો સંગ્રહક તેને અવગણે છે અને તેનો અમલ સામાન્ય કાર્ય તરીકે કરે છે.*

10.7 મૂળભૂત ચલો સાથેનું કાર્ય. (Functions with Default Arguments)

C++ *તેની બધી દલીલો સ્પષ્ટ કર્યા વીના એક કાર્ય કોલ કરવા માટે પરવાનગી આપે છે. આવા કિસ્સાઓમાં કાર્ય, જે પરિમાણ કાર્યના કોલમાં હાજર નથી તેને મૂળભૂત કિંમત આપે છે. મૂળભૂત કિંમતોનો જ્યારે કાર્ય જાહેર થાય છે, ત્યારે જ ઉલ્લેખ થઈ જાય છે. સંગ્રહક* (Compiler) *નમૂના* (Prototype) *માંથી જાણે છે કે કાર્ય કેટલી દલીલો કાર્યને બોલાવવા માટે વાપરે છે.*

દા.ત.

float result (int marks 1, int marks2, int marks 3 = 75) ;

એ પછીનો કાર્ય કોલ Average = result (60,70) ;

marks 1,60 *અને* marks 2 *માટે* 70 *નું મૂલ્ય પસાર થાય છે અને* marks 3 *માં મૂળભૂત કિંમત મળે છે.* Average = result (60,70,80);

ઉપરના કોલમાં marks - 3 *માટે* 80 *નું મૂલ્ય પસાર કરે છે. નોંધ કરવા જેવો મહત્વનો મુદ્દો એ છે કે માત્ર છેવાડાની* (trailing) *દલીલને મૂળભૂત કિંમત હોઈ શકે નીચેના પ્રોગ્રામ ને ધ્યાનમાં લો.*

```
# include <iostream.h>
void repchar ( char = '= ' , int = 30) ;
```

```
void main ()
{
    repchar () ;
    repchar ('*') ;
    repchar ('+' , 45);
}
void repchar (char ch, int x)
{
    for (int i =1;  i < = x ;  i++)
    cout << ch;
    cout << "\n";
}
```

મૂળભૂત દલીલોની પાછળ = ની નિશાની હોય છે જે type ના નામ પછી તરત મૂકવામાં આવે છે. ઉપરના પ્રોગ્રામમાં repchar () કાર્ય બે દલીલો લે છે. તે main () માંથી ત્રણ વખત બોલાવેલ છે.

- *પહેલી વાર કાર્ય દલીલ વગેર બોલાવેલ છે. કાર્ય મૂળભૂત કિંમત ' =' ch ને આપે છે. અને 30 x ને આપે છે અને તે 30 વખત ' =' સ્ક્રીન ઉપર પ્રિન્ટ કરે છે.*
- *બીજી વખત કાર્ય એક દલીલ સાથે બોલાવાય છે. કાર્ય મૂળભૂત કિંમત 30 x ને આપે છે અને '*' ને 30 વખત સ્ક્રીન ઉપર પ્રિન્ટ કરે છે.*
- *ત્રીજી વખત એ બે દલીલ સાથે બોલાવેલ છે. વાસ્તવિક પરિમાણ, મૂળભૂત પરિમાણ કરતા ચડિયાતા હોય છે. તે '+' નિશાની 45 વખત પ્રિન્ટ કરે છે.*

Function Overloading (કાર્ય ઓવરલોડિંગ)

આનો મતલબ છે કે કેટલાક કાર્યના નામનો ઉપયોગ એવા કાર્યો બનાવવા માટે થાય છે જે વિવિધ પ્રકારના કાર્યો કરે છે. આ OOP માં કાર્ય પોલીમોર્ફિઝમ (Polymorphism) તરીકે ઓળખાય છે. કાર્ય અલગ - અલગ પ્રકારની કામગીરી, દલીલ સૂચિ પર આધારિત કરવા જશે. યોગ્ય કાર્યને બોલાવવાનો નિર્ણય દલીલની સંખ્યા અને પ્રકાર પરથી નક્કી થાય છે. દા.ત. એક ઓવરલોડેડ કાર્ય mpy ( ), વિવિધ પ્રકારના ડેટા નીચેની રીતે હેન્ડલ કરે છે.

```
// ઘોષણા (Declaration)
void mpy (int, int) ;          // નમૂનો - 1
void mpy (int, float)          // નમૂનો - 2
void mpy (float, int)          // નમૂનો - 3
void mpy (float, float)        // નમૂનો - 4
```

કાર્ય કોલ્સ

```
cout << mpy (3.5, 10.2) ;        // નૂમના 4 નો ઉપયોગ
cout << mpy (2,5) ;              // નમૂના 1 નો ઉપયોગ
cout << mpy (5.2,6) ;            // નમૂના  3 નો ઉપયોગ
cout << mpy (7.6.2) ;            // નમૂના  2 નો ઉપયોગ
```

કાર્યનો કોલ, પ્રથમ કયા કાર્યના નમૂના (Prototype) સાથે દલીલની સંખ્યા અને પ્રકાર બંધબેસે , છે. તે સરખાવે છે અને પછી યોગ્ય કાર્યનો અમલ કરે છે. શ્રેષ્ઠ સરખામણી અનન્ય હોવી જોઈએ.

કાર્યની પસંદગીમાં નીચેના પગલાઓનો સમાવેશ થાય છે.

(i) કમ્પાઈલર (સંગ્રાહક) વાસ્તવિક દલીલો જે પ્રકારની હોઈ તેવી ચોક્કસ જોડણી શોધવાનો પ્રયત્ન કરે છે, અને તે કાર્ય વાપરે છે.

(ii) જો કોઈ ચોક્કસ જોડણી ના મળે તો કમ્પાઈલર અભિન્ન પ્રમોશન (Integral Promotion) વાસ્તવિક દલીલો માટે વાપરે છે જેવું કે

char to int

float to double

આ કોઈ જોડણી શોધવા માટે કરવામાં આવે છે.

(iii) જ્યારે બેમાંથી એક ઓપ્શન (option) નિષ્ફળ જાય ત્યારે કંમ્પાઈલર વાસ્તવિક દલીલોને રૂપાંતરિત કરવાનો પ્રયત્ન કરે છે અને ત્યારબાદ કાર્ય, કે જેની સરખામણી અન્નય છે તેને વાપરશે. જો રૂપાંતરણ માં એક કરતા વધારે સરખામણીની શક્યતા હોય ત્યારે કંમ્પાઈલર ભૂલ સંદેશો પેદા કરે છે દા.ત. આપણે નીચેના બે કાર્ય વાપરીએ છીએ.

```
long abc (long n) ;
double abc (double x) ;
```

કાર્યનો call જેવો કે abc (10) ના લીધે ભૂલ થાય છે કારણકે int દલીલને long અને double માં રૂપાંતરિત કરી શકાય તેથી તે અસ્પષ્ટ પરિસ્થિતિ ઉભી કરે છે.

(iv) જો બધા પગલા નિષ્ફળ જાય તો કંમ્પાઈલર વપરાશકર્તા વ્યાખ્યાયિત રૂપાંતરણોનો પ્રયત્ન અભિન્ન પ્રમોશન અને નિર્મિત રૂપાંતરણ સાથે અન્નય જોડણી શોધવા માટે કરશે.

પાઠગત પ્રશ્નો 10.2

1. વૈશ્વિક અને સ્થાનિક ચલો શું છે ?
2. ક્યારે તમે કાર્યને ઈનલાઈન તરીકે વ્યાખ્યાયિત કરશો ?
3. ક્યારે તમે મુખ્ય શબ્દ (keyword) return નો ઉપયોગ કાર્ય વ્યાખ્યાયન વખતે કરી શકો.
4. નીચે આપેલ વર્ણનો માટે કાર્યની પ્રોટોટાઈપ રચો.

(અ) abc() કોઈ દલીલ લેતું નથી અને કોઈ પરત મૂલ્ય નથી.

(બ) abc () ફ્લોટ (float)દલીલ લે છે અને int મૂલ્ય પરત કરે છે.

(ક) abc () બે double દલીલો લે છે અને double મૂલ્ય પરત કરે છે.

5. કાર્ય printchar () નીચેની રીતે વ્યાખ્યાયીત થાય છે.

```
void Printchar (char ch = '*', int len = 40)
{
        for (int x = 0; x<len;x++)
        cout << ch ;
        cout << "\n" ;
}
```

તમે કેવી રીતના કાર્ય print char () ને નીચેની નીપજ (output) માટે જગાડશો (invoke)

(અ) '*'40 વખત પ્રિન્ટ કરવા માટે

(બ) '*'20 વખત પ્રિન્ટ કરવા માટે

(ક) '=' 40 વખત પ્રિન્ટ કરવા માટે

(ડ) '=' 30 વખત પ્રિન્ટ કરવા માટે

6. કાર્ય આપેલ છે :

```
int threetimes (int x)
{
   return x * 3 ;
}
```

આના માટે main () કાર્ય લખો જે આ કાર્યને બોલાવે (call કરે.)

7. નીચેના પ્રોગ્રામની નીપજ (output) શું હશે ?

(અ)

```
# include <iostream.h>
void add (float) ;
void main ( )
{
float x = 5.0 ;
add (x) ;
cout << x ;
```

```
}
void add (float x)
{
x = x+2 ;
}
```

(બ)

```
#include <iostream.h>
void add (float) ;
float x = 5.0 ;
void main ( )
{ add (x) ;
cout << x ;
}
void add (float x)
{
x = x+2 ;
}
```

(c)

```
# include < iostream.h >
 float add (float) ;
void main ( )
{
float x = 5.0 ;
float y ;
y = add (x) ;
cout << x << " " << y << "\n"
}
float add (float x)
{
return ( x+2) ;
}
```

(d)

```
# include <iostream.h>
void P1 (int &, int) ;
void main ( )
{
int a = 5 ;
int b = 6 ;
cout<<" output of the program \n" ;
```

```
        cout<<"values of a and b are"<<a<<b<<"\n" ;
        P1 (a,b) ;
        cout<<"values of a and b are" << a<<b ;
    }
        void P1 (int & a, int b)
        {
        a = 7 ;
        b = 10 ;
        }
```

(e)

```
# include <iostream.h>
  int a,b ;
  void try2 (int x, int & y) ;
  void main ( )
  {
        int a = 10 ;
        int b = 7 ;
        try2 (a,b) ;
        cout <<a<<b<<"\n" ;
        }
        void try2 (int x, int & y)
        {
              b = x ;
              y = y+b ;
              x = y ;
              cout <<x<<y<<b<<"\n" ;
        }
```

8. **ખાલી જગ્યા પૂરો.**

(અ) કોઈપણ c++ પ્રોગ્રામ ઓછામાં ઓછા .... કાર્યઓ સમાવે છે.

(બ) દરેક કાર્ય એ કામ કર્યા પછી નિયંત્રણ .... એ પરત થાય છે.

(ક) કોલીંગ પ્રોગ્રામમાં કાર્યની ઘોષણા એ .... કાર્ય તરીકે ઓળખાય છે.

(ડ) કાર્યને .... અને .... બે રીતે બોલાવી શકાય છે.

(ઈ) જ્યારે કાર્ય નો પરત પ્રકાર સ્પષ્ટ ન હોય ત્યારે મૂળભૂત રીતે પરત પ્રકાર ....

.... હોય છે.

(ઉ) નિયંત્રણ પાછું કોલીંગ કાર્ય ને પરત કરવા, આપણે કીબોર્ડ .... ઉપયોગ કરીએ છીએ.

(ઊ) ....કાર્યમાં looping (ફાંસો) ને માન્યતા નથી.

9. **નીચેના વાક્યો સાચા છે કે ખોટા તે જણાવો.**

(અ) c++ પ્રોગ્રામમાં કાર્યોની સંખ્યા પર મર્યાદા છે.

(બ) ચલ જે કાર્યની અંદર ઘોષિત (declare) કરવામાં આવે તેને વૈશ્વિક ચલ કહે છે.

(ક) જયારે કાર્ય કંઈ પરત નથી કરતું ત્યારે પરત પ્રકાર વોઈડ (void) છે.

(ડ) જયારે કાર્યનું કદ નાનું હોય તો ઈનલાઈન કાર્ય પસંદ કરવામાં આવે છે.

(ઈ) સંદર્ભ દ્વારા કોલમાં, કોઈપણ પદાર્થ (object) જે કાર્યની અંદર છે તેમાં બદલાવ, વાસ્તવિક પદાર્થની કિંમતમાં પ્રતિબિંબિત થશે.

(એફ) C++ પ્રોગ્રામમાં એક જ નામના એક કરતા વધારે કાર્યો હોઈ શકે.

(જી) ઈનલાઈન એ C++ માં મુખ્ય શબ્દ (keyword) છે.

(એચ) ઈનલાઈન કાર્યને બોલાવતા પહેલા તે વ્યાખ્યાયિત થયેલું હોવું જોઈએ.

(આઈ) C++ માં કોઈ પણ કાર્ય ઈનલાઈન કાર્ય તરીકે જાહેર કરી શકાય.

(જે) કાર્યના નમૂના (Prototype) માં ચલના નામ સ્પષ્ટ કરવા જરૂરી છે.

10.8 તમે શું શીખ્યા

આ પાઠમાંથી તમે # include નો તમારા પ્રોગ્રામમાં ઉપયોગ શીખ્યા. ઉપરાંત તમે શીર્ષક ફાઈલો જેવી કે ctype.h, math.h, string.h અને stdio.h માં સમાવિષ્ટ કેટલાંક લાઈબ્રેરી કાર્યોનો ઉપયોગ પણ શીખ્યા.

આ પાઠમાં આપણે કાર્યની વ્યાખ્યા અને કાર્ય ને બોલાવવા માટે વિવિધ કાર્ય બોલાવવા માટેના નિવેદનો વિશે શીખ્યા. ઉપરાંત આપણે ઈનલાઈન કાર્ય અને તેના ઉપયોગ વિશે શીખ્યા હવે તમે કાર્યના ખ્યાલ (concept) થી પ્રોગ્રામ લખવાની સ્થિતિમાં હોવા જોઈએ.

10.9 પાઠયાંત પ્રશ્નો (Terminal Questions)

1. પ્રોગ્રામ લખો જે વપરાશકર્તા ને એક અક્ષર દાખલ (નાખવા) પુછશે. દાખલ કરેલ મૂળાક્ષર, અંક કે ખાસ અક્ષર છે તે શોધો.

2. પ્રોગ્રામ લખો જે વપરાશકર્તાને એનું નામ નાખવા (દાખલ) કરવા પૂછે. અપર કેશ (uppercase) અક્ષર ને લોઅરકેશ (lower case) માં રૂપાંતરિત કરે અને તેનું ઉલટું પણ કરે.

3. પ્રોગ્રામ લખો જે વપરાશકર્તા ને અક્ષર દાખલ કરવા કહે. ચકાસો તે મૂળાક્ષર છે કે નહી જો મુળાક્ષર હોય તો ચકાસો કે અપરકેશ (uppercase) માં છે કે લોઅરકેશ (lower

case).

4. નીચેનામાં શું તફાવત છે ?

(1) getchar ( ) & putchar ( )

(૨) gets ( ) & puts ( )

5. પ્રોગ્રામ લખો જે દાખલ કરેલ વાક્યમાંથી અક્ષરોની સંખ્યા ગણે (count).

6. પ્રોગ્રામ લખો જે વપરાશકર્તા એ દાખલ કરેલ વાક્યમાંથી શબ્દોની સંખ્યા ગણે (count).

7. પ્રોગ્રામ લખો જે વાક્યમાંથી A's, E's, I's, O's અને U's ની સંખ્યા ગણે.

8. નીચેના વિશે તફાવત લખો.

(1) વાસ્તવિક અને ઔપચારિક પરિમાણો

(૨) કિંમત (મૂલ્ય) દ્વારા કોલ અને સંદર્ભ દ્વારા કોલ

9. તમારા મતે ઈનલાઈન કાર્યનો અર્થ શું છે ?

10. C++ માં કાર્ય નમૂના (function prototype) ના ફાયદા શું છે ?

11. સંદર્ભથી દલીલ પસાર કરવાનો મુખ્ય ફાયદો શું છે ?

1૨. કાર્યમાં આપણે ક્યારે મૂળભૂત દલીલો ઉપયોગમાં લેવી પડે.

13. કાર્યના ઓવરલોડિંગ નો અર્થ શું છે ?

14. C++ માં પ્રોગ્રામ લખો જે એવા કાર્યને બોલાવે જે આપેલ અંકનો સમઘન, ઈનલાઈન કાર્ય વાપરી પરત કરે.

1૫. C++ માં પ્રોગ્રામ લખો જે કીબોર્ડ થી દાખલ કરેલ બે અંકોમાંથી લાંબો (largest) મોટો અંક પ્રિન્ટ કરે. બે અંક દલીલ તરીકે કાર્ય માં પસાર કરો, મોટો અંક main ( ) કાર્યને પરત કરો.

10.10 પાઠગત પ્રશ્નોના જવાબ

10.1

1. શીર્ષક ફાઈલ કાર્યની ઘોષણા, નમૂના, વર્ગ વ્યાખ્યા વિગેરે ધરાવે છે. તે કમ્પાઈલર (સંગ્રાહક) ને સૂચના આપે છે બીજી ફાઈલ વાંચવા અને તેને સ્રોત કોડ (source) તરીકે સમાવવા.

2. કારણ કે કેટલાક કાર્યો, વર્ગ વ્યાખ્યા વિગેરે જે પ્રોગ્રામમાં વાપરેલ છે તેની જાહેરાત શીર્ષક ફાઈલમાં કરેલ છે.

3. (i) math.h (ii) string.h (iii) conio.h

   (iv) ctype.h

4. C++કમ્પાઈલર માં નિર્મિત કાર્યો, લાઈબ્રેરી કાર્યો કહેવાય છે.

5. બંને કાર્યો વપરાશકર્તા તરફથી એક અક્ષર માટે પૂછે છે. getche( ) કાર્ય અક્ષરનો પડઘો (echoes) પાડે છે અને getch( ) કાર્ય પડઘો પાડતું નથી.

6. *(અ)* તમારૂ નામ નાખો (Enter your name)

   MOHIT

   Hello

   MOHIT

   (બ) Ruby

7. *(અ)* h

   (બ) ctype.h

   (ક) અપરકેશ (uppercase)

   (ડ) S2

   (ઈ) strlen ( )

   (એફ) math.h

8. *(અ)* સાચું *(બ)* ખોટું *(ક)* સાચું *(ડ)* સાચું *(ઈ)* ખોટું
   (એફ) ખોટું *(જી)* સાચું *(એચ)* સાચું *(આઈ)* ખોટું

10.2

1. વૈશ્વિક ચલ બધા જ કાર્યો માટે હાજર હોય છે. સ્થાનિક ચલ માત્ર જે કાર્યમાં એ વ્યાખ્યાયિત હોય ત્યાં હાજર હોય છે.
2. જયારે કાર્યનો કોડ નાનો હોય છે. દા.ત. કાર્ય એક કે બે લીટી સમાવે છે તે ઈનલાઈન કહેવાય છે.
3. જયારે કાર્ય કોઈ મુલ્ય પરત કરે છે ત્યારે return મુખ્ય શબ્દનો એની વ્યાખ્યા સમયે ઉપયોગ થાય છે.
4. *(અ)* void abc( )

   (બ) int abc (float) ;

   (ક) double abc (double,double) ;
5. *(અ)* void printchar ( ) ;

   (બ) void printchar ('*', 20) ;

   (ક) void printchar ('=') ;

નોંધ

(ડ) void printchar ('_', 30) ;

6.
```
void main ( )
  {
    int y, l ;
    cout <<"Enter value" ;
    cin >>y ;
    l = threetimes (y) ;
    cout << l
  }
```

7. (અ) 5

   (બ) 5

   (ક) 5 7

   (ડ) output of the Program

   values of a and b are 5 6

   values of a and b are 7 6

   (ઈ) 17 17 10

   10 17

8.

   (અ) 1

   (બ) કોલીંગ કાર્ય વિધાન પછીના વિધાન પર

   (ક) નમૂનો (Prototype)

   (ડ) કિંમત દ્વારા કોલ અને સંદર્ભ દ્વારા કોલ

   (ઈ) int

   (એફ) return

   (જી) inline

9. (અ) ખોટું (બ) ખોટું (ક) સાચું (ડ) સાચું

   (ઈ) સાચું (એફ) સાચું (જી) સાચું (એચ) સાચું

   (આઈ) ખોટું (જે) ખોટું

11

એરે

11.1 પ્રસ્તાવના

ક્રમમાં ગોઠવાયેલા (સંગ્રહાયેલા) સમાન તત્વોને એરે કહે છે. કેટલીક વાર એક જ પ્રકારના ચલો મોટી સંખ્યામાં સંગ્રહવા જરૂરી છે, તેના માટે આપણે એરે (array) વાપરી શકીએ. ઉદાહરણ તરીકે :

int A [5];

ઉપરોક્ત વિધાન એરે A ની ઘોષણા કરે છે જે પાંચ પૂર્ણાંક પ્રકારની માહિતી સંગ્રહ કરવા માટે જગ્યા અનામત રાખે છે. આ સ્થાનો છે A [0], A [1], A [2], A [3], અનેA [4]. ચોરસ કૌંસની અંદર રહેલ સંખ્યાને ઈન્ડેક્સ અથવા સબસ્ક્રિપ્ટ) કહેવાય છે. એરે એક પરિમાણીય (એક સબસ્ક્રિપ્ટ) અથવા બે પરિમાણીય (બે સબ સ્ક્રિપ્ટ) હોઈ શકે છે.

11.2 હેતુઓ

આ પાઠ ભણી ગયા પછી તમે નીચેનાઓ માટે સક્ષમ હશો:

- *એરેનો ખ્યાલ સમજાવી શકશો*
- *એરેને વ્યાખ્યાયિત, એકસેસ અને એરેના તત્વોનો પ્રારંભ (initialize) કરી શકશો*
- *એરે પર સામાન્ય કામગીરી કરવા*
- *બે પરિમાણીય એરે સમજાવવા*
- *વ્યાખ્યાયિત, વપરાશ અને બે પરિમાણીય એરે તત્વોનો પ્રારંભ કરવો.*

11.3 એક પરિમાણીય એરેનો આરંભ (Initialization of one Dimensional Array)

એરેને ઘોષણા (declaration)ની સાથે initialize પણ કરી શકાય છે. એરેના initialization માટે એરેના તત્વો (elements) ને અલ્પવિરામ દ્વારા અલગ પાડી કૌંસની અંદર મુકવા પડે છે.

```
int A[5] = {1,2,3,4,5};
```

એરેની સાઈઝ (કદ)ને ખાલી છોડી શકાય છે. કંમ્પાઈલર એરેનું કદ જાતે ગણી લેશે.

```
int B [ ] = {6, 7, 8, 9, 10};
```

11.4 સ્ટ્રિંગનો આરંભ (Initialization of string)

અક્ષરોનો એરે જે કેરેક્ટર સ્ટ્રિંગ (character string)તરીકે ઓળખાય છે, જેનો આરંભ (initialization) સ્ટ્રિંગને અવતરણમાં મુકીને કરી શકાય.

```
char A [20] = " computer science";
char B [ ] = "calcutta";
```

સ્ટ્રિંગએ અક્ષરોની શ્રેણી છે જે મેમરીમાં સળંગ બાઈટ્સ સ્વરૂપે સંગ્રહિત હોય છે. આ સૂચિત કરે છે કે આપણે સ્ટ્રિંગને અક્ષરોના એરેમાં સંગ્રહ કરી શકીએ, જેમ દરેક અક્ષર એરેના તત્વ (element)માં રાખવામાં આવે છે. દરેક C++ સ્ટ્રિંગમાં છેલ્લો અક્ષર નલ (NULL) હોય છે. તેથી

```
char city [ ] = {'c', 'a', 'l', 'c', 'u', 't', 't', 'a' };
```

એ અક્ષરોનો એરે છે જ્યારે

```
char city [ ] = {'c', 'a', 'l', 'c', 'u', 't', 't', 'a', '\0'};
```

એ સ્ટ્રિંગ છે કારણ કે તે નલ (Null)અક્ષરથી અંત પામે છે.

11.5 એરેની પ્રક્રિયા (Processing an Array)

એરે પર વિવિધ શક્ય કામગીરી છે :

- એક છેડાથી બીજા છેડા સુધી જવું (Traversal)
- શોધવું (Searching)
- ગોઠવવું (Sorting)
- ઉમેરવું (Insertion)
- કાઢી નાખવું (Deletion)

ટ્રાવર્સલ (Traversal)

તેનો અર્થ થાય છે એરેના દરેક સ્થાનને ચકાસવા, જે બતાવવા (ડિસ્પ્લે)ના હેતુ માટે હોઈ શકે. એવો પ્રોગ્રામ વિચારો જે વપરાશકર્તા પાસેથી પાંચ કિંમતો વાંચી અને તેમાંથી મહત્તમ કિંમત

શોધે.

```
# include<iostream.h>
void main ( )
{        int T, A [5], I ;
         cout << "Enter five values";
         for (I =1; I<5; I ++)
         cin >> A [i];
         T = A [o];
         for (I=1; I<5: I + +)
         {
         if (T< A [I];
         T = A [ I ] ;
}
cout << "Maximum value" << T;
}
```

શોધવું (Searching)

આ પધ્ધતિ વપરાશકર્તા ધ્વારા દાખલ કરેલ ડેટા એરેમાં હાજર છે કે નહિં તે શોધી આપે છે. શોધવાની પધ્ધતિના બે પ્રકાર છે.

(i) *સુરેખ અથવા ક્રમાંકિત શોધ* (linear or sequential search)

(ii) *દ્વિસંગી શોધ* (Binary search)

સુરેખ અથવા ક્રમાંકિત શોધ *(*Linear or sequetial search*)*

આ પધ્ધતિ ધીમી, ઓછી અસરકારક અને ક્રમમાં ગોઠવાયેલ ના હોય તેવી યાદી પર કામ કરે છે. આપણે જે ડેટા શોધીએ છે તે જો યાદીમાં ના હોય તો આપણને તેની જાણ યાદીના અંતમાં થાય છે.

```
// Linear search
# include<iostream.h>
void main ()
{
int A[5], I, data, flag=0;
cout << "Enter five values";
for (I=o; I<5; I++)
cin>>A[I];
cout <<  "Enter data to be searched";
```

નોંધ

```
cin>> data;
for (I= 0 ;  I < 5 ;  I++ )
{
if (A [I]= = data)
flag = 1;
}
if (flag = = 1)
cout << "Data present";
else
cout << "Data not present";
}
```

દ્વિસંગી શોધ (Binary Search)

આ પધ્ધતિ માટે એરે ચડતા અથવા ઉતરતા ક્રમમાં હોવો જરૂરી છે. આ પધ્ધતિમાં પ્રારંભિક અને અંતિમ સ્થળોને ગણતરીમાં લઈ મધ્ય સ્થાન શોધાય છે અને મધ્ય સ્થાને આવેલી કિંમતને હાલના ડેટા સાથે સરખાવે છે. ધારો કે યાદી ચડતા ક્રમમાં ગોઠવાયેલ છે. જો ડેટા મધ્ય કરતાં ઓછો હોય તો તે માહિતી ઉપલા ભાગમાં હાજર છે નહિં તો માહિતી નીચલા અડધા ભાગમાં હાજર છે. પ્રારંભિક અથવા અંતિમની કિંમત બદલાશે. મધ્યની કિંમત ફરીથી શોધાશે અને શોધવાનું ચાલુ રહેશે જ્યાં સુધી ડેટા મળે અથવા પ્રારંભિક એ અંતિમ કરતાં વધી જાય. પ્રારંભિક, અંતિમ અને મધ્યની કિંમતો જે કિંમત 35 ને ચડતા ક્રમમાં ગોઠવાયેલા 9 તત્વોમાં શોધે છે તે નીચે આપેલ છે.

એરેને કિંમત લેવા દો :

	10	15	18	20	27	30	35	40	45
સ્થળ	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)

Initial	Final	Mid = $\frac{\text{Initial + Final}}{2}$	A [Mid]
0	8	4	27
5	8	6	35
			શોધ સફળ રહી

```
//Binary search
# include<iostream.h>
const int N= 10 ;
void main ()
{int A [N], I, inital, final, mid, data;
cout<< "Enter ten values in ascending order";
for (I= 0 ; I < N ; I+ +)
```

```
        cin >> A [I];
        cout<< "Enter data to be searched";
        cin>> data;
        initial=0;
        final=N-1;
        mid = (initial + final)/2;
        while (initial<=final) && (A[mid]!=data))
        {
                if (A [mid]>data)
                final=mid-1;
                else
                initial = mid + 1;
        }
        if (A[mid] = = data)
        cout<< "data is present";
        if (initial>final)
        cout << "data not present in list";
        }
```

દ્વિસંગી શોધનો ફાયદો એ છે કે દરેક શોધ યાદીને અડધી બનાવે છે. 10,000 *નામોની યાદી માત્ર* 12 *શોધમાં શોધી શકાય છે.*

ગોઠવવું (sorting)

તે એક ચડતા અથવા ઉતરતા ક્રમમાં યાદી ગોઠવવાની પધ્ધતિ છે.

*બબલ સોર્ટ (*Bubble sort*)*

પાંચ સ્થાન ધરાવતા એરેને ચડતા ક્રમમાં ગોઠવવા માટે :

	-1	9	5	0	2
Index	0	1	2	3	4

આ સોર્ટિંગ પધ્ધતિમાં A[0] *ને* A[1] *ની સાથે સરખાવામાં આવે છે. જો* A[0] *ની કિંમત* A[1] *કરતાં વધુ હોય તો તેમની કિંમત એક બીજા સાથે બદલાવામાં* (swap) *આવે છે. પછી* A[1] *ને* A[2] *ની સાથે સરખાવાય છે,* A[2] *ને* A [3] *સાથે સરખાવાય છે, અને* A[3] *ને* A[4] *સાથે સરખાવાય છે. દરેક કિસ્સા જ્યારે જગ્યા પરની કિંમત* i +1 *કરતાં વધુ હોય તો તેમની કિંમત એક બીજા સાથે બદલવામાં આવે છે. સમગ્ર પ્રક્રિયાની* N-1 *વખત પુનરાવર્તન કરવામાં આવે છે જ્યાં* N *એ એરેમાં રહેલ તત્વોની સંખ્યા છે.*

```
#include <iostream.h>
const int N=5;
```

```
void main ()
{
        int A[N], I, j, T;
        cout << " Enter Values" ;
        for (I = 0;  I < N ;  I + +)
        cin >> A[I];
// sorting
for (I = 0; I< N-1; I++)
for (j=0; j<N-1; j + +)
if (A [j] >  A [ j  +  1 ] )
{
        T = A [j];
        A[j] = A [j + 1];
        A [j+1] = T;
}
cout << "sorted array is";
for (I = 0;  I < N ;  I ++)
cout  <<  A  [I] ;
}
```

પસંદગી સોર્ટ (Selection sort)

વિચારો કે N તત્વો ધરાવતા એરેના તત્વોને ચડતા ક્રમમાં ગોઠવવા છે. શરૂઆતમાં પ્રથમ તત્વ (element) બીજા તત્વો સાથે સરખાવાય છે અને તે સૌથી નાની કિંમત ધરાવે છે. બીજી વખત (next pass), બીજા નંબરના તત્વને બીજા બધા સાથે સરખાવામાં આવે છે જેથી તે સૌથી નાની કિંમત ધરાવે. આ પ્રક્રિયાનું આખા એરે માટે પુનરાવર્તન કરવામાં આવે છે.

```
# include<iostream.h>

const int N=5;

void main ( )

{

        int A [N],  I,  j,  T ;

        cout << "Enter values";
```

```
        for ( I = 0 ;  I < N ;  I + +)
cin>> A [I];
// sorting
for (I = 0; I < N -1 ;  I + +)
for  (j = 1; J < n ; j + +)
if  (A  [I] >  A  [j] )
{
        T = A [I];
        A [I] =  A [j] ;
        A [j] =  T ;
}
// printing the sorted data
for (I = 0 ;  I < N ;  I++)
cout << A [I];
}
```

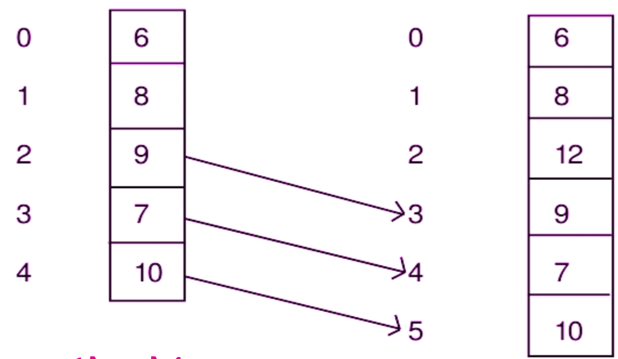
ઉમેરવું (Insertion)

આનો અર્થ થાય છે કે ડેટાને વચ્ચે અથવા એરેના અંતમાં ઉમેરવા. જો ડેટાને કોઈ આપેલી માહિતીની પાછળ ઉમેરવો હોય તો પહેલાં તે માહિતીનુ સ્થાન શોધ પ્રક્રિયા ઉપયોગ કરી શોધો અને પછી ઉમેરવાની પ્રક્રિયા અમલમાં મૂકો.

```
#  include < iostream.h >
void main()
{
int x [20] ;
int i, loc, n, data ;
cout<< " Enter the no. of  elements " ;
cin >> n;
for (I = 0 ;  I < n ; I + +)
{
cout << "Enter the array element";
cin >> x [I];
}
cout << "Enter the location after which data is to be inserted";
cin>> loc;
for(I=n-1; I >= loc ; I --)
x[i+1] = x [1];
cout<< "Enter the new data to be added";
cin>> x [loc];
```

```
n++;
cout << "Array elements after insertion";
for (I = 0 ;  I < n ;  I ++)
cout<< x [I];
}
```

ડેટા 12 ને 2 નંબરના સ્થળ પર ઉમેરીએ.



ઉમેરતા પહેલાં

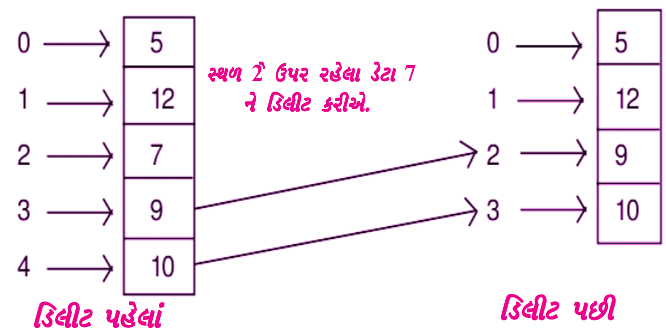
ઉમેર્યા પછી

કાઢી નાખવું (Deletion)

તે માહિતી દૂર કરે છે. પ્રથમ જે સ્થળ પરની માહિતી કાઢી નાખવાની હોય તેને યોગ્ય શોધ પ્રક્રિયાથી શોધવામાં આવે છે અને પછી તે કિંમતને તે સ્થળ પરથી કાઢી નાખવામાં આવે છે.

```
# include<io stream.h>
void main ()
{
int x [20];
int I, j, n, loc, data;
cout << " Enter the no. of elements " ;
cin>>n;
for (I=0;  I < n ;  I ++)
{
cout << "Enter value";
cin >> x [I];
}
cout << "Enter the location to be deleted";
```

```
cin >> loc;
if (loc ! = 0)
{
data= x [loc];
for (j=loc; j<n-1; j ++)
x [J] = x [j + 1];
}
n =  n - 1 ;
cout<< "Elements after deletion";
for (I = 0 ; I < n ; I+ +)
cout << x [I];
}
```



પાઠગત પ્રશ્નો 11.1

1. *એરે શું છે?*

2. *તમારા મતે એરેના ઈન્ડેક્ષનો મતલબ શું છે?*

3. *પાંચ તત્વો ધરાવતા એરેમાં તત્વોની સંખ્યા ________ થી ________ હોય છે.*

4. *એવું નિવેદન લખો જે વિદ્યાર્થી માટે એક પરિમાણીય એરે વ્યાખ્યાયિત કરે અને જેમાં પાંચ પૂર્ણાંક પ્રકારના તત્વો રાખી શકાય.*

5. *નીચે જાહેર કરેલા એરેમાં કેટલા તત્વો છે તે શોધો અને દરેક એરેને સંગ્રહ કરવા કુલ કેટલા બાઈટ્સ જોઈશે તે શોધો.*

(અ) int A[20];

(બ) char z [18];

6. *નીચેના પ્રોગ્રામનો આઉટપુટ શું હશે?*

(અ)
```
# include<iostream.h>
void main ( )
{
int a [5], t;
for (i= 0 ;  i < 5 ;  i  ++)
a [i] = 5x i;
for (i = 0 ;  i < 5 ;  i ++)
cout << a [i];
}
```

(બ)
```
# include < iostream.h>
void main ( )
{
    char title [ ] = "computer";
    for (int i =0; title [i]; i ++)
    cout << "\n" << (title + i);
}
```

11.6 બે પરિમાણીય એરે *(Two Dimensional Array)*

તેને બે સબસ્ક્રિપ્ટ અથવા ઈન્ડેક્ષ હોય છે, પ્રથમ એ હરોળ માટે અને બીજો સ્તંભ માટે. ઉદાહરણ તરીકે:

```
int A [5] [4] ;
```

ઉપરનાને પાંચ હરોળ અને ચાર સ્તંભ હશે. કુલ વીસ (20) તત્વો હશે. આ વીસ તત્વોના સબસ્ક્રિપ્ટ છે:

```
A [0] [0]    A [0] [1]    A [0] [2]    A [0] [3]
A [1] [0]    A [1] [1]    A [1] [2]    A [1] [3]
A [2] [0]    A [2] [1]    A [2] [2]    A [2] [3]
A [3] [0]    A [3] [1]    A [3] [2]    A [3] [3]
A [4] [0]    A [4] [1]    A [4] [2]    A [4] [3]
```

બે પરિમાણીય એરેનો પ્રારંભ *(Initialization of two Dimensional Array)*

એરેનો પ્રારંભ તેની ઘોષણા સમયે કરી શકાય છે. ઉદાહરણ તરીકે:

```
int A[2] [4] = {1,2,3,4,5,6,7,8};
```

વધુ સ્પષ્ટતા માટે,

```
int A [2] [4] = {{1,2,3,4}, {5,6,7,8}};
```

ઉપરોક્ત માહિતીનું જૂથ કરી શકાય છે. આંતરિક કૌંસ કમ્પાઈલર દ્વારા અવગણવામાં આવે છે. બે પરિમાણીય એરે માટે, બે લુપની જરૂર પડે છે. નીચેનો પ્રોગ્રામ બે પરિમાણીય એરેમાંથી મહત્તમ કિંમત શોધશે.

```
# include < iostream.h >
const int M = 5;
void main ( )
{ int A [M] [M], i, j, T;
        for (i = 0; i<=M-1; i ++)
        for (j=o; j<=M-1; j++)
        T = A [0] [0];
        for ( i=o; i<=M-1; i ++)
        for (j=o; i<=M-1; i ++)
{
                if (T<A[i] [j])
                T= A [i] [j] ;
}
        cout << "Largest value" << T<< "\n";
}
```

પાઠગત પ્રશ્નો 11.2

1. *એવા બે પરિમાણીય એરે* A *ની ઘોષણા કરો જેમાં બે હરોળ, ત્રણ સ્તંભ હોય અને તે પૂર્ણાંક માહિતીનો સંગ્રહ કરી શકે.*
2. *પ્રશ્ન એક માં ઘોષણા કરેલ એરેમાં કેટલા તત્વો હશે?*
3. *પ્રશ્ન એકમાં ઘોષણા કરેલ એરેને મેમરીમાં કેટલા બાઈટ્સ જોઈએ*
4. *નીચેની એરે ઘોષણા જુઓ. તે દરેકમાં કુલ કેટલા તત્વો હશે અને દરેક એરેને કુલ કેટલા બાઈટ્સ જોઈશે તે શોધો.*

   (અ) int x [5] [10];

   (બ) long y [5] [10];

11.7 તમે જે શીખ્યાં (What you have learnt)

આ પાઠમાં આપણે શીખ્યા કે કેવી રીતે એક અને બે પરિમાણીય એરેને વ્યાખ્યાયિત કરવા, તેમનો ઉપયોગ કરવો અને તેમાં કિંમતો આપવી. આપણે એક પરિમાણીય એરે પર થતી વિવિધ

પ્રક્રિયાઓની ચર્ચા કરી.

11.8 પાઠ્યાંત પ્રશ્નો

1. એરેની જરૂરીયાત શી છે ?
2. તમારા મતે શોધવાનો મતલબ શું છે?
3. સુરેખ (લીનીયર) અને દ્વિસંગી (બાયનરી) શોધનો એક લાભ અને એક ગેરલાભ જણાવો.
4. તમારા મતે ગોઠવવું (સોર્ટ) નો અર્થ શું છે?
5. કઈ પધ્ધતિ સારી છેઃ બબલ સોર્ટ કે સીલેકશન સોર્ટ ? કેમ ?
6. એવો પ્રોગ્રામ લખો જે વીસ (20) ફ્લોટ (float) કિંમતનો એક પરિમાણીય એરેમાં સંગ્રહ કરે અને નીચેનુ શોધોઃ
   (i) શૂન્ય કરતા વધારે કિંમતોની સંખ્યા.
   (ii) શૂન્ય સમાન કિંમતોની સંખ્યા.
   (iii) શૂન્ય કરતા ઓછી કિંમતોની સંખ્યા.
7. એવો પ્રોગ્રામ લખો જે બે પરિમાણીય એરે A [N] [N] ના બે કર્ણનો સરવાળો શોધે.
8. એવો પ્રોગ્રામ લખો જે નીચેની શોધનો ઉપયોગ કરીને 10 તત્વો ધરાવતા એક પરિમાણીય એરેમાં વપરાશકર્તાએ આપેલી માહિતીને શોધેઃ
   (i) સુરેખ (લીનીયર) શોધ
   (ii) દ્વિસંગી (બાયનરી) શોધ
9. એવો પ્રોગ્રામ લખો જે નીચેની પધ્ધતિનો ઉપયોગ કરીને આપેલા 10 નંબરને ઉતરતા ક્રમમાં ગોઠવે.
10.
   (i) બબલ સોર્ટ
   (ii) સિલેકશન સોર્ટ

11.9 પાઠગત પ્રશ્નોના જવાબો (Feedback to In-text questions)

પાઠગત પ્રશ્નો 11.1

1. એરે એ એક જ પ્રકારની માહિતીનો સંગ્રહ છે.
2. એરે ઈન્ડેક્ષ એ એરેમાં સ્થાન બતાવે છે.
3. 0, 4
4. int student [5];
5. (અ) કુલ તત્વો - 20
   કુલ બાઈટ્સ - 40

(બ) *કુલ તત્વો* - 18
કુલ બાઈટ્સ - 18

6. (અ) 0 5 10 15 20

(બ) computer
omputer
mputer
puter
uter
ter
er
r

પાઠગત પ્રશ્નો 11.2

1. int A [2] [3];

2. 6

3. 12

4. (અ) *કુલ તત્વો* - 50

કુલ બાઈટ્સ - 100

(બ) *કુલ તત્વો* - 50

કુલ બાઈટ્સ - 200

12

સ્ટ્રક્ચર, ટાઈપડેફ અને એન્યુમરેટેડ ડેટા ટાઈપ

12.1 પ્રસ્તાવના

માહિતીના સંગ્રહ માટે, અલગ કે એક જ પ્રકારના ડેટા ટાઈપની જરૂર રહે છે. તે માટે સ્ટ્રક્ચરનો ઉપયોગ થાય છે. આ પાઠમાં C++ લેન્વેજના ત્રણ મુખ્ય સ્ટેટમેન્ટ વિશે વાત કરવામાં આવી છે.

12.2 હેતુઓ

આ આખા પાઠનું પઠન કર્યા બાદ આપણે નીચે મુજબના મુદ્દા શીખીશું.

- વેરીએબલ જાહેર કરવા, સ્ટ્રક્ચરના મેમ્બરનો ઉપયોગ કરવો તથા તેને કિંમતો એસાઈન કરવી.
- નેસ્ટેડ સ્ટ્રક્ચર વિશે માહિતી સમજવી.
- પ્રોગ્રામમાં ટાઈપડેફ સ્ટેટમેન્ટની ઓળખ.
- ઈનમ સ્ટેટમેન્ટને ડીફાઈન કરવું અને તેનો ઉપયોગ.

12.3 સ્ટ્રક્ચર

સ્ટ્રક્ચર એ સામાન્ય વેરીયેબલ કે જે એક પ્રકારના કે અન્ય પ્રકારના ડેટા ટાઈપનો સમૂહ છે. સ્ટ્રક્ચરમાના ડેટાને સ્ટ્રક્ચરનાં સભ્યો તરીકે ઓળખવામાં આવે છે. નીચેના ઉદાહરણ પર ધ્યાન આપે છે.

```
# include < iostream.h>
struct student
{
        char name [20] ;
        int marks ;
};
        void main ()
{
        student S1, S3 ;
        cin >> S1. name >> S1. marks;
        cout << S1. name << S1. marks ;
        student S2 = { "ANKIT", 32};
        cout << S2. name <<S2. marks ;
        S3 = S2 ;
        cout << S3. name <, S3. marks;
}
```

12.3.1 સ્ટ્રક્ચરના વેરીએબલ

S1 અને S2 સ્ટ્રક્ચર student ના વેરીએબલ તરીકે ઓળખવામાં આવે છે તેમને તમામ સભ્યોને સંગ્રહવા માટે મેમરીમાં પૂરતા પ્રમાણમાં સ્પેશ આપવામાં આવે છે. (ઉપરના પ્રોગ્રામ માટે દરેક સ્ટ્રક્ચરના વેરીયેબલને 22 bytes આપવામાં આવશે)

12.3.2 ડેટા મેમ્બરનો ઉપયોગ

ડેટા મેમ્બરનો ઉપયોગ નીચે આપેલ માળખા દ્વારા કરી શકાશે.

structure variable. member name

12.3.3 સ્ટ્રક્ચર વેરીયેબલનું ઈનિશીયલાઈઝેશન

વેરીએબલને જાહેર કરવાના સમયે જ ઈનિશીયલાઈઝેશન કરવામાં આવ્યું. જેમ કે,

Student S2 = { "ANKITA", 32} ;

12.3.4 એસાઈનમેન્ટ સ્ટેટમેન્ટમાં સ્ટ્રક્ચર વેરીએબલ

નિવેદન

S3 = S2 ;

S2 ના તમામ મેમ્બરની કિંમતને S3 ના તમામ મેમ્બરને એસાઈન કરશે. મોટા સ્ટ્રક્ચરમાં વધુ પ્રમાણમાં મેમ્બર્સ હોય છે, જેને એસાઈન કરવા કમ્પાઈલરને એકધારા પ્રમાણમાં કાર્યરત

નોંધ

રહેવું પડે છે. ખાસ ધ્યાન રાખો કે એક સ્ટ્રકચર વેરીએબલ બીજાને ત્યારે જ એસાઈન કરી શકશે જ્યારે બંને સ્ટ્રકચર વેરીયેબલ એક જ સ્ટ્રકચરના હશે, અન્યથા કંપાઈલર એરર આપશે.

12.3.5 સ્ટ્રકચરની અંદર સ્ટ્રકચર

સ્ટ્રકચરની અંદર સ્ટ્રકચરને નેસ્ટેડ સ્ટ્રકચર કહે છે. નીચેના પ્રોગ્રામ તરફ ધ્યાન આપો તેમાં એક ડેટા ફિલ્ડ સ્ટ્રકચર છે.

```
# include <iostream. h>
# include <stdio.h>
struct today
{
        int month ;
        int date ;
        int year ;
} ;
struct biodata
{
        char name [20] ;
        today date _ of _ birth ;
        char add [30] ;
}
void main ()
{
        biodata b ;
        cout << 'Enter name" ;
        gets (b.name) ;
        cout << "Enter birth month" ;
        cin >> b.date_ of_ birth.month ;
        cout << "Enter birth date";
        cin>> b. date _ of _ birth. date;
        cout << "Enter birth year";
        cin >> b. date _ of_ birth. year ;
        cout << "Enter Address";
        gets (b.add) ;
        cout << "Name" << b.name <<. "\ n";
        cout << "Month" <<b.date_ of_ birth. month <<"\ n";
        cout << "Date" < b. date_ of_ birth. date <<" \ n";
        cout << " year" << b.date_ of_ birth.year <<"\n" ;
```

```
        cout << "Address" << b.add << "\n";
}
```

12.3.6 નેસ્ટેડ સ્ટ્રકચર મેમ્બરનો ઉપયોગ

સ્ટ્રકચરની અંદર સ્ટ્રકચર આવેલ હોવાથી આપણે (.) ડોટ ઓપરેટરનો ઉપયોગ બે વખત કરવો પડશે.

```
        cin>> b.date_ of _ birth. month;
```

12.3.7 નેસ્ટેક સ્ટ્રકચરને ઈનિશીયલાઈઝ કરવું

સ્ટ્રકચર વેરીયેબલના ડીકલેરેશન વખતે તેને ઈનિશીયલાઈઝ થશે.

```
biodata bd = { "MOHIT", {3,21,1991}, "Rohini"} ;
```

અંદરના કૌંસ ફકત કિંમતને અલગ પાડવા વપરાય છે જેને કંપાઈલર ગણકારશે નહિ.

પાઠગત પ્રશ્નો 12.1

1. *સ્ટ્રકચર શું છે ?*
2. *સ્ટ્રકચર એરે થી કઈ રીતે અલગ પડે છે ?*
3. *સ્ટ્રકચર વેરીયેબલને મેમરીમાં જગ્યા આપવામાં આવે છે. આ વિધાન સાચું છે કે ખોટું ?*
4. *સ્ટ્રકચરના ડેટા મેમ્બરનો ઉપયોગ કરવા માટેનું સૂત્ર કયું છે ?*
5. *ડેટા ટાઈપ એસાઈન કરતી વેળાએ જ સ્ટ્રકચરનું ઈનિશીયલાઈઝેશન કરવામાં આવે છે ? સાચું કે ખોટું ?*
6. *નેસ્ટીંગ ઓફ સ્ટ્રકચર શું છે ?*
7. int *પ્રકારના ત્રણ વેરીએબલ* hrs, mns *અને* secs *સાથેનું સ્ટ્રકચરનું માળખું બનાવો અને તેનું નામ* structure time *રાખો.*
8. time *સ્ટ્રકચરના વેરીયેબલના મેમ્બરને* 11 *કિંમત એસાઈન કરવા માટેનું વિધાન લખો.*
9. *જો ત્રણ વેરીયેબલ, સ્ટ્રકચર,* time *પ્રકારના જાહેર કરેલ હોય અને આ સ્ટ્રકચરમાં* int *પ્રકારના ત્રણ વેરીયેબલ જાહેર કરેલ હોય તો એક સાથે ત્રણ સ્ટ્રકચર વેરીયેબલ મેમરીમાં કુલ કેટલા* bytes *રોકશે ?*
10. *નીચે આપેલ માહિતી પરથી* C++ *માં નેસ્ટેડ સ્ટ્રકચર લખો. યોગ્ય ડેટા ટાઈપનો ઉપયોગ કરવો.*

Name	Class	Roll	Address
			House No. Street City State Pincode

નોંધ

સ્ટ્રક્ચરના વેરીએબલ જાહેર કરો તથા નેસ્ટેડ સ્ટ્રક્ચરના તમામ વેરીએબલને કિંમત એસાઈન કરો.

11. *નીચેના પ્રોગ્રામના માળખામાં સ્ટ્રક્ચરની તમામ ફીલ્ડમાં કિંમત એસાઈન કરવાના વિધાન લખો.*

```
struct date
{
int dd, mm,yy ;
} ;
struct donor
{
        char N [20]
        date dob ;
} ;
void main ( )
{
        donar person ;
        _ _ _ _ _ _ _ _ _
        _ _ _ _ _ _ _ _ _
}
```

12. *નીચેના પ્રોગ્રામનો આઉટપુટ લખો.*

(a)

```
#include < iostream.h>
void main ()
{
struct rollno
{
int roll ;
};
rollno r ;
r. roll = 100 ;
cout << "roll" = <<r. roll ;
}
```

(b)

```
# include <iostream. h>
void main ()
{
struct student
{
        int roll ;
        int age ;
} ;
```

```
        student S1 = { 100, 15} ;
        student S2 = S1 ;
if (S1 = = S2)
        cout << " The structures are equal" ;
        else
        cout << " The structures are not equal" ;
}
```

12.4 ટાઈપ ડેફ :

પહેલેથી હયાત ડેટા ટાઈપ માટે નવ ડેટા ટાઈપ જાહેર કરવા ટાઈપ ડેફનો ઉપયોગ થાય છે. તે સ્ટાન્ડર્ડ ડેટા ટાઈપનું અન્ય નામ આપવા ઉપયોગી છે. વિસ્તૃત નામ દ્વારા તે કોડનું સ્વ - ડોકયુમેન્ટીંગનું કામ કરે છે. (બેલટેસ સમજણ માટે) સ્ટાન્ડર્ડ ડેટા ટાઈપ માટે.

સામાન્ય માળખું :

typedef existing datatype new datatype ;

ઉદાહરણ તરીકે :

typedef float real;

હવે, પ્રોગ્રામમાં float *ના બદલે* real *ડેટા ટાઈપ વાપરી શકાશે, માટે નીચેનું વિધાન માન્ય છે.*

real amount ;

12.2 પાઠગત પ્રશ્નો

1. *ટાઈપડેફ વિધાનનું મુખ્ય કાર્ય શું છે ?*
2. *ટાઈપડેફ વિધાન પ્રોગ્રામમાં શા માટે વપરાય છે ?*
3. *નીચેના વિધાનોનો અર્થ શું છે ?*

   typedef int integer ;

12.5 ઈન્યુમેરટેડ ડેટા ટાઈપ

નીચે મુજબની લાક્ષણિકતાઓ ઈનમ ડેટા ટાઈપમાં છે :

- *તે ઉપયોગકર્તા દ્વારા વ્યાખ્યાયિત છે.*
- *જો તમે યાદીની કિંમત જાણતા હો તો તે ખૂબ જ ઉપયોગી થાય છે.*
- *યાદી ઉપયોગકર્તા દ્વારા દાખલ નહી કરી શકાય કે તેને સ્ક્રીન પર દશાવી પણ શકાશે નહી.*

દા.ત.

```
enum months { jan, feb, mar, apr, may } ;
enum days {sun, mon, tue, wed, thu};
enum toys { cycle, bicycle, scooter};
```

ઈનમ સમૂહમાંના માન્ય નામની કિંમતોને Integer Constant *દ્વારા આંતરિક રીતે જાહેર કરવામાં આવે છે. પ્રથમ નામને* 0 *(શૂન્ય) કિંમત અપાશે અને બીજી કિંમતને* 1 *અને તે મુજબ અન્ય.*

દા.ત.

Jan = 0, feb = 1, mar =2, apr =3, may =14

'=' સાઈન દ્વારા કિંમતના ક્રમને બદલી શકાય છે.

enum months { jan = 1, feb, mar, apr, may}

અહીં

jan =1, feb =2, mar = 3, apr = 4, mar = 5

પછીના એલીમેન્ટની કિંમત પહેલાંની કિંમતમાં એક ઉમેરીને દર્શાવશે.

દા.ત.

```
# include < iostream.h>
enum months { jan, feb, mar, apr, may} ;
void main ()
{
months m1, m2;
m1 = jan ;
m2 = apr ;
int diff = m2-m1 ;
cout << "months between" << din << "\n";
if ( m1> m2)
cout <<" m2 Comes before m1";
}
```

ઉપરના પ્રોગ્રામનું આઉટપુટ

Months between 3

કારણ કે ઈનમ સીરીઝના મેમ્બર્સને મેમરીમાં ઈન્ટીજર કોન્સ્ટન્ટ તરીકે સંગ્રહવામાં આવે છે.

12.3 પાઠગત પ્રશ્નો

1. ઈનમ ટાઈપને અંદરખાને Integer ટાઈપ ગણવામાં આવે છે. ખરું કે ખોટું ?
2. નીચેના વિધાનનો અર્થ શું થાય છે ?

   enum days { sun, mon, tue};

3. શું ઉપયોગકર્તા દ્વારા ઈનમ ડેટા દાખલ કરી શકાશે ?
4. (અ) નીચેની કીંમતો સાથેના Months નામના ઈનમ ડેટા ટાઈપને જાહેર કરવાનું વિધાન લખો.

   jan, feb, mar, apr, may, jun, july, aug, sep, oct, nov, dec

   (બ) બે વેરીએબલ M1 અને M2 જાહેર કરો અને તેમને અનુક્રમે jan અને mar કિંમતો એસાઈન કરો.

5. સામાન્ય રીતે ઈનમ ટાઈપના પહેલા ત્રણ મેમ્બર્સ ...., ... અને ..... કીંમતો દ્વારા દર્શાવવામાં આવે છે.
6. Colour નામના ઈનમ ડેટા ટાઈપને જાહેર કરવા માટેનું વિધાન લખો અને તેમાં મેમ્બર તરીકે pink, red, green અને blue દાખલ કરો અને તેમને 33, 45, 58 અને 78 કિંમતો એસાઈન કરો.

12.6 તમે જે શીખ્યા.

આ પાઠમાં તમે ત્રણ વિધાન વિશે જોયું : સ્ટ્રક્ચર, ટાઈપડેફ અને ઈનમ. હવે તમે આ વિધાનનો ઉપયોગ C++ પ્રોગ્રામમાં કરી શકશો.

12.7 પાઠચાંત પ્રશ્નો

1. નીચે આપેલી માહિતી માટે C++ નેસ્ટેડ સ્ટ્રક્ચર લખો. તેને લગતા ડેટા પ્રકારનો ઉપયોગ કરો.

   સ્ટ્રક્ચરના બધા સભ્યોને કિંમત એસાઈન કરો.

નામ	રોલ નંબર	જન્મદિવસ
		દિવસ મહિનો વર્ષ

2. નીચેના પ્રોગ્રામનું આઉટપુટ શું થશે ?

```
# include < iostream.h >
enum day {sun, mon, tues, wed, thu, fri, sat} ;
```

```
void main ()
{
days today;
for (today = sun; today < = sat ; today  ++)
cout <<"\n " today is " << today;
}
```

3. *કેમ ઈનમ બુલિયન (ખોટું, સાચું) એ ઈનમબુલિયન (સાચું, ખોટું) કરતાં સાચું છે ? તેનું કારણ આપો.*

4. *ટાઈપડેફ વિધાનનો લાભ શું છે ?*

5. *ફોન નંબર જે 786-6370 બે ભાગ એટલે કે એકસચેન્જ કોડ અને નંબર તરીકે હોય છે. C++ માં પ્રોગ્રામ લખો કે જે સ્ટ્રકચર ફોનનો ઉપયોગ કરે અને બે ભાગનો સંગ્રહ કરે. જેને સ્ટ્રકચર ફોન કહો. બે સ્ટ્રકચર વેરીએબલના ટાઈપ ફોન બનાવો.એકની શરૂઆત કરી અને યુઝર બીજા એક માટે નંબર ઈનપુટ કરે પછી બંને નંબરને ડિસપ્લે કરો.*

12.8 પાઠગત પ્રશ્નોના જવાબ

પાઠગત પ્રશ્નો 12.1

1. *સ્ટ્રકચર એ સાદા વેરીએબલનો સમૂહ છે કે જે સમાન અથવા જુદા - જુદા પ્રકારના હોઈ શકે.*
2. *એરે સરખાં ડેટા ટાઈપના વેરીયેબલ રાખી શકે છે જયારે સ્ટ્રકચર જુદા - જુદા ડેટા ટાઈપ અથવા સમાન ડેટા ટાઈપના વેરીયેબલ રાખી શકે છે.*
3. *સાચું*
4. structure variable. member
5. *સાચું*
6. *સ્ટ્રકચરની સાથેના સ્ટ્રકચરને નેસ્ટીંગ ઓફ સ્ટ્રકચર કહે છે.*
7. struct time

```
{
int hrs;
int mins;
int secs ;
}
```

8. time time 2;

```
time 2.hrs = 11;
```

9. 18

10.
```
struct add
{
char house no [5] ;
char street [10] ;
char city [10] ;
char state [10] ;
char zip [10] ;
} ;
struct biodata
{
char name [20] ;
char class [5];
int roll;
add address [30] ;
} ;
biodata data ;
cin > > data. name;
cin > > data. class;
cin >> data. roll;
cin >> data. address. houseno ;
cin > > data. address. street;
cin > > data. address. city ;
cin > > data. address. state;
cin >> data. address. zip ;
```

11.
```
cin  > > person. N;
cin >> person. dob.dd;
cin >> person. dob. mm;
cin >> person. dob. yy ;
```

12. *(અ)* 100

(બ) સ્ટ્રક્ચર સમાન છે.

નોંધ

પાઠગત પ્રશ્નો 12.2

1. *પહેલેથી હયાત ડેટા ટાઈપ માટે નવા પ્રકારનો ડેટાટાઈપ જાહેર કરવો.*
2. *પ્રોગ્રામને વાંચી શકાય તેવા બનાવી શકાય.*
3. *અહીં* int data type *ની જગ્યાએ પ્રોગ્રામમાં* integer *ઉપયોગ કરી શકાય છે.*

પાઠગત પ્રશ્નો 12.3

1. *સાચું*
2. sun = 0, mon =1, tue = 2
3. *ના*
4. *(અ)* enum months

   { jan, feb, mar, apr, may, jun, july, aug, sep, oct, nov, dec};

   (બ) months m1, m2;

   m1 = jan ;

   m2 = mar ;
5. 0,1,2
6. enum colour { pink = 33, red = 45, green= 58, blue = 78} ;

13

કલાસ અને ઓબ્જેકટો સાથે કંસ્ટ્રકટર્સ /ડિસ્ટ્રકટર

13.1 પ્રસ્તાવના

ઓબ્જેકટ ઓરીએન્ટેડ પ્રોગ્રામિંગમાં ડેટા પર ફંકશન કરતાં વધારે ભાર મૂકવામાં આવે છે. Class *એક રસ્તો છે, જે ડેટા અને ફંકશનને જોડે છે.કંસ્ટ્રકટર્સ* (constructor) *એ કલાસ બનાવે છે અને ડિસ્ટ્રકટર એ કંસ્ટ્રકટર્સ થી બનેલા ઓબ્જેક્ટને નાબૂદ કરે છે.*

13.2 હેતુઓ

આ પાઠના અધ્યયન બાદ તમે નીચેના મુદ્દાઓથી પરિચિત થશો.

- *કલાસ અને ઓબ્જેકટને વર્ણવી શકશો.*
- *કલાસના સભ્યોને* (Member) *ઉપયોગ કરી શકશો.*
- 3 *ઘટકો વિશે શીખશો.* - (Public) *પબ્લીક*, (Private) *પ્રાઈવેટ અને* (Protected) *પ્રોટેકટેડ*
- *કંસ્ટ્રકટર્સ અને તેના પ્રકાર વિશેની જાણકારી.*
- *ડિસ્ટ્રકટરનો ઉપયોગ.*

નોંધ

13.3 વર્ગ (કલાસ)

કલાસ (Class) *એ ડેટાને અને તેની સાથે સંકળાયેલ* function *ને જોડે છે. તે ડેટા* function *ને ગોપનીય રાખવાની મંજૂરી આપે છે.* Class *ના મુખ્ય બે ભાગ છે.*

(i) Class *જાહેર કરવું.*

(ii) Class *ની વ્યાખ્યા બાંધવી.*

જાહેર કરવાનું માળખું નીચે મુજબ છે.

```
Class class - name
{
private :
Variable      જાહેર કરવા ;
Function      જાહેર કરવા ;
public :
Variable      જાહેર કરવા ;
function      જાહેર કરવા ;
protected :
variable      જાહેર કરવા ;
function      જાહેર કરવા ;
} ;
```

Class *બે કૌંસની વચ્ચે હોય છે અને (;) ગુરુવિરામથી પૂર્ણ થાય છે.* Class *માં* Function *અને* Variable *જાહેર કરવામાં આવે છે. તેને* Class *ના સભ્યો કહેવાય છે જે* Variable Class *ની અંદર જાહેર કરવામાં આવે છે, તેને ડેટા મેમ્બર કહેવાય છે. ફંકશનને મેમ્બર ફંકશન કહેવાય છે.* Keyword Public, Private *અને* Protected *ને વિઝીબીલીટી મોડ કહે છે.*

ડેટા મેમ્બર અને મેમ્બર ફંકશન private *અને* protected *મોડમાં હોય છે, અને જે* public *મોડમાં* member function *થી* access *પ્રવેશ થાય છે.* private *અને* protected *મોડ લગભગ સરખા જ હોય છે. સિવાય કે* protected mode inherited *(વારસાગત) (પાછળના પાઠમાં આવશે) હોય છે. પણ* private mode inherited *નથી તેથી* class *માં આપણે* private *નો ઉપયોગ કરીશું.*

private *કીવર્ડનો ઉપયોગ અવેજીમાં છે. સામાન્ય રીતે* class *ના સભ્યો* private *હોય છે. જો કોઈ લેબલ ના હોય તો સભ્યો* private *જ હોય છે. આવા* class *પૂરેપૂરા બહારની દુનિયા માટે છુપાયેલ હોય છે અને કોઈ* function *ને જાહેર કરતું નથી.*

ઉદાહરણ : 1

```
        # include <iostream.h>
        class student
        {
        private :
                char  name [80];
                int rn ;
                float marks ;
                private :
                        void getdata () ;
                        void putdata () ;
        } ;
        void student : : getdata ()
        {
        cin >> name >> rn >> marks ;
        }
        void student :: putdata ()
        {
        cout <<  name << rn << marks ;
        }
void  main ()
{
        student st ;
        st. getdata ();
        st. putdata () ;
}
```

13.3.1 ઓબ્જેકટ બનાવવા

Object *એ* class instance *છે અને તે* class *ના નામથી બનાવી શકાય છે.*

student st;

Object *બનાવી શકાય છે, જ્યારે* Class *જાહેર કર્યો હોય તેના બંધ કૌંસ પછી તરત જ.*

```
        class student
        {
        } X,Y,Z ;
```

ઉપરનું માળખું student *ના* object x,y,z *બનાવે છે.*

નોંધ

13.3.2 Class ના મેમ્બરનો વપરાશ

Object ના ડેટા મેમ્બર અને મેમ્બર ફંકશન જે Public માં હાજર હોય છે. તેનો વપરાશ થાય છે. તેનું માળખું નીચે મુજબ છે.

Object name. data member ;

object name member function ;

ડેટા ઓપરેટરને Class મેમ્બરના વપરાશનો ઓપરેટર કહે છે.

13.3.3 મેમ્બર ફંકશનની વ્યાખ્યા

મેમ્બર ફંકશન બે રીતે વ્યાખ્યાયિત કરી શકાય છે. (i) Class ની અંદર અને (ii) Class ની બહાર

(i) Class ની અંદર : જ્યારે મેમ્બર ફંકશનને કલાસની અંદર વ્યાખ્યાયિત કરવામાં આવે છે. ત્યારે તેને ઈનલાઈન સમજવામાં આવે છે. જો મેમ્બર ફંકશન ખૂબ નાનો હોય ત્યારે તેને Class ની અંદર વ્યાખ્યાયિત કરવો જોઈએ. આગળના પ્રોગ્રામનો Class વ્યાખ્યાયિત નીચેની રીતે થાય.

```
class student
{
char Name [ 20]
int : rn ;
float marks ;
public :
void getdata ()
{
cin >> name >> rn >> marks ;
}
void putdata ( )
{
        cout << name << rn << marks ;
} ;
```

(2) class ની બહાર

જ્યારે મેમ્બર ફંકશનનો મોટો કોડ હોય ત્યારે તેને કલાસની બહાર વ્યાખ્યાયિત કરવો જોઈએ. આવા ફંકશનનો Prototype કલાસ વ્યાખ્યાયિત કરીએ તેમાં હોવો જ જોઈએ. ઓપરેટર '::' (Scope resolution operator) સ્કોપ રીઝોલ્યુશન ઓપરેટર કહેવાય છે, જે મેમ્બર

ફંકશનને તેના કલાસને સંબંધિત કરે છે.

તેનું માળખું આ પ્રમાણે છે :

```
return_ type   class_ name :: function_ name
```

13.3.4 મેમ્બર ફંકશનનું જાળું (Nesting of member functions)

મેમ્બર ફંકશનને તેના નામથી તે જ કલાસના અન્ય મેમ્બર ફંકશનમાં કોલ કરી શકાય છે. જેને મેમ્બર ફંકશનનું જાળું (નેસ્ટીંગ) કહે છે. નીચેનો પ્રોગ્રામ તેને સમજાવે છે.

```
class greatest
{
int x,y,z ;
public :
        void  getdata () ;
        void display () ;
        int largest () ;
} ;
int greatest :: largest ()
{
int T;
if (x >y)
T = x ;
else
T = y ;
if (T > Z)
        return (T) ;
else
        return (z);
}
void greatest : : getdata ()
{
cout << "enter value of x,y,z" << " \n" ;cin >> x >> y >> z ;
}
void greatest : : display ( )
{
cout << "largest value" << largest () <<" \n" ;
}
```

```
    void main ()
    {
        greatest A ;
        A. getdata () ;
        A. display () ;
    }
```

13.3.5 object માટે મેમરી ફાળવણી

મેમ્બર ફંકશન બને છે અને તેને મેમરીમાં જગ્યા અપાય છે. ફક્ત એક જ વાર કે જયારે તે class *માં વ્યાખ્યાયિત થાય છે. તે* Class *ના બધા ઓબ્જેક્ટ* (Object) *સરખા મેમ્બર ફંકશનનો ઉપયોગ કરે છે. દરેક ઓબજેક્ટના મેમ્બર વેરીએબલ માટે અલગથી જગ્યા આપવામાં આવે છે, કારણ કે મેમ્બર વેરીએબલ જુદા - જુદા ઓબ્જેક્ટની અલગ અલગ કિંમત રાખે છે.*

13.3.6 Object નો Array (એરે)

નીચેનો પ્રોગ્રામ ધ્યાનમાં લો.

```
class emp
{
    char name [30] ;
    int empno ;
    public :
    void getdata () ;
    void putdata () ;
} ;
void main ()
{
    emp e [ 10] ;
    for (i=0; i < 10; i ++)
    e[i]. getdata();
    for (i=0, i < 10 ; i ++ )
    e[i]. putdata () ;
}
```

ઉપરના પ્રોગ્રામમાં દસ ઓબ્જેક્ટ બને છે.

e[0], e[1] .... e[9]

e[i] . getdata ()

ઉપરનું વાકય એરે e *ના* i *નંબરના એલીમેન્ટમાંથી ડેટા લેશે.*

પાઠગત પ્રશ્નો 13. 1

1. *કલાસ શું છે ?*
2. *ઓબ્જેક્ટ શું છે ?*
3. *ત્રણ વિઝિબિલીટી મોડ કયા કયા છે ?*
4. *કલાસના મેમ્બર માટે ડીફોલ્ટ મોડ શું છે ?*

5. class ના પબ્લીક મેમ્બર એ class ના પ્રાઈવેટ મેમ્બરથી અલગ કઈ રીતે પડે છે ?

6. એવો કલાસ બનાવો કે જેમાં class નું નામ student હોય અને તેમાં rn નામનો પ્રાઈવેટ ડેટા મેમ્બર હોય અને putdata () તથા setdata () નામના બે પબ્લીક મેમ્બર ફંકશન હોય.

7. **ખાલી જગ્યા પૂરો.**

(અ) Class ના બધા મેમ્બર સામાન્ય રીતે .... હોય છે.

(બ) Class ની અંદર જે ફંકશન બહાર પડે અને વ્યાખ્યાયિત થાય તે .... છે.

(ક) Class ની બહાર મેમ્બર ફંકશનને માટે .... ઓપરેટર વપરાય છે.

(ડ) ઓપરેટર :: .... તરીકે ઓળખાય છે.

8. **નીચેના વાકયો સાચા છે કે ખોટા તે કહો.**

(અ) Class માત્ર પ્રાઈવેટ ડેટા મેમ્બર જ ધરાવે છે.

(બ) Class મેમ્બર ડોટ ઓપરેટરથી વપરાશ થાય છે.

(ક) Class જાહેર કરીએ તો ગુરુવિરામથી જ પૂરો થવો જોઈએ.

(ડ) સ્કોપ રીઝોલ્યુશન ઓપરેટરનો હંમેશા કલાસ જાહેર થવામાં બહારના મેમ્બર ફંકશન નો ઉપયોગ કરે છે.

13.4 કંસ્ટ્રકટર્સ (Constructors)

constructor એક ખાસ મેમ્બર ફંકશન કરે છે, જે class ના ઓબ્જેકટને શરૂ કરે છે. તે ખાસ છે કારણ કે તેનું નામ અને class નું નામ સમાન જ હોય છે. જયારે ઓબજેકટ બને છે. ત્યારે તે પોતાની રીતે બને છે. તેને constructor કહે છે, કારણ કે તે class ના ડેટા મેમ્બરની કિંમત બાંધે છે. તેને કોઈ return type કે void હોતું નથી.

constructor જાહેર અને વ્યાખ્યાયિત નીચે પ્રમાણે થાય છે.

```
class student
{
        int rn;
        int total;
public :
        student ()
        {
        rn = 0; total = 0 ;
        } ;
        જયારે
```

```
student st;
```

જાહેર કરીએ ત્યારે student કંસ્ટ્રકટર rn = 0 અને total = 0 સેટ કરે છે.

13.4.1 મૂળભૂત (default) કંસ્ટ્રકટર

constructor કે જે પેરામીટરને સ્વીકારે નહી તેને મૂળભૂત constructor કહે છે. જો આવા કોઈ constructor વ્યાખ્યાયિત કરવામાં આવે નહી ત્યારે કમ્પાઈલર તેને મૂળભૂત constructor ને પુરું પાડે છે. આવી વખતે તેને કંઈ ના કરતું constructor કહે છે.

13.4.2 પેરામીટર ધરાવતો કંન્સ્ટ્રકટર (Parameterized constructor)

પેરામીટર ધરાવતા constructor ને પેરામીટરાઈઝડ કંસ્ટ્રકટર કહે છે.

```
class student
{
int rn, total ;
public :
student (int, x, int y)
{
rn = x;  total = y ;
}
} ;
```

જયારે Object બને છે ત્યારે આપણે તેના constructor ફંકશનને આર્ગ્યુમેન્ટ પહોંચાડવી જ જોઈએ. જે બે રીતે થાય છે :

- ફંકશનને સ્પષ્ટ રીતે કોલ કરીને (explicitly)
- ફંકશનને ઈમ્પ્લીસીટલી (implicitly) કોલ કરીને

પહેલું નીચેની રીતે થાય છે.

```
student s1 = student (1,70) ;
```

બીજું નીચેની રીતે થાય છે.

```
student s1 (1,70) ;
```

બીજી રીતે વાંરવાંર વપરાય છે, કારણ કે તે ટૂંકી છે.

13.4.3 કોપી (copy) constructor

કોપી constructor પોતાનો એક આર્ગ્યુમેન્ટ તરીકે તે જ class ના object ના ભલામણ (reference) તરીકે લે છે. નીચેના પ્રોગ્રામ જુઓ.

```
class student
{
```

```
int rn, total ;
public  :
student (int x, int y)
{
rn = x; total = y ;
}
student (student & i )
{
rn = i  rn ;
total = i. total ;
}
} ;
```

ઉપરના પ્રોગ્રામમાં કોપી constructor *અને પેરામીટર ધરાવતાં* constructor *બને છે.*

student s1 (1,75) ; *જે પેરામીટર* (constructor) *ને* call *કરે છે અને* rn *ને* 1 *અને* total *ને* 75 *આપે છે.*

student s2 (s1) ; *જે કોપી* constructor *નો ઉપયોગ કરે છે. જે ઓબજેકટ* S1 *માંથી ઓબ્જેકટ* s2 *ને બનાવે છે. અન્ય રીતે પણ આ શકય છે.*

```
student s2 = s1 ;
```

કોષી કન્સ્ટ્રકટર વાપરીને પ્રારંભ કરવાની ક્રિયાને કોપી ઈનીશયીલાઈઝેનન (initialization) *કહે છે.*

નીચેનું વાકય જુઓ.

```
S2 = S1 ;
```

જે કોપી Constructor *ને* call *કરતો નથી, પરંતુ તે સામાન્ય રીતે* S1 *ની કિંમત* (*મેમ્બર બાય મેમ્બર)* S2 *ને આપે છે.*

13.5 કંસ્ટ્રકટર મુળભૂત આગ્યુંમેન્ટ સાથે (Constructor with default arguments)

constructor or *મુળભૂત* argument *સાથે નીચેની રીતે જાહેર કરાય છે.*

ઉદાહરણ : student (int, rn, int total = 0) ;

અહીં total *ની મૂળભૂત કિંમત* 0 *છે. નીચેનું વાકય જુઓ.*

student s1 (2) ;

જે કિંમત 2, rn ને આપે છે અને 0. total ને આપે છે.પરંતુ સ્ટેટમેન્ટ

student s2 (3,75) ;

જે 3 rn ને આપે અને 75 total ને આપે છે. આ પરિસ્થિતિમાં સાચો પેરામીટર મુળભૂત પેરામીટર કરતાં આગળ પડતો રહે છે. બધાની મુળભૂત કિંમત તેની જમણી બાજુ હોવી જોઈએ.

નીચેનું વાકય જુઓ.

A (int = 0) ;

તેમાં એક જ argument છે જે બે રીતે call થાય છે.

AB ;

AB (5) ;

પહેલાં વાકયમાં કોઈ પેરામીટર પૂરા પડાતા નથી. બીજામાં એક પેરામીટર અપાય છે જયારે કોઈ પેરામીટર પૂરા પાડવામાં આવતા નથી, ત્યારે તે મુળભૂત constructor બને છે. જયારે બંને class માં ઉપયોગમાં લેવાય છે, (મૂળભૂત constructor અને constructor એક મૂળભૂત argument સાથે) ત્યારે તે અનિશ્ચિતતા કરે છે. જેમ કે,

AB ;

તે A () Call કરશે અથવા A (int = 0) ને

13.6. ડિસ્ટ્રકટર (Destructor)

તે constructor થી બનેલા object ને નાબૂદ કરે છે. destructor એક મેમ્બર ફંકશન છે. જેનું નામ એ class નું નામ હોય તે જ હોય છે. પરંતુ તેની ઓળખ ~ નિશાનીથી થાય છે. ઉદાહરણ તરીકે class student નો destructor નીચેની રીતે વ્યાખ્યાયિત થાય છે.

~ student () ;

તે કયારેય કોઈ આર્ગ્યુમેન્ટ લેતું નથી. કે કોઈ કિંમત return કરતું નથી. તે કમ્પાઈલર દ્વારા પ્રોગ્રામનો અંતમાં મેમરી સાફ કરવા માટે જાગૃત કરાય છે. destructor ને પ્રોગ્રામમાં જાહેર કરવો સારી તૈયારી છે કારણ કે ભવિષ્યમાં મેમરીની જગ્યાના ઉપયોગ માટે મેમરી મુક્ત કરે છે.

પાઠગત પ્રશ્નો 13.2

---

1. constructor શું છે ?
2. મૂળભૂત constructor શું છે ?
3. જો L1 અને L2 એક જ class ના Object હોય તો નીચે શું થશે ?

   L1 = L2;

4. Constructor નો અમલ પોતાની રીતે થાય છે જયારે object બનતો હોય આ વાકય સાચું છે કે ખોટું ?

5. destructor શું છે ?

6. **ખાલી જગ્યા પૂરો.**

   (અ) Constructor નું નામ ....ને સમાન હોય છે.

   (બ) જ્યારે Object .... ત્યારે constructor પોતાની રીતે અમલ થાય છે.

   (ક) destructor નો અમલ .... પ્રોગ્રામના અંતમાં થાય છે.

   (ડ) class ને કોઈ પણ સંખ્યામાં constructor થઈ શકે પરંતુ destructor .... એક જ વખત હોય.

   (ઈ) destructor ફંકશનનું નામ class નું જે નામ હોય તે જ પરંતુ તેની આગળ ...... નિશાની હોય છે.

   (ઉ) constructor કે જે બહારની કિંમત લે છે તેને ...... constructor કહે છે.

7. **નીચેના વાકય સાચા છે કે ખોટાં તે કહો.**

   (અ) constructor નું નામ class નું નામ હોય તે જ હોતું નથી.

   (બ) class માં એક કરતાં વધારે constructor સમાન નામ સાથે હોય છે.

   (ક) constructor કોઈ કિંમત return કરતું નથી.

   (ડ) constructor કે જે કોઈ પેરામીટર સ્વીકારતું નથી. તેને મૂળભૂત Constructor કહે છે.

   (ઈ) destructor ની કયારેય કોઈ આર્ગ્યુમેન્ટ હોતી નથી.

   (ઉ) મુખ્ય પ્રોગ્રમના અંતમાં destructor પોતાની રીતે call થાય છે.

   (આ) constructor ના ડેટા મેમ્બર નો પ્રારંભ (initialize) કરે છે.

13.7 તમે જે શીખ્યા

આ પાઠમાં તમે class, object, constructor અને તેના પ્રકાર destructor વિશે શીખ્યા. હવે તમે તમારી રીતે પ્રોગ્રામમાં destructor નો ઉપયોગ કરી શકશો. તમે એ destructor નો ઉપયોગ શા માટે થાય છે, તે પણ શીખ્યા.

13.8 પાઠયાંત પ્રશ્નો

1. class data hiding કેવી રીતે પૂર્ણ કરે છે ?

2. C++ માં class નું બંધારણ કઈ રીતે અલગ પડે છે ?

3. class બનાવવામાં તમે વિઝીબલીટી મોડ શું છે ? તે મોડ કયા કયા છે ?

4. class Teacher ને નીચેના વર્ગીકરણ સાથે વ્યાખ્યાયિત કરો.

Private members :

Name	20 characters
Subject	10 characters
Basic, DA, HRA	float
Salary	float

Calculate - ફંક્શન પગારની ગણતરી કરી પરત કરે. Salary એ Basic, DA અને HRA નો સરવાળો છે.

Public member :

Readdata () જે ડેટાની કિંમત સ્વીકારે અને તે Calculate ( ) ફંક્શનને ગણતરી કરવા પ્રેરે. Displaydata () ફંક્શન ડેટાને સ્ક્રીન પર Print કરે.

5. class worker ને નીચેના વર્ગીકરણ સાથે વ્યાખ્યાયીત કરો.

Private member

wno	integer
wname	25 char
hrwk wgrate	float (hour worked અને wage per hour)
totwage	float (chrwk *wgrate)
calcwg ()	ફંક્શન છે જે hrwk * wgrate શોધે જેની રિર્ટન ટાઈપ float હોય.

Public member

in_data ()	જે wno, wname, hrwk, wgrate કિંમત સ્વીકારે અને cal cwg () ને ગણતરી કરવા પ્રેરે.
out data ()	જે બધા ડેટા મેમ્બરને સ્કીનખ પર બતાવે.

6. constructor ની ખાસ લાક્ષણિકતા કઈ છે ?

7. પેરામીટર સાથે constructor શું છે ?

8. કોપી constructor શું છે?

9. constructor નું મહત્વ શું છે ?

13.9 પાઠગત પ્રશ્નોના જવાબ

પાઠગત પ્રશ્નો - 13.1

1. Class ડેટા અને ફંકશનને સાંકળે છે.
2. class ના variable ને object કહેવાય.
3. public, private, protected
4. private
5. classના public મેમ્બર ઓબજેક્ટથી access થાય છે જયારે private મેમ્બરના public ના મેમ્બર ફંકશનથી access થાય છે.
6. class student

```
{
int rn;
public :
    void getdata () ;
    void putdata () ;
} ;
```

7. (a) private (b) inline (c) : : (d) સ્કોપ રીઝોલ્યુશન
8. (અ) ખોટું (બ) સાચું (ક) સાચું (ડ) સાચું.

પાઠગત પ્રશ્નો 13.2

1. એક ખાસ ફંકશન કે જેનું નામ અને Class નું નામ સમાન હોય તેને constructor કહે છે. તેને કોઈ return type હોતું નથી. તે તેની જાતે જાગૃત થાય છે જયારે પણ object બને.
2. કોઈ પેરામીટર વગરના constructor ને મૂળભૂત construcor કહે છે.

3. તે ઓબ્જેકટના બધા મેમ્બર L2 થી L1 ને એક પછી એક આપે છે.

4. સાચું

5. તે એક ફંકશન છે. જેનું નામ class ના નામની સમાન હોય છે. પરંતુ તેની આગળ ~ ની નિશાની હોય છે. તે ફંકશન પુરું થાય ત્યારે પોતાની રીતે જાગૃત થાય છે. તે મેમરીને મુક્ત કરે છે, જે બીજા ડેટા અને સૂચનો સંગ્રહ કરવા પણ વપરાય છે.

6. (અ) class નામ

   (બ) બને છે.

   (ક) પોતાની જાતે

   (ડ) એક

   (ઈ) ~

   (ઉ) પેરામીટર સાથે.

7. (અ) ખોટું

   (બ) સાચું

   (ક) સાચું

   (ડ) સાચું

   (ઈ) સાચું

   (ઉ) સાચું

   (ઊ) સાચું.

14

વારસો - કલાસનું વિસ્તરણ

14.1 પ્રસ્તાવના

આ પાઠ વારસા (ઈનરીટન્સ) વિશે ચર્ચા કરે છે. એક બાળક જેવી રીતે તેના માતા પિતા પાસેથી વારસામાં ગુણધર્મો મેળવે છે. તેવી જ રીતે એક કલાસ પણ બીજા કલાસના ગુણધર્મો વારસામાં મેળવી શકે છે, જેને ઈનહેરીટન્સ કહે છે. ઈનહેરીટન્સ (inheritance) *નો સૌથી મહત્વનો ફાયદો કોડની પુનઃ ઉપયોગિતાનો છે. એક વખત બેઝ કલાસ લંખાઈ અને ડીબગ થઈ જાય પછી તેને અલગ - અલગ પરિસ્થિતિઓમાં ફરીથી લખ્યા વગર ઉપયોગ કરીશકાય છે. એક વખત લખાઈ ગયેલા કોડને વાંરવાંર ઉપયોગ કરવાથી સમય, પૈસા અને કોડ ફરીથી લખવાના પ્રયાસથી બચાવે છે. જૂના કલાસને ફરીથી વ્યાખ્યાયિત કર્યા વગર, તમે તમારા ઈચ્છિત કલાસમાં નવા ગુણધર્મો ઉમેરી શકો, અને ઈનહેરીટેડ કલાસના મેમ્બર ફંકશનને ફરી વ્યાખ્યાયિત કરી શકો છો.*

14.2 ઉદ્દેશો

આ પાઠ ભણ્યા પછી તમે નીચેનાઓ માટે સક્ષમ હશો.

- *ઈન્હેરીટેન્સ (વારસા) ના ખ્યાલને સમજાવવા.*
- *ઈન્હેરીટન્સના પાંચ સ્વરૂપો વર્ણવવા.*
- *ત્રણ પ્રકારના ઈનહેરીટેન્સને વ્યાખ્યાયિત કરવા.*
- *ત્રણેય દશ્યતા સ્થિતિઓ સમજાવવા.*
- *એબ્સ્ટ્રેકટસ અને વર્ચ્યુઅલ કલાસ વર્ણવવા.*

14.3 ઈનહેરીટન્સની જરૂર

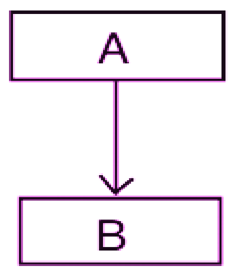
ઈનહેરીટન્સ એ ઓબ્જેક્ટ લક્ષી (ઓરીએન્ટેડ) ભાષાનો મહત્વનો ખ્યાલ છે. આ ખ્યાલ ઓબ્જેક્ટ ઓરીએન્ટેડ ભાષામાં રજૂ કરવા પાછળ ઘણા કારણો છે. કેટલાક મુખ્ય કારણો છે. :

1. ઈનહેરીટન્સની સંબંધો વ્યકત કરવાની ક્ષમતા જે સ્તવિ દુનિયાના મોડેલને ઘણા નજીકથી મળતા આવે છે.
2. પુનઃ ઉપયોગિતાનો વિચાર, દા.ત. નવો કલાસ જૂના કલાસના અમુક લક્ષણો વાપરી શકે.
3. ઈનહેરીટન્સની બદલાતા રહેવાની પ્રકૃતિ, તે આગળ વધારી શકે છે.

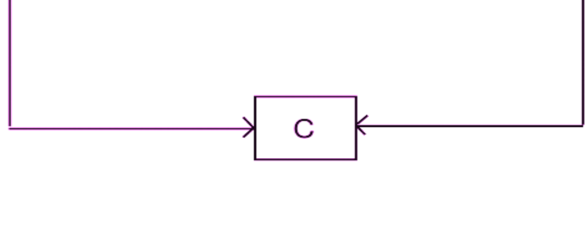
14.4 ઈનહેરીટન્સના વિવિધ સ્વરૂપો (Different Forms of Inheritance)

જૂના કલાસમાંથી નવો કલાસ બનાવવાની પધ્ધતિને ઈનહેરીટન્સ (વારસો) કહેવાય છે. જૂના કલાસને બેઝ (આધાર) કલાસ અને નવા કલાસને ડીરાઈવ્ડ (ઉતરી આવેલ) કલાસ કહેવાય છે. ઈનહેરીટન્સના વિવિધ સ્વરૂપો છે.

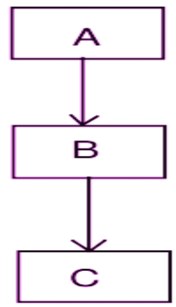
1. **સિંગલ (એક) ઈનહેરીટન્સ** (Single Inheritance) - ડીરાઈલ્ડ કલાસ જે ફકત એક જ બેઝ કલાસમાંથી બનેલ છે તેને સિંગલ ઈનહેરીટન્સ કહે છે.



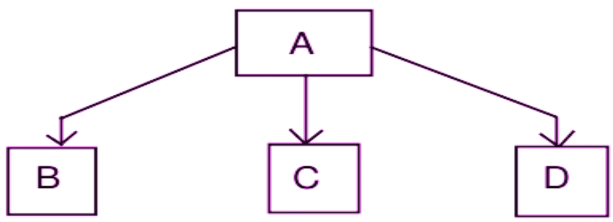
2. **મલ્ટીપલ (બહુ) ઈનહેરીટન્સ** (Multiple Inheritance) - એક કરતાં વધુ બેઝ કલાસમાંથી બનાવેલ કલાસને મલ્ટીપલ ઈન્હેરીટન્સ કહે છે.



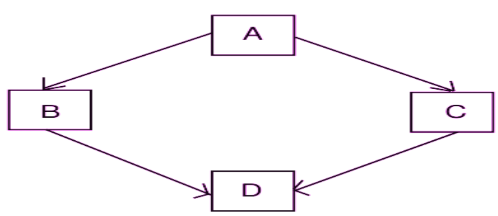
3. **મલ્ટીલેવલ ઈનહેરીટન્સ (Multilevel Inheritance)** - *એક ડીરાઈવ્ડ કલાસમાંથી બીજો હીરાઈડ કલાસ બનાવવાની પધ્ધતિને મલ્ટિલેવલ ઈનહેરીટન્સ કહે છે.*



4. **અધિક્રમિક ઈનહેરીટન્સ (Hierarchical Inheritance)** - *એક કલાસને એક કરતા વધું કલાસ ઈનહેરીટ કરી શકે. આ પ્રક્રિયાને અધિક્રમિક ઈનહેરીટન્સ કહે છે.*



5. **મિશ્ર ઈનહેરીટન્સ (Hybrid Inheritance)** - *તે અધિક્રમિક અને મલ્ટીપલ ઈનહેરીટન્સનું સંયોજન છે.*



14.5 તારવેલ કલાસની વ્યાખ્યા (Defining Derived Class)

તારવેલ કલાસને વ્યાખ્યાયિત કરવા માટે બેઝ કલાસની દશ્યતા સ્થિતિને (Visibility mode) *ધ્યાનમાં રાખીને તેની સાથે સંબંધ સ્પષ્ટ કરવામાં આવે છે.*

તારવેલ કલાસને વ્યાખ્યાયિત કરવા માટેનું સામાન્ય સ્વરૂપ નીચે મુજબ છે.

```
class derived _ class: visibility_ mode base_ class
{
________
_________ // તારવેલ કલાસના સભ્યો.
};
```

નોંધ

કોલોન (;) બતાવે છે કે derived_ class *એ બેઝ કલાસમાંથી (કેટલાક ગુણધર્મો સાથે) તારવેલ છે.*

બેઝ કલાસના નામની પાછળ (:) આવે છે. બધા બેઝ કલાસના નામો, તારવેલ કલાસની પાછળ કોલન (:) મુકી અને અલ્પવિરામ દ્વારા અલગ પાડી લખવામાં આવે છે. દશ્યતા સ્થિતિ ખાનગી (Private) *અથવા જાહેર* (Public) *અથવા સુરક્ષિત* (Protected) *હોઈ શકે છે. જો કોઈ દશ્યતા સ્થિતિ આપેલ ના હોય તો, મૂળભૂત રીતે ખાનગી દશ્યતા સ્થિતિ ગણવામાં આવે છે. તારવેલ કલાસના કેટલાક ઉદાહરણો નીચે આપેલ છે :*

```
Class Marksheet : .public student // public derivation
{
        // members of derived class
} ;
class Marksheet : private student// private deriovation
        // member of derived class
} ;
class Marksee : protected student // protected derivation
{
        // members of protected class
} ;
```

ઉપરની વ્યાખ્યાઓમાં માર્કશીટ એ સ્ટુડન્ટ બેક કલાસમાંથી તારવેલ છે. દશ્યતા સ્થિતિ જાહેર એ બતાવે છે કે સ્ટુડન્ટ કલાસ એ જાહેર બેઝ કલાસ છે. તેવી જ રીતે, દશ્ય સ્થિતિ ખાનગી અથવા સુરક્ષિત એ સૂચવે છે કે સ્ટુડન્ટ બેઝ કલાસએ અનુક્રમે ખાનગી અથવા સુરક્ષિત બેઝ કલાસ છે.

આપણે જયારે કહીએ કે કલાસના સભ્યો ઈનહેરીટેબલ છે, તેનો અર્થ એ થાય કે તે તારવેલ કલાસ સીધો જ તેમનો ઉપયોગ કરી શકે છે. જો કે તારવેલ કલાસ ફકત બિન ખાનગી (non private) *સભ્યોને જ વાપરવાનો વિશેષાધિકાર ધરાવે છે. બેઝ કલાસના ખાનગી સભ્યોને વાપરી શકાતા નથી. પણ તારવેલ કલાસના ઓબ્જેકટ બિન ખાનગી તારવેલ સભ્યો દ્વારા તેમને વાપરી શકે છે.*

14.6 મલ્ટિપલ ઈનહેરીટન્સ (Multiple inheritance)

આપણે જાણીએ છીએ કે મલ્ટીપલ બેઝ કલાસમાંથી ઈનહેરીટઝ કરેલા સબકલાસને મલ્ટીપલ ઈનહેરીટન્સ કહે છે. તારવેલા કલાસને વ્યાખ્યાયિત કરવાનું સિન્ટેકસ નીચે આપેલ છે :

```
class derived _ classname : mode baseclass1, mode baseclass2
{
    // members of derived class
};
```

ઉદાહરણ **: 1**

```
class marks : public semester1, private semester2
{
        // members
    };
```

14.7 દશ્યતા સ્થિતિઓ (Visibility Modes)

તે જાહેર, ખાનગી અને સુરક્ષિત હોઈ શકે છે. બેઝ કલાસના ખાનગી ડેટાને ઈનહેરીટ કરી શકાતા નથી.

1. *જો ઈનહેરીટન્સ જાહેર સ્થિતિમાં કરવામાં આવે તો, બેઝ કલાસના જાહેર સભ્યો તારવેલ કલાસમાં જાહેર સભ્યો બને છે અને સુરક્ષિત બેઝ કલાસના સુરક્ષિત સભ્યો તારવેલ કલાસમાં સુરક્ષિત સભ્યો બને છે.*
2. *જો ઈનહેરીટન્સ ખાનગી સ્થિતિમાં કરવામાં આવે તો, બેઝ કલાસના જાહેર અને સુરક્ષિત સભ્યો તારવેલ કલાસમાં ખાનગી સભ્યો બને છે.*
3. *જો ઈનહેરીટન્સ સુરક્ષિત સ્થિતિમાં કરવામાં આવે તો, બેઝ કલાસના જાહેર અને સુરક્ષિત સભ્યો તારવેલ કલાસમાં સુરક્ષિત સભ્યો બને છે. નીચેનું કોષ્ટક ઈનહેરીટન્સનાં ત્રણ પ્રકાર બનાવે છે.*

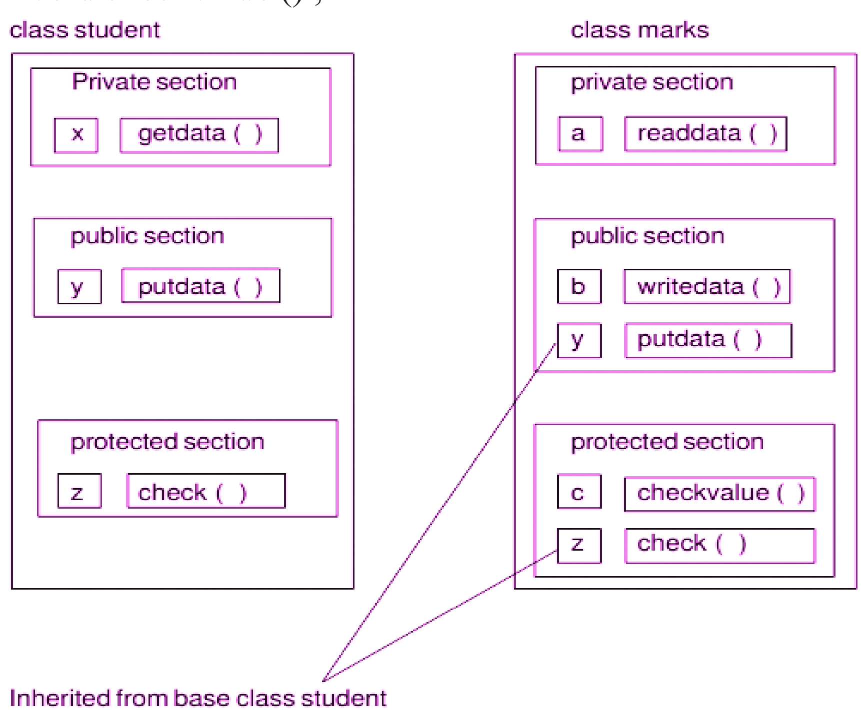
બેઝ કલાસ	**ડીરાઈવ્ડ (તારવેલ) કલાસ**		
એક્સેસ દર્શક (સ્પેસીફાયર)	*જાહેર*	*ખાનગી*	*સુરક્ષિત*
જાહેર	*જાહેર*	*ખાનગી*	*સુરક્ષિત*
ખાનગી	*ઈનહેરીટ કરતું નથી.*	*ઈનહેરીટ કરતું નથી.*	*ઈનહેરીટ કરતું નથી.*
સુરક્ષિત	*સુરક્ષિત*	*સુરક્ષિત*	*સુરક્ષિત*

જાહેર દશ્યતા સ્થિતિ

નીચનું ઉદાહરણ અને આકૃતિ જાહેર રીતે તારવેલ કલાસ સમજાવે છે.

```
class student
{
private;
        int x ;
        void  get data () ;
public ;
        int y ;
        void putdata () ;
```

```
protected :
        int z ;
        void check () ;
} ;
class marks : public student
{
private :
        int a ;
        void readdata () ;
public :
        int b ;
        void writedata () ;
protected :
        int c ;
void checkvalue () ;
```



આકૃતિ. 14.1 **ક્લાસની જાહેર તારવણી (public derivation of a class)**

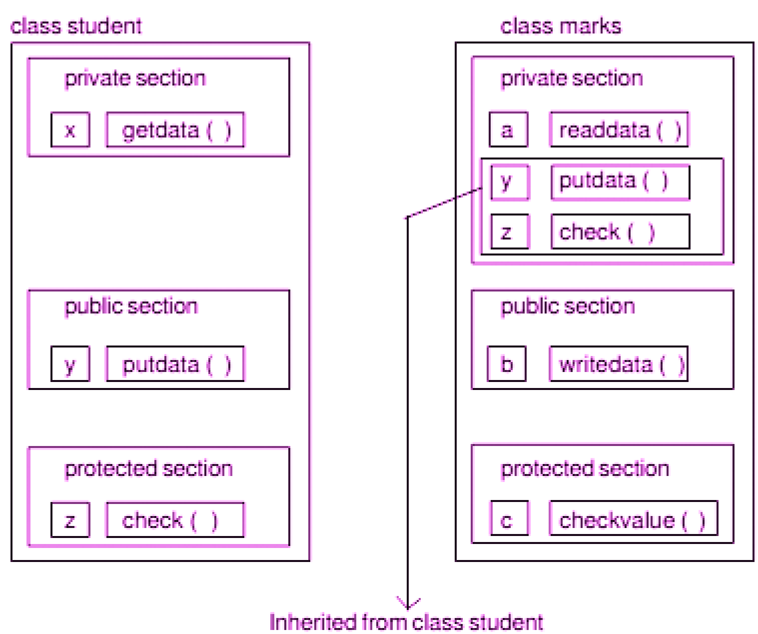
જાહેર તારવણી એ તારવેલ કલાસના ઈનહેરીટેડ સભ્યોના એકસેસ સ્પેશીફાયર બદલતું નથી. સ્ટુડન્ટ બેઝ કલાસના ખાનગી ડેટાને ઈનહેરીટ કરી શકાતા નથી.

જાહેર દશ્યતા સ્થિતિ

આપણે તે જ ઉદાહરણ આપીશું, પરંતુ તારવણી ખાનગી સ્થિતિ દ્વારા

```
class student
{
// same as in previous example
} ;
class marks : private student
{
//
};
```

નીચેની આકૃતિ કલાસની ખાનગી તારવણી દર્શાવે છે.



આકૃતિ. 14.2 કલાસની ખાનગી તારવણી (private derivation of a class)

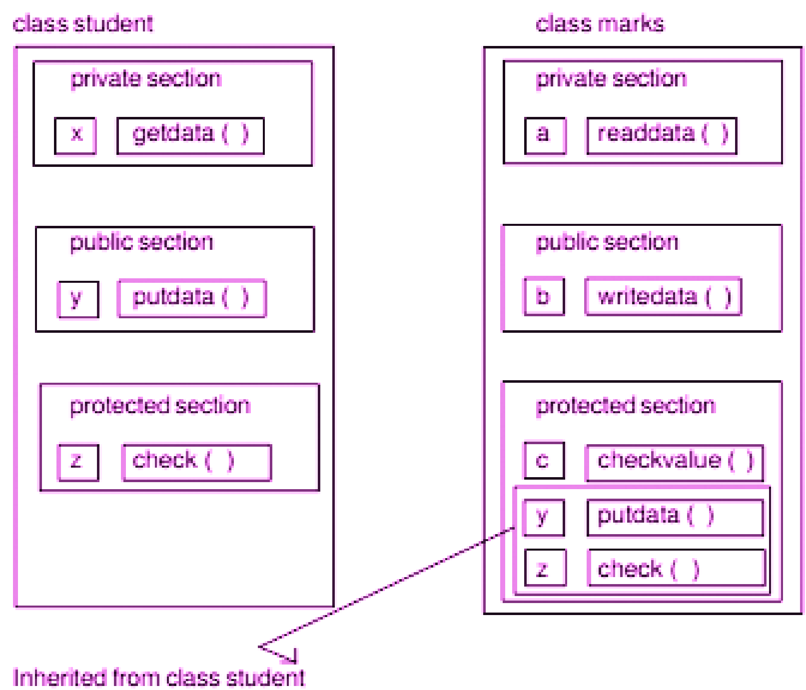
આકૃતિ પરથી સ્પષ્ટ થાય છે કે બેઝ કલાસના જાહેર અને સુરક્ષિત વિભાગમાં રહેલો ડેટા એ તારવેલ કલાસમાં ખાનગી સભ્યો બને છે. બેઝ કલાસના ખાનગી વિભાગમાં રહેલા ડેટા ઈનહેરીટ (તારવી) કરી શકાતો નથી.

સુરક્ષિત દશ્યતા સ્થિતિ (Protected visibility mode)

આપણે એ જ ઉદાહરણનો ઉપયોગ કરીશું પરંતુ તારવણી સુરક્ષિત સ્થિતિમાં કરેલ છે.

```
class student
{
// same as in previous example
} ;
class marks : protected student
{
} ;
```

નીચેની આકૃતિ કલાસની સુરક્ષિત તારવણી દર્શાવે છે.



આકૃતિ. 14.3 કલાસની સુરક્ષિત તારવણી (Protected Derivation of a class)

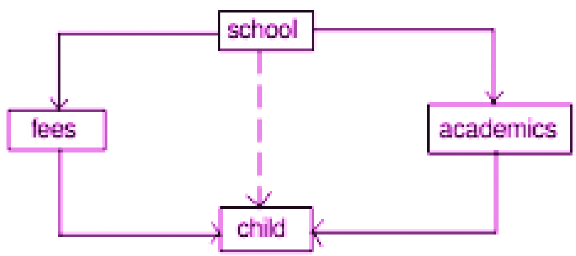
બેઝ કલાસના ખાનગી વિભાગોમાં રહેલા ડેટાને ઈનહેરીટ કરી શકાતા નથી. ખાનગી અને સુરક્ષિત વિભાગ વચ્ચે તફાવત એ છે કે સુરક્ષિત વિભાગમાં રહેલા ડેટાને (તારવી) શકાય છે. નહિ તો વિભાગના કલાસનો ઓબ્જેકટ બંને વિભાગોને એકસેસ (ઉપયોગ) કરી શકાતો નથી.

14.8 એબ્સ્ટ્રેકટ કલાસ (Abstract class)

એબ્સ્ટ્રેકટ કલાસ એ એવો કલાસ છે જે ઓબ્જેકટ બનાવવા માટે ઉપયોગમાં આવતો નથી. એબ્સ્ટ્રેકટ કલાસનું આલેખન ફકત બેઝ કલાસ તરીકે કાર્ય કરવા થયેલ છે. (બીજા કલાસની તારવણી માટે). આ એક પ્રોગ્રામ બનાવતી વખતે વપરાતો વિચાર છે, જેનો ઉપયોગ બીજા કલાસ બનાવતી વખતે આધાર (બેઝ) તરીકે કરી શકાય. આગળના ઉદાહરણમાં સ્ટુડન્ટ કલાસ એ એબ્સ્ટ્રેકટ કલાસ છે કારણ કે તેનો ઉપયોગ કોઈ ઓબ્જેકટ બનાવવા માટે થયેલ નથી.

14.9 આભાસી બેઝ કલાસ (Virtual Base Class)

એવી પરિસ્થિતિની કલ્પના કરો જેમાં બધાં ત્રણે પ્રકારના ઈનહેરીટન્સ, મલ્ટીલેવલ, મલ્ટીપલ અને અધિક્રમિકના દૃષ્ટાંતો સંકળાયેલા હોય. નીચે આપેલી આકૃતિમાં જણાવેલ ઉદાહરણ ધ્યાનમાં લો.



આકૃતિ. 14.4

child ના સીધી રીતે બે બેઝ કલાસ છે. fees અને academics, જેનો સહિયારો બેઝ કલાસ school છે. child એ school ના લક્ષણો વારસામાં બે અલગ અલગ રીતે મેળવે છે. તે સીધી રીતે પણ વારસામાં લક્ષણો મેળવી શકે છે, જે તૂટેલી લીટી દ્વારા બતાવ્યું છે. school ને ઘણી વખત પરોક્ષ (ઈનડાયરેકટ) બેઝ કલાસ તરીકે ઓળખાય છે. school ના બધા જ જાહેર અને સુરક્ષિત સભ્યો child ના વારસામાં બે વખત આવે છે, એક fees દ્વારા અને ફરીથી academics દ્વારા. આનો મતલબ થયો કે school તરફથી child ને વારસામાં બધા સભ્યોની બેવડી નકલ મળશે. આનાથી અનિશ્ચિતતા ઉમેરાશે, જેને ટાળવી જોઈએ.

ઘણા રસ્તાઓને લીધે વારસામાં મળેલ બેવડા સભ્યો જો સામાન્ય બેઝ કલાસના બદલે આભાસી બેઝ કલાસ વાપરવામાં આવે તો તેનાથી બચી શકાય છે, જે નીચે બતાવ્યું છે. :

```
class school
{ --------
----------
} ;
class fees : virtual public school
{ ------
-------
} ;
```

```
class academics : public virual school
{ ------
-------
} ;
class child ; public fees, public academics
{
    // school  ની ફક્ત એક કોપી વારસામાં મળશે.
} ;
```

નોંધ : - *તમે* Virtual *અથવા* Public virtual *બેમાંથી કોઈ એક રીતે લખી શકો છો.*

પાઠગત પ્રશ્નો 14.1

1. *ઈનહેરીટન્સ શું છે ? બેઝ કલાસ અને ડીરાઈવ્ડ (તારવેલ) કલાસ શું છે ?*
2. *ઈનહેરીટન્સના વિવિધ રૂપો કયાં છે ?*
3. *ઈનહેરીટન્સની ત્રણ સ્થિતિઓ કઈ છે ?*
4. *બેઝ કલાસના ખાનગી વિભાગને ઈનહેરીટ કરી શકાય, સાચું કે ખોટું ?*
5. *ખાનગી અને સુરક્ષિત વિભાગો વચ્ચે શું તફાવત છે ?*
6. *એબ્સ્ટ્રેકટ કલાસ શું છે ?*
7. *ખાલી જગ્યા પૂરો :*
   (અ) *બેઝ કલાસને .... તરીકે ઓળખાય છે.*
   (બ) *ડીરાઈવ્ડ કલાસને બેઝ કલાસમાંથી .... અથવા .... રીતે તારવી શકાય.*
   (ક) *મૂળભૂત રીતે ડીરાઈવ્ડ કલાસમાં બેઝ કલાસ .... સ્થિતિમાં દશ્યીત હોય છે.*
   (ઈ) *જયારે ડીરાઈવ્ડ કલાસ એ એક કરતાં વધુ બેઝ કલાસમાંથી ઈનહેરીટ કરવામાં આવે ત્યારે .... ઈનહેરીટન્સ કહેવાય છે.*
8. *નીચેનાઓ સાચાં છે કે ખોટા તે જણાવો.*
   (અ) *ઈનહરેટીન્સનો મતલબ થાય છે કે બાળક તેના વાલી પાસેથી કેટલાક લક્ષણો મેળવે છે.*
   (બ) *મૂળભૂત રીતે બેઝ કલાસએ જાહેર સ્થિતિમાં ડીરાઈવ્ડ કલાસમાં દશ્યીત હોય છે.*
   (ક) *જયારે ડીરાઈવ્ડ કલાસ એક કરતાં વધુ બેઝ કલાસમાંથી તારવવામાં આવે છે, તેને અધિક્રમિક ઈનહેરીટન્સ કહે છે.*

(ડ) બેઝ કલાસને એબ્સ્ટ્રેકટ કલાસ કહે છે.

(ઈ) બેઝ કલાસના ખાનગી ડેટાને ઈનહેરીટ કરી શકાય.

9. **નીચેના કલાસને ધ્યાનમાં લો અને તેની નીચે આપેલા પ્રશ્નોના જવાબ આપો :**

```
class zoo
{
char location [20];
protected :
    int no_ of_ animals ;
public :
    void inputdata (char,int) ;
    void outputdata ( ) ;
} ;
class animal  : protected zoo
{
int tail ;
protected :
    int legs ;
public :
    void readdata(int, int);
    void writedata ( ) ;
};
class carnirvorous : private animal
{
int paw_ size ;
public :
    void fetchdata (int);
    void  displayed ( ) ;
};
```

(અ) animal કલાસના બેઝ કલાસ અને ડીરાઈવ્ડ કલાસના નામ આપો.

(બ) displayed ફંકશન દ્વારા ઉપયોગમાં લઈ શકાય તેવા ડેટા સભ્યોના નામ આપો.

(ક) Carnivorous કલાસના ઓબ્જેકટ દ્વારા વપરી શકાય તેવા ડેટા સભ્યોના નામ આપો.

(ડ) animal કલાસના ઓબ્જેકટ દ્વારા Outputdata સભ્ય ફંકશન વાપરી શકાય ?

14.10 તમે શું શીખ્યા (what you have Learn)

આ પાઠમાં ઓબ્જેક્ટ ઓરીએન્ટેડ પ્રોગ્રામીંગના ખૂબ જ અગત્યનો મુદ્દો, ઈનહેરીટન્સ (વારસો) વિશે શીખ્યા. આપણે ઈનહેરીટન્સના અલગ અલગ સ્વરૂપ અને પ્રકારો વિશે ચર્ચા કરી જેનો ઉપયોગ તમે C ++ પ્રોગ્રામીંગમાં કરી શકશો.

14.11 પાઠયાંત પ્રશ્નો

1. *ઈનહેરીટન્સની જરૂરિયાત શી છે ?*
2. *ઈનહેરીટન્સની ત્રણ સ્થિતિઓ કઈ છે ? સમજાવો.*
3. *આભાસી બેઝ કલાસ શું છે ? ઉદાહરણ સાથે તેને સમજાવો.*
4. *નીચેના કલાસને ધ્યાનમાં લઈ તેની નીચે જણાવેલ પ્રશ્નોના જવાબ આપો :*

```
class vehicle
{
int wheels ;
protected :
        int passenger;
public:
      void inputdata (int, int) ;
      void output data ( ) ;
} ;
class heavy _ vehicle : protected vehicle
{
      int diesel _ petrol ;
      protected :
        int load;
public :
      void readdata (int,int) ;
      void writedata ( )  ;
} ;
class bus : private heavy _ vehicle
{
      char make [20] ;
public :
      void fetchdata (int) ;
      void displaydata ( ) ;
} ;
```

(i) heavy_vehicle કલાસના બેઝ કલાસ અને ડીરાઈવ્ડ કલાસના નામ આપો.

(ii) displaydata ફંકશન દ્વારા ઉપયોગમાં લઈ શકાય તેવા ડેટા સભ્યોના નામ આપો.

(iii) bus કલાસના ઓબ્જેકટ દ્વારા વાપરી શકાય તેવા ડેટા સભ્યોના નામ આપો.

(iv) heavy કલાસના ઓબ્જેકટ દ્વારા સભ્ય ફંકશન outputdata વાપરી શકાય ?

14.12 પાઠગત પ્રશ્નોના જવાબ

પાઠગત પ્રશ્નો 14.1

1. જૂના કલાસમાંથી નવા કલાસને તારવવાની પધ્ધતિને ઈનહેરીટન્સ કહે છે.

   જૂના કલાસને બેઝ કલાસ અને નવા કલાસને ડીરાઈવ્ડ કલાસ તરીકે ઓળખાય છે.

2. 1.સિંગલ ઈનહેરીટન્સ

   2.મલ્ટીપલ ઈનહેરીટન્સ

   3.મલ્ટીલેવલ ઈનહેરીટન્સ

   4.મિશ્ર ઈનહેરીટન્સ

   5. અધિક્રમિક ઈનહેરીટન્સ

3. ખાનગી, જાહેર અને સુરક્ષિત

4. ખોટું

5. protected સેકશનનો ડેટા ઈનહેરીટ કરી શકાય.

6. બેઝ કલાસને એબ્સ્ટ્રેકટ કલાસ કહે છે.

7. (અ) એબ્સ્ટ્રેકટ કલાસ

   (બ) જાહેર, ખાનગી, સુરક્ષિત

   (ક) ખાનગી

   (ડ) મલ્ટિપલ

8. (અ) T (બ) F (ક) F (ડ) T (ઈ) F

9. (અ) બેઝ કલાસ - zoo

   ડીરાઈવ્ય કલાસ - carnivorous

   (બ) legs, no - o- animals, paw - size.

   (ક) કોઈ નહી.

   (ડ) ના

15

નિર્દેશક

15.1 પ્રસ્તાવના

C++માં, *નિર્દેશક* (Pointer) *સાથે પ્રોગ્રામિંગ વધુ શક્તિશાળી બને છે અને તેનો વ્યાપક ઉપયોગ થાય છે. તે પ્રક્રિયા કરવા માટેનો સમય બચાવે છે. પોઈન્ટર એ એક ચલ* (Variable) *છે જેમાં બીજા ચલનું સરનામું* (address) *હોય છે. તેથી, પ્રોગ્રામિંગને સરનામા* (address)*સાથે નિસ્બત છે, નહિ કે તેમાં સંગ્રહ થયેલ કિંમત સાથે.*

15.2 હેતુઓ

આ પાઠનો અભ્યાસ કર્યા પછી તમે નીચેના મુદ્દા સમજી શકશો :

- ***એરે*** (arrays)***ની અંદર પોઈન્ટરનો ઉપયોગ***
- ***માળખા*** (structure) ***ની અંદર પોઈન્ટર ચલ*** (variable) ***વ્યાખ્યાયીત કરી શકશો અને પોઈન્ટર મારફતે ડેટા સભ્યો*** (data member)***ને વાપરી શકશો.***
- ***પોઈન્ટર દ્વારા વર્ગ*** (class) ***ની અંદર વસ્તુઓ*** (objects) ***વ્યાખ્યાયીત કરી શકશો અને સભ્યો*** (members) ***ને વાપરી શકશો.***
- ***ફાઈલની ખોલી અથવા બંધ કરી શકશો.***

15.3 નિર્દેશક (Pointer)

નિર્દેશક (પોઈન્ટર) એ એક ચલ (variable) ***છે જે વસ્તુઓ જેવી કે ચલ*** (variable) ***અથવા એરે તત્વ*** (array element) ***ની માહિતીનું સ્થાન (નહિં કે તેમની કિંમત) રજુ કરે છે. C++માં પોઈન્ટરને વારંવાર ઉપયોગ કરવામાં આવે છે, કારણ કે તે ઘણી રીતે ઉપયોગી છે. નીચેના ઉદાહરણો ધ્યાનમા રાખો :***

```
# include <iostream.h>
void main ()
```

```
{
int A = 5 ;
cout << &A ;
int *ptr ;
ptr = &A ;
cout << *ptr ;
}
```

ઉપરના ઉદાહરણમાં ચલ A ***ની કિંમત*** 5 ***છે, જ્યારે*** & A ***ની અંદર ચલ*** A ***ના*** memory cell ***નું સરનામું હશે. જે ચલ*** (variable) ***સરનામું ધરાવે છે તેને પોઈન્ટર ચલ કહે છે.*** int *ptr ***નો અર્થ એવો થાય છે કે ચલ*** ptr ***એ એવા મેમરી સેલ*** (memory cell) ***ને નિર્દેશ કરે છે જે*** int ***પ્રકારના ડેટાનો સંગ્રહ કરે છે.*** *ptr ***મેમરી સેલમાં સંગ્રહાયેલ કિંમત માટે વપરાય છે, અને તે મેમરી સેલ*** ptr ***પોઈન્ટર ધ્વારા નિર્દેશાય છે. આને પોઈન્ટરનું ડી-રેફરેન્સીંગ*** (de-referencing)***કહેવાય છે. ઉપરોક્ત પ્રોગ્રામનો આઉટપુટ મેમરી સેલ*** A ***નું સરનામું અને કિંમત*** 5 ***દર્શાવશે.***

```
void *ptr ;
```

અહીં ptr ***કોઈપણ પ્રકારની માહિતી*** (data types) ***માટે નિર્દેશ કરી શકે છે.***

15.3.1 એરે માટે નિર્દેશક (Pointer to Array)

નીચેની ઘોષણા ધ્યાનમાં રાખો :

```
int A [5] ;
```

એરે (Array) A *નું નામ એ પોતે જ એક પોઈન્ટર છે જે એરેના શૂન્ય નંબરના* element (&A[O]) *નું સરનામું ધરાવે છે. પ્રોગ્રામની અંદર તે અચળ છે, તેની કિંમત બદલી શકાતી નથી. નીચેનો પ્રોગ્રામ એરે* A *ની બધી જ કિંમતો દર્શાવશે.*

```
#include <iostream.h>
void main ()
{
   int A [5] = {20,35,25,22,27} ;
   for (int i = O; i<5 ; i ++)
   cout << " \n" << A [i] ;
   }
```

આ પ્રોગ્રામનો આઉટપુટ છે

```
20
35
25
22
27
```

ઉપરોક્ત પ્રોગ્રામ પોઈન્ટરનો ઉપયોગ કરીને પણ લખી શકાય છે.

```
#include <iostream.h>
void main ()
```

નોંધ

```
{
    int A [5] = {20,35,25,22,27} ;
    for (int i = O; i<5 ; i ++)
    cout << " \n" << *(A +i) ;
    }
```

એક તબક્કે i *ની કિંમત* 2 *છે, તેથી* (A + i) *બે થશે, એટલે કે શૂન્યથી બે સ્થળ આગળ. આ* *(A +i) *તે સ્થાન પર સંગ્રહ કરેલો ડેટા છાપશે.*

15.3.2 શબ્દમાળાના અચળનો નિર્દેશક (Pointer to String constant)

નીચેનો પ્રોગ્રામ જુઓ :

```
# include <iostream.h>
void main ()
{
char stu1[ ] = " work as an array" ;
char *stu2 = "work as a pointer" ;
cout << stu1 ;
cout << stu2 ; //display both the strings
stu1 ++ ; // it is a wrong statement
stu2 ++ ;
cout << stu2 ; // it prints "ork as a pointer"
}
```

stu1++ *એ વિધાન ખોટું છે કારણકે* stu1 *એક પોઈન્ટર છે જે શૂન્ય સ્થળ* (&stu1[o]) *નું સરનામું ધરાવે છે. તે પ્રોગ્રામની અંદર અચળ છે.*

15.3.3 માળખાનું પોઈન્ટર (Pointer to Structure)

નીચેનો પ્રોગ્રામ જુઓ :

```
struct student
{
char name [30] ;
int rn ;
} ;
```

અહીં વિધાન

```
student st ;
```

st *નામનો* structure student *પ્રકારનો ચલ* (variable) *જાહેર કરે છે.*

અહીં વિધાન

```
student *ptr ;
```

એ ptr *નામનો* student *માટેનો પોઈન્ટર ચલ* (variable) *જાહેર કરે છે.*

ptr *નો ઉપયોગ કરીને તે ડેટા સભ્યો* (data member)*નો ઉલ્લેખ કરી શકાય છે.*

```
ptr → name ;
ptr → rn ;
```

ડેટા સભ્યોનો ઉલ્લેખ નીચે જણાવ્યા પ્રમાણે અન્ય રીતે પણ શક્ય છે.

```
(*ptr). name ;
(*ptr).rn ;
```

15.3.4 પદાર્થો (ઓબ્જેક્ટ) માટે નિર્દેશક (Pointer to Objects)

પદાર્થો માટેનો પોઈન્ટર પ્રોગ્રામ રન કરતી વખતે પદાર્થો (objects) *બનાવવામાં ઉપયોગી છે. પદાર્થ માટેનો પોઈન્ટર પદાર્થ* (object) *ના જાહેર સભ્યો* (Public members) *ને ઉપયોગ કરવા માટે વપરાય છે.*

નીચે જણાવ્યા પ્રમાણે class student *જુઓ :*

```
class student
{
    char name [20] ;
    int rn ;
public :
    int marks ;
    void getdata ();
    void putdata () ;
} ;
void student : : getdata ()
{
    cin >> name ;
    cin >> rn ;
    cin >> marks ;
}
void student : : putdata()
{
    cout << "Name" << name << "\n" ;
    cout << "Roll no" << rn << "\n" ;
    cout << "Marks" << marks << "\n" ;
}
```

નીચે જણાવ્યા પ્રમાણે ઓબ્જેક્ટ st *અને પોઈન્ટર* ptr *જાહેર કરો :*

```
student st ;
student *ptr ;
```

આપણે બે રીતે student *ના મેમ્બર ફંક્શન* (member functions) *અને ડેટા સભ્યોનો* (data member) *ઉલ્લેખ કરી શકીએ.*

(i) **ડોટ (.) ઓપરેટરનો ઉપયોગ કરીને**

```
st. marks = 90 ;
```

```
st. getdata () ;
```

(i) *તીર ઓપરેટર* (arrow operator) *અને ઓબ્જેક્ટ પોઈન્ટરનો ઉપયોગ કરીને*

```
ptr  →   marks = 90 ;
ptr  →   getdata () ;
```

(*ptr) *ને વાપરવાનો બીજો એક રસ્તો નીચે મુજબ છે.*

```
(*ptr). marks = 90 ;
(*ptr). getdata () ;
```

15.4 **"આ" પોઈન્ટર (This pointer)**

C++ *એક ખાસ પ્રકારનો શબ્દ "આ"* (this) *વાપરે છે જે પદાર્થ* (object) *ના મેમ્બર ફંક્શનને આહવાન આપવા વપરાય છે.આ એક પોઈન્ટર છે જે એવા પદાર્થનો નિર્દેષ કરે છે જેના માટે* this *ફંક્શન બોલાવાયુ હોય. ઉદાહરણ તરીકે :*

```
class ABC
{
    int rn ;
public :
    void getdata()
    {
    cin >> this  →   rn ;
    }
    void putdata()
    {
    cout << this  →  rn ;
    } ;
    void main ()
    {
    ABC A, B ;
    A. getdata () ;
    A. putdata () ;
    B. getdata () ;
    B. putdata () ;
    }
```

જ્યારે પદાર્થ A *ધ્વારા* getdata () *અથવા* putdata () *ફંક્શન બોલાવાયુ હોય ત્યારે* this *ની અંદર પદાર્થ* A *નું સરનામું* (address) *હોય છે. તેવી જ રીતે જ્યારે પદાર્થ* B *ધ્વારા* getdata () *અથવા* putdata () *ફંક્શન બોલાવાયુ હોય ત્યારે,* this*ની અંદર પદાર્થ* B *નું સરનામું* (address) *હોય છે.*

પાઠગત પ્રશ્નો 15.1

---

1. *સામાન્ય ચલ* (ordinary variable) *કરતાં નિર્દેષક ચલ* (pointer variable) *કેવી રીતે*

અલગ છે ?

2. પોઈન્ટર એ એક
   (i) ચલનું સરનામું છે.
   (ii) હવે પછીના ચલને તપાસવા માટેનું સૂચન છે.
   (iii) શબ્દ (string)ના સરનામા માટેનો ચલ છે.
   (iv) સરનામા ચલનો માહિતી પ્રકાર (data type) છે.
3. જો ptr પોઈન્ટર એ ચલ A નો નિર્દેશ કરે, એવુ નિવેદન લખો, જે A નું નામ વાપર્યા વગર તેની સામગ્રી (content) દર્શાવે.
4. સમીકરણ *ptr માટે કહી શકાય -
   (i) ptr માટે નિર્દેશક હોઈ શકે
   (ii) ptr ની સામગ્રી (content) નો સંદર્ભ આપે.
   (iii) ptr ધ્વારા નિર્દેશાતા ચલની કિંમતનો સંદર્ભ (refer) આપે.
   (iv)ptr ના સંદર્ભને દૂર કરે (derefernce)
5. અભિવ્યક્તિ int * ને કહી શકાય -
   (i) એવો પોઈન્ટર જે int પ્રકારના ચલનો નિર્દેશ કરે
   (ii) int ની અંદર રહેલ સામગ્રીને સંદર્ભે.
   (iii) એવો પોઈન્ટર જે int પ્રકારની કિંમત હોય.
6. **ખાલી જગ્યા પુરો.**
   (અ) એક પોઈન્ટર ચલ બીજા ચલનું ....સંગ્રહ કરે છે.
   (બ) .... ને સરનામું આપનાર ઓપરેટર (address operator) કહે છે.
   (ક) t_age નામના પોઈન્ટર ટુ પૂર્ણાંક (integer) જાહેર કરવા આપણે .... લખવું જોઈએ.
   (ડ) જો A એક પૂર્ણાંક (integer) પ્રકારનો ચલ હોય તો .... તેનું સરનામું આપશે.
   (ઈ) જો ptr એ pointer to integer પ્રકારનો ચલ હોય તો તેની અંદર સંગ્રહાયેલ નંબર ...... થી આપી શકાય.
7. **નીચેનાઓ માટે સાચું અથવા ખોટું લખો.**
   (અ) પોઈન્ટર એ ચલનું સરનામુ છે.
   (બ) dereferencing ઓપરેટર (*) એ એક યુનરી (unary) ઓપરેટર છે, તે ગુણાકાર ઓપરેટર (*) થી અલગ છે જેને બે operands જોઈએ છે.
   (ક) this પોઈન્ટર એવા પદાર્થનો નિર્દેશ કરે છે જે હાલના ફંકશનને જગાડવા (invoke) વપરાયુ હોય.

નોંધ

15.5 તમે જે શીખ્યા

આ પાઠ ભણી ગયા પછી તમે શીખ્યા કે પોઈન્ટર ટુ એરે (Pointer to array), માળખાગત ચલ (structure variable) અને ક્લાસ ઓબ્જેક્ટ સાથે પોઈન્ટર કેવી રીતે વાપરવું. C++ ની અંદર this નામનો એક ખાસ પોઈન્ટર છે. આપણે this પોઈન્ટર કેવી રીતે વાપરવો તે પણ શીખ્યા.

15.6 પાઠ્યાંત પ્રશ્નો

1. જો arr એ પૂર્ણાંક (integer) નો એરે હોય તો શા માટે અભિવ્યક્તિ arr ++ એ ખોટી છે ?
2. arr [4]અને * (arr + 4) વચ્ચે શું તફાવત છે ?
3. જો જાહેર કરેલા માળખા (structure) માં પોઈન્ટર ચલ હોય, તો તે કેવી રીતે માળખાના સભ્યોનો ઉપયોગ કરી શકે ? ઉદાહરણ સાથે સમજાવો.
4. પબ્લીક ક્લાસની અંદર આવેલ ડેટા મેમ્બર અને મેમ્બર ફંકશનને પોઈન્ટરનો ઉપયોગ કરી કેવી રીતે વાપરી શકાય તે ઉદાહરણ લઈ સમજાવો.
5. "this" પોઈન્ટર શું છે તે ટૂંકમાં સમજાવો.

15.7 પાઠગત પ્રશ્નોના જવાબ

પાઠગત પ્રશ્નો 15.1

1. પોઈન્ટર ચલ એ સરનામાનો સંગ્રહ કરે છે જ્યારે સામાન્ય ચલ એ કિંમતનો સંગ્રહ કરે છે.
2. (i)
3. *ptr
4. (iv)
5. (iii)
6. (અ) સરનામુ (address)

   (બ) &

   (ક) int *t_age ;

   (ડ) **&A**

   (ઈ) *ptr
7. (અ) સાચું (બ) સાચું (ક) સાચું

16

ફાઈલો

16.1 પ્રસ્તાવના

અમુક સમયે કેટલાક એપ્લીકેશન પ્રોગ્રામનો ડેટા હાર્ડ ડિસ્ક અથવા ફ્લોપી ડિસ્ક પર સંગ્રહ કરવો જરૂરી છે. ફાઈલનો ઉપયોગ કરીને ડેટાને આવા ઉપકરણોમાં સંગ્રહાય છે.

16.2 હેતુઓ

આ પાઠ ભણ્યા પછી તમે નીચેના મુદ્દાઓ માટે સક્ષમ હશો :

- *ફાઈલમાં ડેટાનો સંગ્રહ કરી શકશો.*
- *ફાઈલમાં રહેલ ડેટાનો એક પછી એક ઉપયોગ કરી શકશો.*
- *ફાઈલની અંદર પોઈન્ટર વાપરી શકશો.*
- *ફાઈલની ખુલ્લી અથવા બંધ કરી શકશો.*

16.3 ફાઈલ (File)

ફાઈલ એક તાર્કિક રીતે જોડાયેલ રેકોર્ડનો સંગ્રહ છે. સામાન્ય રીતે પ્રોગ્રામમાં બે પ્રકારના માહિતી સંચારની (data communication) *જરૂર છે.*

(i) ડેટા ફાઈલમાં માહિતીને લખવી (writing data on a file) *:*

ડેટા કીબોર્ડ તરફથી મેમરી તરફ અને મેમરી તરફથી સંગ્રહ ઉપકરણ તરફ વહે છે.

કીબોર્ડ → મેમરી → હાર્ડ ડિસ્ક / ફ્લોપી ડિસ્ક

આને આઉટપુટ સ્ટ્રીમ (પ્રવાહ) કહેવાય છે કે જે ડેટાનો પ્રવાહ છે અને તેને ofstream.h હેડર ફાઈલની જરૂર પડે છે.

(ii) **ડેટા ફાઈલમાંથી માહિતીને વાંચવી (reading data from datafile)**

ડેટા સંગ્રહ ઉપકરણ તરફ વહે છે, જે ખાસ કરીને મોનીટર હોય છે.

ડેટાફાઈલ →મેમરી →આઉટપુટ → ઉપકરણ (સ્ક્રીન)

બાહ્ય સંગ્રહ

ઉપકરણ

(હાર્ડ ડિસ્ક/ફ્લોપી)

આને આઉટપુટ સ્ટ્રીમ કહે છે અને તેને ifstream.h હેડર ફાઈલની જરૂર પડે છે. જો ઈનપુટ સ્ટ્રીમ અને આઉટપુટ સ્ટ્રીમ બંને એક જ પ્રોગ્રામમાં વપરાયા હોય તો તેને fstream.h ફાઈલની જરૂર પડે છે.

જો પ્રોગ્રામમાં fstream.h હેડર ફાઈલ સમાવવામાં આવેલી હોય તો iostream.h ફાઈલને અલગથી લખવાની જરૂર નથી.

16.3.1 ફાઈલને ખોલવી (opening a file)

ફાઈલને બે રીતે ખોલી શકાય છે :

(i) ક્લાસના કંસ્ટ્રક્ટર (constructor) ફંકશનને વાપરીને

(ii) ક્લાસના મેમ્બર ફંકશન ઓપન (open()) નો ઉપયોગ કરીને

કંશટ્રકશન ફંકશનનો ઉપયોગ કરી ફાઈલ ખોલવી :

નીચેના નિવેદન STU.DAT ફાઈલને આઉટપુટ સ્થિતિ (mode) એટલે કે ફાઈલની અંદર ડેટા લખવા માટે ખોલશે.

```
ofstream outfile ("STU.DAT") ;
```

ofstream એ કંમ્પાઈલર ફાઈલમાં રહેલો એક ક્લાસ છે.

outfile એ વપરાશકર્તાએ વ્યાખ્યાયિત કરેલ પદાર્થ (object) છે.

નીચેના નિવેદનો

```
outfile << "TOTAL MARKS" << "\n" ;
outfile << total << "\n" ;
```

ફાઈલમાં ડેટા લખવા માટે વપરાય છે. New line ("\n") અક્ષર એ પોઈન્ટરને નવી લાઈનમાં ખસેડે છે.

તેવી જ રીતે નીચેનું નિવેદન

```
ifstream infile ("STU.DAT") ;
```

એ STU.DAT ફાઈલને ઈનપુટ સ્થિતિ એટલે કે વાંચવા માટે ખોલે છે.

નિવેદનો,

```
infile >> string ;
infile >> number ;
```

એ ડેટાને ફાઈલમાંથી વાંચે છે.

નીચે આપેલ પ્રોગ્રામ, લખવા અને વાંચવા માટે એક જ ફાઈલનો ઉપયોગ કરે છે. પહેલા તે કીબોર્ડ પરથી ડેટા લઈ અને ફાઈલમાં લખે છે. લખાણ પતી જાય પછી તે ફાઈલને બંધ કરે છે. પ્રોગ્રામ ફરીથી તે જ ફાઈલને ખોલે છે, તેમાં રહેલી માહિતીને વાંચે છે અને સ્ક્રીન પર બતાવે છે.

```
# include <iostream.h>
void main()
{
    char name [30] ;
    int rn, marks ;
    ofstream outfile ("INF") ;
    cout <<"Enter student name" ;
    cin >> name ;
    cout << "Enter student roll number" ;
    cin >> rn ;
    cout << " Enter student marks" ;
    cin >> marks ;
    outfile << name << "\n" ;
    outfile << rn << "\n" ;
    outfile << marks << "\n" ;
    outfile.close() ;
    ifstream infile ("INF") ;
    infile >> name ;
    infile >> rn ;
    infile >> marks ;
    cout << "Name" << name << "\n" ;
    cout << "Roll No" << rn << "\n" ;
    cout << "Marks" << marks << "\n" ;
    inlife. close () ;
    }
```

આ પ્રોગ્રામનો આઉટપુટ હશે :

Enter student name   MOHIT

Enter student roll number	20
Enter student marks	90
Name	MOHIT
Roll No	20
Marks	90

ઓપન ફંકશન વાપરીને ફાઈલને ખોલવી (opening a file using open() function) :

ઓપન ફંકશન (વિધેય) ઘણીબધી ફાઈલો સાથે ઉપયોગ કરી શકાય કે જે એકજ સ્ટ્રીમ પદાર્થ (ઓબ્જેક્ટ) વાપરે છે. પહેલાં સ્ટ્રીમ ઓબ્જેક્ટ સોંપાય (assigned) *છે અને પછી ફાઈલને ખોલવા તેનો ઉપયોગ થાય છે.*

```
filestream_class stream_object ;
stream_object.open ("filename") ;
```

ઉદાહરણ તરીકે :

```
ofstream outfile ;
outfile.open ("ABC") ;
................
outfile.close() ;
outfile.open("XYZ") ;
................
outfile.close() ;
```

16.3.2 ઓપન વિધેય (open function)

ઓપન વિધેયને બે પરિમાણો (parameters) *હોય છે : ફાઈલનું નામ અને વાપરવાની સ્થિતિ* (access mode). *તેનુ સામાન્ય બંધારણ નીચે મુજબ છે :*

```
stream_object.open("filename",access mode) ;
```

બીજી દલીલ દર્શાવે છે કે ફાઈલ કઈ સ્થિતિમાં ખોલી છે. ifstream *અથવા* ofstream *વિધેયો માટે ડિફોલ્ટ કિંમત લેવાય છે (બાહ્ય રીતે સ્થિતિ દર્શાવતી નથી).*

ifstream *વિધેય માટે* ios::in

ofstream *વિધેય માટે* ios::out

ફાઈલની સ્થિતિ દર્શાવતા પરિમાણો (parameters), ios *ક્લાસમાંથી એક અથવા વધારે સ્થિરાંકો* (constants) *લઈ શકે.નીચેનું કોષ્ટક ફાઈલની સ્થિતિ દર્શાવતા પરિમાણો બતાવે છે.*

પરિમાણ (Parameter)	**અર્થ** (Meaning)
ios :: app	*તે ફાઈલને આઉટપુટ સ્થિતિમાં ખોલે છે. ફાઈલ પોઈન્ટરએ ફાઈલના છેડે હોય છે અને તે નવો રેકોર્ડ ઉમેરી શકે છે.*
ios :: ate	*ફાઈલ પોઈન્ટર ફાઈલના છેડે હોય છે અને તે ડેટા ઉમેરવા અથવા હાલનો ડેટા ફાઈલમાં ગમે તે જગ્યાએ બદલવા થાય છે.*

ios :: binary	દ્વિસંગી (binary) ફાઈલ
ios :: in	તે ફાઈલને ઈનપુટ સ્થિતિમાં ખોલે છે. ફાઈલ પોઈન્ટર ફાઈલની શરૂઆતમાં હોય છે અને તે વાંચવા માટે તૈયાર હોય છે.
ios :: nocreate	જો ફાઈલ પહેલેથી હાજર હશે તો ખૂલશે નહીં તો ઓપન નિષ્ફળ જાય છે.
ios :: noreplace	જો ફાઈલ હાજર નહીં હોય તો ફાઈલ ખોલશે નહીં તો ઓપન નિવેદન નિષ્ફળ જાય છે.
ios :: out	તે ફાઈલને આઉટપુટ સ્થિતિમાં ખોલે છે. ફાઈલ પોઈન્ટર ફાઈલના છેડે હોય છે. જો તેમાં પહેલેથી ડેટા હોય તો આઉટપુટ સ્થિતિ ફાઈલના ડેટાને ભૂંસી નાંખે છે.
ios :: trunc	જો હાજર હોય તો, તે ફાઈલના ડેટાને કાઢી નાંખે (delete) છે.

બીટવાઈઝ ઓર ઓપરેટર (bitwise OR operator) વાપરીને ફાઈલની સ્થિતિના બે કે તેથી વધારે પરિમાણો ભેગા કરી શકાય છે.

ઉદાહરણ :

```
outfile .open ("ABC",ios :: in|ios :: out|ios :: binary) ;
```

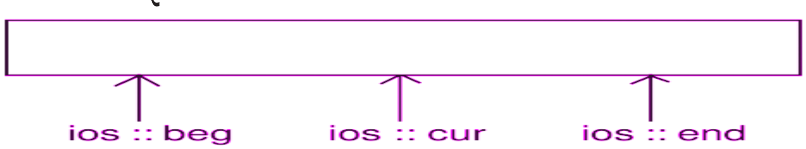
16.3.3 ફાઈલ નિર્દેશક (File Pointer)

ફાઈલની સાથે બે નિર્દેશક સંકળાયેલ છે, ઈનપુટ પોઈન્ટર (અથવા ગેટ પોઈન્ટર) અને આઉટપુટ પોઈન્ટર (અથવા પુટ પોઈન્ટર). દરેક વખતે જ્યારે ઈનપુટ અથવા આઉટપુટની કામગીરી થાય છે ત્યારે પોઈન્ટર તેની જાતે ખસે છે.

seekg() તે ગેટ પોઈન્ટર (get pointer) ને દર્શાવેલ જગ્યાએ ખસેડે છે.

seekp() તે પુટ પોઈન્ટર (put pointer) ને દર્શાવેલ જગ્યાએ ખસેડે છે.

ફાઈલને નીચેના સ્વરૂપે જોઈ શકાય છે.



ios :: beg એ ફાઈલની શરૂઆત દર્શાવે છે.

ios :: cur એ ફાઈલ પોઈન્ટરની અત્યારની સ્થિતિ દર્શાવે છે.

ios :: end એ ફાઈલનો અંત દર્શાવે છે.

seekg()અને seekp() વિધાનને બે પરિમાણો છે.

```
object.seekg (no.of bytes, refposition) ;

object.seekp (no.of bytes, refposition) ;
```

refposition એ ઉપર જણાવ્યા પ્રમાણે ios ક્લાસમાંના ત્રણ સ્થિરાંકોમાંથી કોઈ એક લે

છે.

ઉદાહરણ 1

```
infile.seekg (0,ios :: beg) ;
```

તે પોઈન્ટરને ફાઈલની શરૂઆતમાં ખસેડે છે. આ કિસ્સામાં, refposition ios :: beg એ વૈકલ્પિક છે.

```
infile.seekg (100, ios :: cur) ;
```

તે પોઈન્ટરને વર્તમાન સ્થળથી 100 બાઈટ્સ ખસેડે છે.

```
infile.seekg (-200, ios :: end) ;
```

તે પોઈન્ટરને ફાઈલના છેડાથી ૨00 બાઈટ્સ આગળ ખસેડે છે.

16.3.4 tellg() અને tellp () વિધેય

tellg () વિધેય એ get pointerની પરિસ્થિતિ બાઈટ્સ (bytes) માં આપે છે. એ જ રીતે tellp() વિધેય put pointer ની પરિસ્થિતિ બાઈટ્સમાં આપે છે.

ઉદાહરણ 2

```
ifstream  infile ;
infile.open ("ABC", ios :: ate) ;
int B = infile.tellg() ;
```

ઉપરના નીવેદનોના અમલ પર, ઈનપુટ પોઈન્ટરએ ફાઈલના છેડા સુધી ખસેડાય છે અને B ફાઈલમાં બાઈટ્સની સંખ્યા આપે છે.

16.3.5 Write() અને read() વિધેયો

વિધેય write()અને read() ને બે પરિમાણો છે : ચલનું સરનામું અને કદ. ચલનું સરનામું char* પ્રકારમાં જ હોવું જોઈએ. તેનું સામાન્ય બંધારણ નીચે મુજબ છે :

```
infile.read ((char *)&V, sizeof v) ;
outfile.write ((char *) &V, sizeof v) ;
```

જ્યાં V એક ચલ છે.

16.3.6 Close () વિધેય

જે ફાઈલને constructor અથવા open() વિધેય ધ્વારા ખોલી હોય તેને છેલ્લે બંધ કરવી જોઈએ. તેનું સામાન્ય બંધારણ નીચે મુજબ છે :

```
stream_object.close() ;
```

નીચેનું ઉદાહરણ ક્લાસના પદાર્થ (object) સાથે કામ કરે છે અને નીચે જણાવ્યા પ્રમાણેના કાર્યો કરે છે :

- ફાઈલ બનાવે

- ફાઈલને દર્શાવે
- નવી માહિતી (રેકોર્ડ) ઉમેરે
- હાલના રેકોર્ડ સુધારે

ઉદાહરણ 3

```
# include<fstream.h>
class student
{
char name [30] ;
int rn ;
public :
void getdata() ;
void putdata() ;
} ;
void student :: getdata()
{
cout << "Enter student name" ;
cin >> name ;
cout << " Enter roll number" ;
cin >> rn ;
}
void student :: putdata()
{
cout << " Student name" << name << "\n" ;
cout << " Student roll number" << rn << "\n" ;
}
void main()
{
fstream file ;
file.open ("ABC", ios :: in |ios :: out |ios :: binary) ;
student st ;
//create a data file
int i, n ;
cout << "How many record to enter" ;
cin >> n ;
for ( i =1; i < = n ; i ++)
{
st. getdate() ;
```

```
file. write ((char*) & st, size of st);
}
//Display a data file
file. seekg( o, ios:: beg) ;
while (file. read ((char*) & st, size of st))
{
st.putdata( );
}
file.clear ( ) ; //To make the end of file mark false
// To append record
st. getdata( );
file.write ((char*) & st, sizeof st);
// To modify a record
file.clear ( ) ;
cout << "Enter record number" ;
cin >> n;
file.seekp ((n-1) * size of st, ios ::  beg);
st.getdata ( ) ;
file. write ((char*) & st, size of st);
// To close a file
file.close ( ) ;
```

પાઠગત પ્રશ્નો 16.1

1. *ફાઈલ શું છે ?*
2. ios :: ate *અને* ios :: app *સ્થિતિ વચ્ચે શું તફાવત છે ?*
3. *બે પ્રકારના ફાઈલ પોઈન્ટર કયા છે ?*
4. *સાચું કે ખોટું કહો : મૂળભુત રીતે* ios :: out *સ્થિતિ એ* ios :: trunc *સ્થિતિ છે.*
5. tellg ( ) *વિધેયનું મહત્વ શું છે ?*
6. **ખાલી જગ્યા પુરો :**
   (અ) ....એ તાર્કિક રીતે જોડાયેલ રેકોર્ડનો સંગ્રહ છે.
   (બ) ofstreum*થી ખોલેલ ફાઈલ ફક્ત .... માટે જ ઉપલબ્ધ છે.*
   (ક) ifstream *થી ખોલેલ ફાઈલ ફક્ત .... માટે જ ઉપલબ્ધ છે.*
   (ડ) આપણે ફાઈલને .... વિધેય વાપરીને ખોલી શકીએ.
   (ઈ) mode opp *ફાઈલને .... માટે ખોલે છે.*
   (ઈ) આઉટપુટ સ્થિતિમાં ખોલેલ ફાઈલ મૂળભૂત રીતે .... હોય છે.
7. **નીચેનાઓ માટે સાચું અથવા ખોટું જણાવો.**

(અ) ફાઈલએ રેકોર્ડનો સંગ્રહ છે.

(બ) ofstream માં ખોલેલી ફાઈલ લખવા માટે ઉપલબ્ધ હોય છે.

(ક) ફાઈલ ખોલવા માટેનો output mode, માહિતી (Contents) જો ફાઈલમાં હોય તો તેને કાઢી નાંખે છે.

(ડ) close ( ) વિધેય ફાઈલને બંધ કરવા વપરાય છે.

(ઇ) નીચેનું નિવેદન

outfile.write ((char*) & obj, sizeof obj);

એ ફક્ત objમાં રહેલ ડેટાને outfile માં લખશે.

(ઈ) ios :: ate સ્થિતિ આપણને ફક્ત ફાઈલના છેડે લખવાની પરવાનગી આપે છે.

(ઉ) આપણે append mode માં ફાઈલને ખોલીને તેમાં ડેટા ઉમેરી શકીએ છીએ.

(ઊ) write ( ) વિધેયનો ઉપયોગ કરીને ફાઈલમાં લખાયેલ ડેટા, get ( ) વિધેય વાપરીને વાંચી શકાય છે.

16.4. તમે જે શીખ્યા

આ પાઠમાં તમે ફાઈલ પ્રોગ્રામીંગ વિશે શીખ્યા જે કોઈપણ ભાષાનું અગત્યનું પાસું છે. અમે તમને એવા ઉદાહરણો આપ્યા છે જે તમને પ્રોગ્રામમાં ફાઈલ ઉપયોગ કરવામાં મદદ કરશે.

16.5. પાઠ્યાંત પ્રશ્નો (Terminal Questions)

1. ઈનપુટ અને આઉટપુટ સ્ટ્રીમ શું છે ?
2. Constructor વિધેય વાપરીને ફાઈલ ખોલવી અને Open ( ) વિધેય વાપરીને ફાઈલ ખોલવામાં શું તફાવત છે ?
3. ફાઈલ એક્સેસ મોડ (access mode) શું છે? ફાઈલના અલગ અલગ મોડ (સ્થિતિ) વિશે સમજાવો.
4. એક ફાઈલમાં ૫ રેકોર્ડ છે, દરેક 100 બાઈટ્સ જગ્યા રોકે છે :

```
fstream file;
file. seekg ( O, ios :: end);
N = file.tellg ( ) ;
```

(i) N ની ડેટા ટાઈપ શું હશે ?

(ii) N ની કિંમત શું હશે ?

5. **નીચેના વાક્યો લક્ષમાં લો :**

```
fstream file ;
file. open ( "ABC", ios :: in  ios :: out);
```

નીચેનાઓ માટે C++ માં નિવેદન લખો : |

(i) *પોઈન્ટરને ફાઈલની શરૂઆતમાં ખસેડવા.*

(ii) *પોઈન્ટરને ફાઈલના અંતમાં ખસેડવા.*

(iii) *બાઈટ્સની કુલ સંખ્યા શોધવા.*

(iv) *ફાઈલને બંધ કરવા.*

6. **નીચેનાની કામગીરી સમજાવો :**

(i) fstream file;

(ii) file.seekg(100, ios ::cur);

(iii) file.seekg(-100, ios ::end);

(iv) file. seekg (100, ios :: beg);

7. *રેકોર્ડ બે ફીલ્ડનો બનેલો છે :* name *અને* roll no.*એવો પ્રોગ્રામ લખો જે નીચેના કાર્યો કરે :*

(i) 5 *રેકોર્ડ ધરાવતી ડેટા ફાઈલ બનાવો*

(ii) *ડેટા ફાઈલ બતાવો.*

(iii) *રેકોર્ડ ઉમેરો*

(iv) *કોઈ એક રેકોર્ડને સુધારવો*

16.6 પાઠગત પ્રશ્નોના જવાબો (feedback to Intext Questions)

પાઠગત પ્રશ્નો 16.1

1. *તે તાર્કિક રીતે જોડાયેલા રેકોર્ડનો સમુહ છે.*
2. ios :: ate mode *રેકોર્ડ ઉમેરી શકે અથવા પહેલેથી હાજર રેકોર્ડને સુધારી શકે છે જયારે* ios :: add mode *ફક્ત રેકોર્ડ ઉમેરી શકે છે.*
3. put *અને* get
4. *સાચું*
5. *તે* get pointer *ની બાઈટ્સના સ્વરૂપે સ્થિતિ દર્શાવે છે.*
6. *(અ) ફાઈલ*

   (બ) આઉટપુટ

   (ક) ઈનપુટ

   (ડ) Open

(ઇ) આઉટપુટ

(ઈ) trunc

7. (અ) સાચું

(બ) સાચું

(ક) સાચું

(ડ) સાચું

(ઇ) સાચું

(ઈ) ખોટું

(ઉ) સાચું

(ઊ) ખોટું.

પુર્ણ કરી આજે પ્રતિક્રિયા ફોર્મ પોસ્ટ કરો

અંતિમ ફોલ્ડ અને સીલ

પ્રતિક્રિયા અને પાઠો

પાઠ નં.	પાઠનું નામ	વિષયવસ્તુ			ભાષા		ચિત્ર		તમે શું શીખ્યા	
		મુશ્કેલ	મનોરંજક	મુંઝવતું	સાદી	જટિલ	ઉપયોગી	બિન ઉપયોગી	ખુબ જ ઉપયોગી	બિન ઉપયોગી
1.										
2.										
3.										
4.										
5.										
6.										
7.										
8.										
9.										
10.										
11.										
12.										
13.										

---ચોથું ફોલ્ડ---

---ત્રીજો ફોલ્ડ---

વિદ્યાર્થી મિત્રો,

તમે આ આખા સ્વાધ્યાય પુસ્તકને ચોકકસપણે આનંદથી માણ્યું હશે. અહી અમારો પ્રયત્ન આ અભ્યાસ સામગ્રીને સુસંગત, અસરકારક અને રસપ્રદ બનાવવાનો રહ્યો છે. અભ્યાસ સામગ્રી તૈયાર કરવી એ દ્વિમાર્ગી પ્રક્રિયા છે. તમારા પ્રતિભાવો અમને આ અભ્યાસ સામગ્રીને સુધારવામાં મદદરૂપ થશે. થોડો સમય કાઢીને અહી આપેલ પ્રતિભાવપત્રક ભરો જેથી વધુ રસપ્રદ અને ઉપયોગી સામગ્રી બનાવી શકાય.

આભાર

કો–ઓર્ડિનેટર

(**નામા પધ્ધતિ**)

પ્રતિક્રિયા ઉપર પ્રશ્નો

પાઠ નં.	પાઠનું નામ	Intext પ્રશ્નો		ટર્મિનલપ્રશ્નો		
		ઉપયોગી	બિનઉપયોગી	સહેલા	મુશ્કેલ	ખુબ મુશ્કેલ
1.						
2.						
3.						
4.						
5.						
6.						
7.						
8.						
9.						
10.						
11.						
12.						
13.						

બીજો ફોલ્ડ

તમારું સુચન

____

____

____

____

શું તમે મનોવિજ્ઞાનના અભ્યાસ માટે બીજા કોઈ પુસ્તકનો ઉપયોગ કર્યો ?

હા / ના

જો હા, તો ઉપયોગ કરવાનું કારણ આપો.

____

____

નામ : ______________________

એનરોલમેન્ટ નં. ______________________

સરનામું : ______________________

____

વિષય : ______________________

પુસ્તક નં. : ______________________

ટપાલ
ટિકિટ

મદદનીશ નિયામક (શૈક્ષણિક)

રાષ્ટ્રીય મુકત વિદ્યાલય શિક્ષણ સંસ્થાન

(MHRD), ભારત સરકાર હસ્તક (સ્વાયત્ત) સંસ્થા)

એ-૨૪-૨૫, ઇન્સ્ટિટયૂશનલ વિસ્તાર, સેક્ટર- ૬૨, નોઈડા-

૨૦૧૩૦૯ (યુ. પી.)

કોઈ જોડાણ માન્ય નથી