C પ્રોગ્રામિંગનો પરિચય

પરિચય

અગાઉના પ્રકરણમાં, આપણે અલ્ગોરિધમ કેવી રીતે ડિઝાઈન કરવા અને ફ્લોચાર્ટ કેવી રીતે દોરવા તે શીખ્યા. કોડિંગ શરૂ કરીએ તે પહેલાં, આ વસ્તુઓ આપણને સ્ટેપ-બાય-સ્ટેપ ઉકેલનું આયોજન કરવામાં અને તેને સમજવામાં મદદ કરે છે. પરંતુ આયોજન એ માત્ર પ્રથમ ભાગ છે. આપણા ઉકેલને કમ્પ્યૂટર પર કાર્યરત કરવા માટે, આપણે આ તાર્કિક પગલાંઓને Python, Java કે C જેવી ચોક્કસ પ્રોગ્રામિંગ ભાષા (Programming Language) માં વ્યક્ત કરવા આવશ્યક છે. આ ભાષાઓમાં, C ખૂબ જ મહત્વપૂર્ણ છે. તે ઝડપી અને શક્તિશાળી હોવા ઉપરાંત ઘણી આધુનિક પ્રોગ્રામિંગ ભાષાઓનો આધાર છે. આ પ્રકરણમાં, આપણે C પ્રોગ્રામિંગ શીખવાની શરૂઆત કરીશું. આપણે સરળ C પ્રોગ્રામ કેવી રીતે લખવા, C પ્રોગ્રામના મુખ્ય ઘટકોને સમજવા અને C પ્રોગ્રામમાં વપરાતા મૂળભૂત અક્ષરો, સંખ્યાઓ અને પ્રતીકો વિશે જાણીશું. કમ્પ્યૂટર પ્રોગ્રામિંગની દુનિયામાં આ આપણું પ્રારંભિક પગલું છે.

પ્રોગ્રામિંગ ભાષાની આપણને શી જરુરીયાત છે?

પ્રાકૃતિક ભાષાઓમાં ઘણીવાર અસ્પષ્ટતા (ambiguity) જોવા મળે છે, જ્યાં એક જ વાક્યના અનેક અર્થઘટન (interpritations) થઈ શકે છે. કમ્પ્યૂટર સાથે વાતચીત કરતી વખતે આ એક મોટો પડકાર છે, કારણ કે કમ્પ્યૂટરને ચોક્કસ અને સ્પષ્ટ હોય તેવી સૂચનાઓની જરૂર હોય છે. મનુષ્યોથી વિપરીત, કમ્પ્યૂટર અસ્પષ્ટ સૂચનાઓમાંથી અર્થ તારવી શકતા નથી. પ્રોગ્રામિંગ ભાષાઓ કડક નિયમો લાગુ કરીને આ સમસ્યાનું નિરાકરણ લાવે છે. આ નિયમો સુનિશ્ચિત કરે છે કે દરેક સૂચનાનો માત્ર એક જ ઉદ્દેશિત અર્થ હોય. પ્રોગ્રામિંગ ભાષા શીખવી એ કોમ્યુનિકેશનનું એક નવું સ્વરૂપ છે, જેના દ્વારા આપણે કમ્પ્યૂટરને ચોક્કસ કાર્યો કરવા માટે અસરકારક રીતે આદેશ આપી શકીએ છીએ.

પ્રોગ્રામ અને પ્રોગ્રામિંગની સમજણ

C ભાષા શીખતા પહેલાં, ચાલો આપણે પ્રોગ્રામિંગના મુખ્ય ખ્યાલોથી શરૂઆત કરીએ. પ્રોગ્રામ એ સૂચનાઓનો એક ચોક્કસ ક્રમ છે જે કમ્પ્યૂટરને ગણતરીઓ કરવા અથવા ડેટા પ્રદર્શિત કરવા જેવા કાર્યો કરવા માટે નિર્દેશિત કરે છે. કમ્પ્યૂટર ફક્ત બાઈનરી કોડ્સ (0 અને 1) પર પ્રક્રિયા કરતા હોવાથી, આપણે C જેવી માળખાગત પ્રોગ્રામિંગ ભાષાઓનો ઉપયોગ કરીએ છીએ જે સ્પષ્ટ હોય તેવી વાક્યરચના (statement) લાગુ કરે છે. ત્યારબાદ, કમ્પાઈલર (compiler) નામનું વિશિષ્ટ સોફ્ટવેર આ માનવ-વાંચનીય કોડને મશીન ભાષા (બાઈનરી કોડ્સ) માં અનુવાદિત કરે છે, જેનો કમ્પ્યૂટર અમલ કરી શકે છે. પ્રોગ્રામિંગ ભાષાઓ જુદી જુદી શ્રેણીઓમાં અસ્તિત્વ ધરાવે છે: ઉચ્ચ-સ્તરીય ભાષાઓ (High-level languages) માનવ-વાંચનીયતા અને અમૂર્તતા (abstraction) ને પ્રાથમિકતા આપે છે. જ્યારે નિમ્ન-સ્તરીય ભાષાઓ (low-level languages) જટિલતાના ભોગે હાર્ડવેર નિયંત્રણ પ્રદાન કરે છે. C ભાષા અનન્ય રીતે આ બંને શ્રેણીઓ વચ્ચે સેતુ બાંધે છે તેમજ કાર્યક્ષમતા અને અભિવ્યક્ત શક્તિ બંને પૂરી પાડે છે.

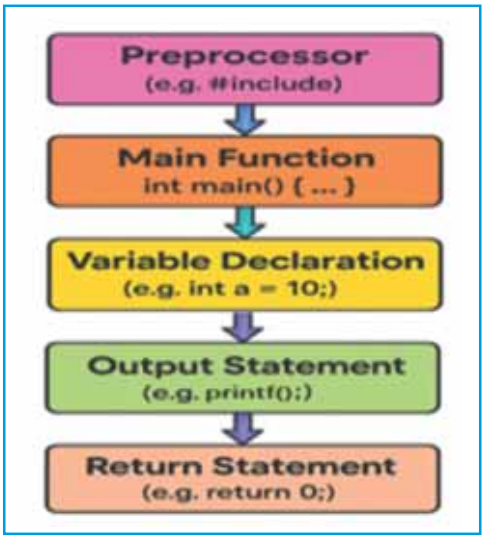
C ભાષાનો ઈતિહાસ

C ભાષાનો વિકાસ કમ્પ્યૂટર વૈજ્ઞાનિક ડેનિસ રિચી (Dennis Ritchie) એ 1972 માં બેલ લેબ્સ (Bell Labs) ખાતે કર્યો હતો. B અને BCPL જેવી અગાઉની પ્રોગ્રામિંગ ભાષાઓમાં જોવા મળતી મર્યાદાઓને દૂર કરવા માટે C પ્રોગ્રામિંગ ભાષા બનાવવામાં આવી હતી. રિચીનો ઉદ્દેશ એક એવી પ્રોગ્રામિંગ ભાષા બનાવવાનો હતો જે સરળ,

ઝડપી હોય અને કમ્પ્યૂટર હાર્ડવેર સાથે સીધો સંવાદ કરી શકે. આજે વર્ષો વીતી ગયા હોવા છતાં, C વિશ્વભરમાં વ્યાપકપણે ઉપયોગમાં લેવાય છે કારણ કે તે પ્રમાણમાં શીખવામાં સરળ છે, ઘણી જુદી જુદી કમ્પ્યૂટર સિસ્ટમ પર કાર્યક્ષમ રીતે ચાલી શકે છે અને પ્રોગ્રામરને શક્તિશાળી અને કાર્યક્ષમ કોડ લખવાની સવલત આપે છે. તેના મુખ્ય ખ્યાલોએ તેના પછી ઉદ્ભવેલી ઘણી મુખ્ય કમ્પ્યૂટર ભાષાઓને ભારે પ્રભાવિત કરી. આજે પણ, C શાળાઓમાં શીખવવામાં આવતી અને ઈન્ડસ્ટ્રીમાં સોફ્ટવેર સિસ્ટમ બનાવવા માટે વપરાતી એક આધારભૂત ભાષા બની રહી છે.

C પ્રોગ્રામનો બ્લોક ડાયાગ્રામ

આકૃતિ 6.1, C પ્રોગ્રામનો મૂળભૂત પ્રવાહ દર્શાવે છે. તે *#include* જેવી પ્રીપ્રોસેસર ડાયરેક્ટિવથી શરૂ થાય છે, ત્યારબાદ main ફંક્શન (જેમ કે *int main()*) આવે છે. પછી, ચલો (variables) ઘોષિત કરવામાં આવે છે. *printf()* નો ઉપયોગ કરીને આઉટપુટ (પરિણામ) પ્રિન્ટ કરવામાં આવે છે, અને પ્રોગ્રામ સામાન્ય રીતે *return 0*; વિધાન (statement) સાથે સમાપ્ત થાય છે, જે સફળતાપૂર્વક અમલ (execution) પૂર્ણ થયાનું સૂચવે છે. C પ્રોગ્રામ લખવાનું પ્રથમ પગલું એ છે કે તેને એડિટર (editor)નો ઉપયોગ કરીને ટાઈપ કરવો. આપણે આપણા કમ્પ્યૂટર સિસ્ટમ પર ઉપલબ્ધ કોઈપણ એડિટરનો ઉપયોગ કરી શકીએ છીએ. GCC જેવા કમ્પાઈલરનો ઉપયોગ C પ્રોગ્રામને કમ્પાઈલ (compile – મશીન કોડમાં રૂપાંતર) કરવા માટે કરી શકાય છે.



આકૃતિ 6.1 : C પ્રોગ્રામનો મૂળભૂત પ્રવાહ

પગલાંઓ : પ્રોગ્રામ બનાવી, સંગ્રહી, ખોલી અને ચલાવવાની પ્રક્રિયા – hello.c

1. એડીટર ઓપન કરો
2. તમારો કોડ ટાઈપ કરો
3. **File → Save As** ક્લિક કરો.
4. ફાઈલને નામ આપો : hello.c
5. ખાત્રી કરો કે ફાઈલ ".c" એક્સ્ટેંશન (.txt નહીં) સાથે સંગ્રહ કરી છે

નીચેનો C પ્રોગ્રામ સ્ક્રીન પર "Hello, World!" કેવી રીતે પ્રિન્ટ કરવું તે દર્શાવે છે.

```
/* C Program to print Hello World! */
#include <stdio.h>
int main()
{
        printf("Hello, World!\n");
        return 0;
}

Result:
Hello, World!
```

આ એક સાદો પ્રોગ્રામ છે, જે નવા શીખનારાઓ માટે C પ્રોગ્રામિંગ કેવી રીતે કાર્ય કરે છે તે સમજવા માટે છે. C પ્રોગ્રામ *#include <stdio.h>* લાઈનથી શરૂ થાય છે, જે કમ્પ્યૂટરને સ્ટાન્ડર્ડ ઈનપુટ/આઉટપુટ (Standard

Input/Output) લાઇબ્રેરીનો સમાવેશ કરવા જણાવે છે. આનાથી આપણે સ્ક્રીન પર આઉટપુટ દર્શાવવા માટે *printf()* જેવા ફંક્શનનો ઉપયોગ કરી શકીએ છીએ. બીજી લાઇન, *int main()*, પ્રોગ્રામનો પ્રારંભ દર્શાવે છે. આ પછી, ખૂલતો કર્લી બ્રેસ { ફંક્શનના મુખ્ય ભાગની શરૂઆત સૂચવે છે, જ્યાં પ્રોગ્રામની સૂચનાઓ લખવામાં આવશે. આ ભાગની અંદર, *printf("Hello, World!\n")*; લાઇનનો ઉપયોગ સ્ક્રીન પર "Hello, World!" સંદેશ પ્રિન્ટ કરવા માટે થાય છે અને સ્ટ્રિંગમાં રહેલ \n (ન્યૂ લાઇન કેરેક્ટર) કર્સરને આગલી લાઇન પર ખસેડે છે. *return 0*; લાઇન કમ્પ્યૂટરને જણાવે છે કે પ્રોગ્રામ સફળતાપૂર્વક તેનો અમલ (execution) પૂર્ણ કરી ચૂક્યો છે. અંતે, બંધ થતો કર્લી બ્રેસ } *main()* ફંક્શનનો અંત દર્શાવે છે. આ સરળ પ્રોગ્રામ એક સામાન્ય C પ્રોગ્રામની મૂળભૂત રચના અને કાર્ય દર્શાવે છે. C પ્રોગ્રામ લખવા, સેવ કરવા, કમ્પાઇલ કરવા અને અમલ કરવાના પગલાં પ્રકરણના અંતમાં આપેલા છે.

C ભાષાનો કેરેક્ટર સેટ (Character Set in C Language)

C પ્રોગ્રામિંગમાં, કેરેક્ટર સેટ (Character Set) એ અક્ષરોના સમૂહને દર્શાવે છે જેનો ઉપયોગ પ્રોગ્રામ લખવા માટે થાય છે. જેમાં અક્ષરો (letters), અંકો (digits), ખાસ પ્રતીકો (special symbols), ખાલી જગ્યાઓ (white spaces) અને અન્ય નિયંત્રણ અક્ષરો (control characters)નો સમાવેશ થાય છે. કેરેક્ટર સેટને સમજવું આવશ્યક છે કારણ કે દરેક પ્રોગ્રામ માન્ય અક્ષરોથી બનેલો હોય છે, જે ભાષાના ચલ, ફંક્શન, ઓપરેટર અને અન્ય ઘટકોની રચના કરે છે. ટેબલ 6.1 C ભાષાનો કેરેક્ટર સેટ દર્શાવે છે.

કેરેક્ટર સેટ કેટેગરી	વર્ણન	ઉદાહરણ
1. અક્ષરો (Letters)	C ભાષામાં અક્ષરો (Alphabats)નો ઉપયોગ આઇડેન્ટિફાયર (identifiers) (જેમ કે ચલ), કીવર્ડ્સ અને અન્ય નામોની રચના કરવા માટે થાય છે. આઇડેન્ટિફાયર જાહેર કરતી વખતે અપરકેસ (મોટા અક્ષરો) અને લોઅરકેસ (નાના અક્ષરો) બંને માન્ય છે.	• અપરકેસ-મોટા અક્ષરો (A–Z) • લોઅરકેસ-નાના અક્ષરો (a–z) **ઉદાહરણ :** ચલોના નામ ઘોષિત કરવા • int Age; char name;
2. અંકો (Digits)	અંકો (Digits) નો ઉપયોગ અચલ (constants), એરે ઇન્ડેક્સ (array index), વગેરેમાં આંકડાકીય મૂલ્યો (numeric values) રજૂ કરવા માટે થાય છે.	• 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 **ઉદાહરણ :** અચલ, એરે ઇન્ડેક્સ માટે • int num = 45; • int num[5];
3. ખાસ પ્રતીકો (Special symbol / characters)	ખાસ પ્રતીકો (Special Symbols) એવા પ્રતીકો છે જેનો C ભાષામાં વિશિષ્ટ અર્થ હોય છે. તેનો ઉપયોગ ઓપરેટરો (operators), ડિલિમિટર્સ (delimiters), કોડની રચના (structuring code) અને અર્થને વ્યાખ્યાયિત કરવા માટે થાય છે.	• ગણિતીય ઓપરેટર જેવા કે +, –, *, / • Address-of ઓપરેટર (&) • તાર્કિક AND (&&), તાર્કિક OR (||) વગેરે • કોડ બ્લોક માટે કર્લી બ્રેસીસ { } • એસાઇન્મેન્ટ ઓપરેટર (=) અને • બીજા ઘણાં જેવાકે = =, (), [], #, %, !, ^

ટેબલ 6.1 : C કેરેક્ટર સેટ

વ્હાઇટ સ્પેસ (White Spaces)

C પ્રોગ્રામિંગમાં, વ્હાઇટ સ્પેસ (ખાલી જગ્યા) નો ઉપયોગ પ્રોગ્રામિંગ કોડમાં શબ્દો, પ્રતીકો અને ઘટકોને અલગ કરવા માટે થાય છે. તેઓ પ્રોગ્રામના અમલ પર અસર કરતા નથી, પરંતુ કોડને વાંચવા યોગ્ય (readable) અને વ્યવસ્થિત (organized) બનાવવા માટે મહત્વપૂર્ણ છે. સામાન્ય વ્હાઇટ સ્પેસ અક્ષરોમાં સ્પેસ (જગ્યા), ટેબ (tab) અને નવી લાઈન (newlines) નો સમાવેશ થાય છે. ટેબલ 6.2 C માં ઉપયોગમાં લેવાતા વિવિધ પ્રકારના વ્હાઇટ સ્પેસ પ્રતિકો દર્શાવે છે.

વ્હાઇટ સ્પેસ કેરેક્ટર	ઉપયોગ/વર્ણન
સ્પેસ (Space)	કીવર્ડ, આઈડેન્ટિફાયર અને ઑપરેટરને અલગ પાડવા માટે ઉપયોગમાં લેવાય છે. (e.g., 'int a = 10;')
ટેબ (Tab - \t)	કોડને વધુ માળખાકીય અને વાંચવામાં સરળ બનાવવા માટે ઉપયોગમાં લેવાય છે.
નવી લાઈન (Newline - \n)	કર્સરને આગળની લાઈન પર ખસેડે છે. સામાન્ય રીતે આઉટપુટમાં ઉપયોગમાં લેવાય છે.
કરેજ રીટર્ન (Carriage Return - \r)	કર્સરને લાઈનની શરૂઆતમાં પરત લાવે છે.
ફોર્મ ફીડ (Form Feed - \f)	પ્રિન્ટિંગ દરમિયાન પછીના પાનાં પર જવા માટે ઉપયોગમાં લેવાય છે (ભાગ્યે જ વપરાય છે).

ટેબલ 6.2 : વ્હાઇટ સ્પેસ

કીવર્ડ્સ (Keywords)

C પ્રોગ્રામિંગમાં કીવર્ડ્સ (મુખ્ય શબ્દો) એ પૂર્વનિર્ધારિત અને સંગ્રહિત શબ્દો (reserved words) છે, જે કમ્પાઇલર માટે વિશેષ અર્થ ધરાવે છે. આ શબ્દોનો ઉપયોગ નિર્ધારિત કાર્ય કરવા અને પ્રોગ્રામના પ્રવાહને નિયંત્રિત કરવા માટે થાય છે. કીવર્ડ્સનો ઉપયોગ ચલ, ફંક્શન અથવા અન્ય આઈડેન્ટિફાયરના નામ તરીકે કરી શકાતો નથી, કારણ કે તે C ભાષાનો એવો ભાગ છે, જે ખાસ અર્થ ધરાવે છે. C ભાષામાં કુલ 32 કીવર્ડ્સ છે. ટેબલ 6.3 માં સામાન્ય રીતે ઉપયોગમાં લેવાતા કેટલાક કીવર્ડ્સ દર્શાવવામાં આવ્યા છે.

કીવર્ડ (Keyword)	અર્થ/ઉપયોગ
int	પૂર્ણાંક ચલ ઘોષિત કરવા માટે. જેમકે, 1, 500, 78
float	અપૂર્ણાંક ચલ ઘોષિત કરવા માટે. જેમકે, 784.5, 6.78
char	કેરેક્ટર ચલ ઘોષિત કરવા માટે
return	ફંક્શનમાંથી મુલ્ય પાછું આપવા
if	શરતી બ્રાન્ચિંગ માટે
else	'if' સાથે વૈકલ્પિક અમલ માટે
while	લૂપ બનાવવા માટે જેનો અમલ શરત સાચી હોય ત્યાં સુધી થયા કરે
for	એવી લૂપ બનાવવા કે જેમાં પ્રારંભ (initialization), શરત (condition) અને વધારો/ઘટાડો (increment/decrement) એક જ લાઈનમાં હોય
void	ફંક્શન કોઈ મુલ્ય પરત આપતું નથી તે દર્શાવવા
break	લૂપ કે switch વિધાનને પૂરું કરવા

ટેબલ 6.3 : કીવર્ડ્સ

આઈડેન્ટિફાયર્સ (Identifiers)

C પ્રોગ્રામિંગમાં, આઈડેન્ટિફાયર્સ (Identifiers) એવા નામો છે જેનો ઉપયોગ ચલ, ફંક્શન, એરે (arrays), સ્ટ્રક્ચર (structure) અને અન્ય યુઝર-નિર્મિત ઘટક (user-defined elements) ને ઓળખવા માટે થાય છે. આઈડેન્ટિફાયર પ્રોગ્રામરને કોડમાં ચોક્કસ ડેટા અથવા પ્રક્રિયાનો સંદર્ભ લેવાની મંજૂરી આપે છે. C માં આઈડેન્ટિફાયર્સને ચોક્કસ નિયમોનું પાલન કરવું પડે છે:

1. આઈડેન્ટિફાયરની શરૂઆત અક્ષર (A-Z અથવા a-z) અથવા અન્ડરસ્કોર (_) થી થવી જોઈએ.
2. પ્રથમ કેરેક્ટર પછી, અંકો (0–9), અક્ષરો અથવા અન્ડરસ્કોરનો ઉપયોગ કરી શકાય છે.
3. કીવર્ડ (Keyword) નો ઉપયોગ આઈડેન્ટિફાયર તરીકે કરી શકાતો નથી.

જેમ કે આપણે અગાઉ ચર્ચા કરી કે, C એ કેસ-સેન્સિટિવ (Case-Sensitive) ભાષા છે, તેથી 'Value' અને 'value' અલગ-અલગ આઈડેન્ટિફાયર્સ ગણાય છે. ટેબલ 6.4 માન્ય અને અમાન્ય આઈડેન્ટિફાયર્સના ઉદાહરણો પૂરા પાડે છે.

આઈડેન્ટિફાયર (Identifier)	માન્ય/અમાન્ય અને કારણ
totalMarks	માન્ય – અક્ષરથી શરું થઈ બધા જ અક્ષરો ધરાવે છે
_value	માન્ય – અન્ડરસ્કોરથી શરું થાય છે
count1	માન્ય – પહેલા અક્ષર પછી અક્ષરો અને અંકો માન્ય છે
1value	અમાન્ય – અંકથી શરુ થાય છે
float	અમાન્ય - 'float' કીવર્ડ છે
user-name	અમાન્ય – હાઈફન (hyphen) માન્ય નથી
Value	માન્ય - 'value' કેસ-સેન્સિટિવ હોવાથી તેના કરતા અલગ છે

ટેબલ 6.4 : આઈડેન્ટિફાયર્સ

ચલો અને અચલો (Variables and Constants)

ચલ (Variable) અને અચલ (Constant) પ્રોગ્રામમાં ડેટા સ્ટોર કરવા માટે વપરાતા મૂળભૂત ઘટકો છે. ચલ એ સ્ટોરેજ લોકેશનને (સંગ્રહ સ્થાનને) આપેલું નામ છે, જ્યાં પ્રોગ્રામના અમલ દરમિયાન ડેટા સંગ્રહિત થાય છે. પ્રોગ્રામના અમલ દરમિયાન ચલની કિંમત બદલાઈ શકે છે. આપણે ચલનો ઉપયોગ કરતા પહેલા તેના ડેટા પ્રકાર (દા.ત., *int, float, char*) સ્પષ્ટ કરીને અને તેને એક નામ આપીને તેને ઘોષિત કરવો જોઈએ. ઉદાહરણ તરીકે:

int age = 25;

જે age નામનો પૂર્ણાંક (integer) ચલ જાહેર કરે છે અને તેને 25 ની કિંમત સાથે આરંભ (initialize) કરે છે. આ કિંમતનો ઉપયોગ આપણા પ્રોગ્રામમાં પછીથી કરી શકાય છે. અચલ એક નિશ્ચિત કિંમત છે, જે એકવાર વ્યાખ્યાયિત થઈ ગયા પછી, પ્રોગ્રામ દ્વારા બદલી શકાતી નથી. અચલ એ સુનિશ્ચિત કરે છે કે મહત્વપૂર્ણ કિંમતો આકસ્મિક રીતે સંશોધિત (modified) ન થાય. અચલને વ્યાખ્યાયિત કરવાની એક સામાન્ય રીત *#define* પ્રી-પ્રોસેસર ડાયરેક્ટીવ અથવા *const* કીવર્ડનો ઉપયોગ કરવાની છે. ઉદાહરણ તરીકે :

#define PI 3.14159

આ ઘોષણા (declaration) ખાતરી કરે છે કે PI ની કિંમત પ્રોગ્રામ દરમિયાન 3.14159 (અચલ) જ રહે છે. ચલ અને અચલ પ્રોગ્રામમાં ડેટાનું અસરકારક રીતે સંચાલન અને ફેરફાર કરવાની મંજૂરી આપે છે.

C પ્રોગ્રામ ઉબુન્ટુમાં ચલાવવાની પ્રક્રિયા (Executing C Program in Ubuntu (Linux))

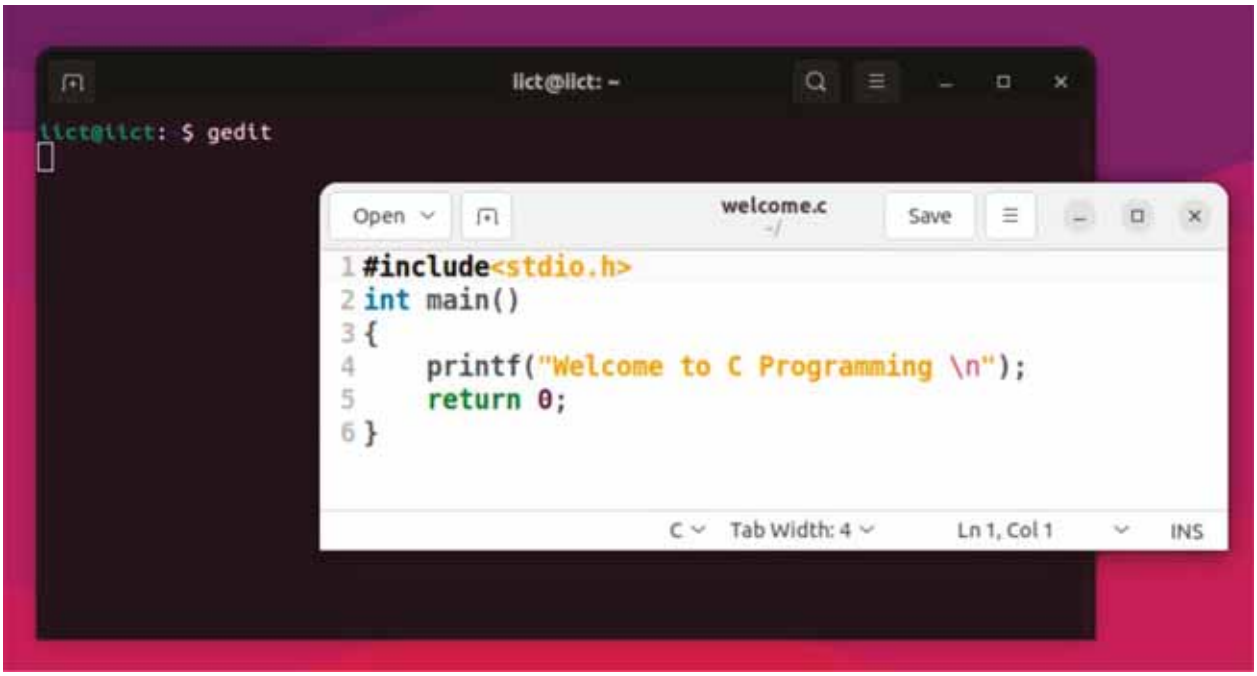
ઉબુન્ટુ ઑપરેટિંગ સિસ્ટમ પર C પ્રોગ્રામ લખવા, કમ્પાઈલ કરવા અને ચલાવવાની પ્રક્રિયાને સમજીએ. આપણે ઉબુન્ટુ OS ટર્મિનલનો ઉપયોગ કરીને C પ્રોગ્રામ લખી અને ચલાવી શકીએ છીએ.

C પ્રોગ્રામ ઉબુન્ટુમાં અમલ કરવા માટેના પગલાં:

1. **પ્રોગ્રામ લખો (Write the program)**

   આકૃતિ 6.2માં દર્શાવ્યા મુજબ gedit જેવા કોઈપણ ટેક્સ્ટ એડિટરને ખોલો અને તેમાં નીચે આપેલ C કોડ લખો:

```
/* C program execution in Ubuntu OS */
#include <stdio.h>
int main()
{
    printf("Welcome to C Programming \n");
    return 0;
}
```



આકૃતિ 6.2 : gedit એડીટરમાં C પ્રોગ્રામ લખવો

2. **પ્રોગ્રામને સંગ્રહો (Save the Program)**

   ફાઈલનું નામ "welcome.c" રાખો. એ સુનિશ્ચિત કરો કે ફાઈલનું ".c" એક્સ્ટેંશન હોય જેથી તે C કોડ ફાઈલ તરીકે ઓળખાય.

3. કમ્પાઈલ કરો (Compile the Code)

C કોડને કમ્પાઈલ કરવા માટે GNU કમ્પાઈલર કલેક્શન (GCC) જરૂરી છે. મોટા ભાગના ઉબુન્ટુ (Ubuntu) વર્ઝન GCC સાથે આવે છે, જેમાં C કમ્પાઈલરનો સમાવેશ થાય છે. તમારું ઉબુન્ટુ ટર્મિનલ ખોલો અને નીચે આપેલા (આકૃતિ 6.3ની લાઈન 2) GCC કમાન્ડનો ઉપયોગ કરીને પ્રોગ્રામને કમ્પાઈલ કરો:

gcc welcome.c -o welcome

આ કમાન્ડ "welcome.c" ને કમ્પાઈલ કરે છે અને "welcome" નામની એક એક્ઝિક્યુટેબલ ફાઈલ (executable file) જનરેટ કરે છે.

આકૃતિ 6.3 : ઉબુન્ટુ ટર્મિનલ પર C પ્રોગ્રામને કમ્પાઈલ અને અમલ કરવો

4. પ્રોગ્રામનો અમલ કરો (Execute the Program)

કમ્પાઈલ કરેલા પ્રોગ્રામને રન કરવા માટે, નીચેનો કમાન્ડ (આકૃતિ 6.3ની લાઈન 3) દાખલ કરો:

./welcome

અમલ થયા બાદ નીચેનો આઉટપુટ ટર્મિનલ પર જોવા મળશે:

Welcome to C Programming

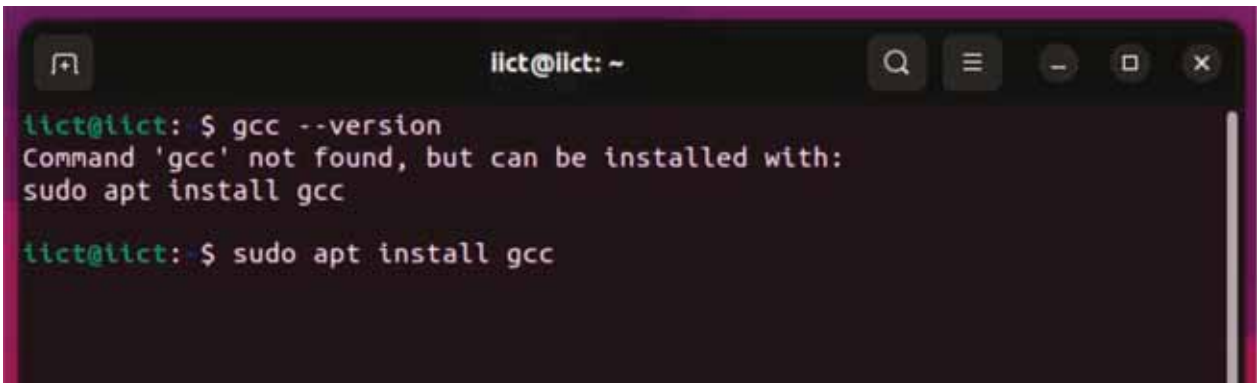
આ પગલાંઓને અનુસરીને, તમે ઉબુન્ટુ ઑપરેટિંગ સિસ્ટમ પર C પ્રોગ્રામને સફળતાપૂર્વક લખી, કમ્પાઈલ અને અમલ કરી શકો છો.

C પ્રોગ્રામને ઉબુન્ટુ (અથવા કોઈપણ Linux - આધારિત) OS પર અમલ કરતી વખતે નીચેના મહત્વપૂર્ણ મુદ્દાઓ ધ્યાનમાં લો:

1. GCC ઈન્સ્ટોલેશન (GCC Installation):

સામાન્ય રીતે ઉબુન્ટુ ઓપરેટિંગ સિસ્ટમમાં GCC ઈન્સ્ટોલ થયેલું જ હોય છે.

- **ઈન્સ્ટોલેશન તપાસવા માટે :** તમારા ઉબુન્ટુ ટર્મિનલ પર **gcc --version** કમાન્ડ ચલાવો, જે આકૃતિ 6.4 માં દર્શાવેલ છે.
- **જો ઈન્સ્ટોલ ન હોય, તો :** GCC ને તમારા કમ્પ્યૂટરમાં ઈન્સ્ટોલ કરવા માટે **sudo apt install gcc** કમાન્ડ ચલાવો, જે આકૃતિ 6.4 માં દર્શાવેલ છે.



આકૃતિ 6.4 : ઉબુન્ટુમાં GCC ઈન્સ્ટોલેશન

2. **ફાઈલનું એક્સ્ટેંશન (File extension) :** તમારા પ્રોગ્રામને .c એક્સ્ટેંશન સાથે સેવ કરો (દા.ત., welcome.c) જેથી કમ્પાઈલર તેને C કોડ તરીકે ઓળખે.
3. **કમ્પાઈલેશનની ભૂલો (Compilation errors) :** જો તમારા કોડમાં સિન્ટેક્સની ભૂલો (syntax errors) હશે, તો GCC ભૂલના સંદેશાઓ (error messages) લાઈન નંબર સાથે દર્શાવશે. તેમને ધ્યાનથી વાંચો અને ભૂલો સુધારો.
4. **કેસ-સેન્સિટિવિટી (Case Sensitivity) :** Linux કેસ-સેન્સિટિવ છે (એટલે કે તે નાના અને મોટા અક્ષરો વચ્ચે તફાવત કરે છે). Welcome.c અને welcome.c જેવા ફાઈલના નામોને અલગ-અલગ ફાઈલો તરીકે ગણવામાં આવે છે.
5. **સારી પ્રથા (Good Practice) :** મૂળ ફાઈલો (જેમ કે welcome.c) અને આઉટપુટ ફાઈલો માટે હંમેશા અર્થપૂર્ણ ફાઈલના નામ (meaningful file names) નો ઉપયોગ કરવો જોઈએ.

સારાંશ

આ પ્રકરણ, C પ્રોગ્રામિંગના મુખ્ય ખ્યાલોનો પરિચય આપે છે, જે તેની ઝડપ, કાર્યક્ષમતા અને પોર્ટેબિલિટી (એક પ્લેટફોર્મ પરથી બીજા પ્લેટફોર્મ પર સરળતાથી લઈ જવાની ક્ષમતા) માટે જાણીતી ભાષા છે. આપણે પ્રોગ્રામિંગમાં સૂચનાઓના મહત્વનો અભ્યાસ કર્યો અને કેવી રીતે C ભાષા માનવ તર્ક (human logic) અને મશીન અમલ (machine execution) વચ્ચે સેતુ તરીકે કાર્ય કરે છે તે પણ જોયું. C પ્રોગ્રામની મૂળભૂત રચનાની ચર્ચા કરવામાં આવી, જેમાં કેરેક્ટર સેટ્સ (character sets), કીવર્ડ્સ (keywords), અને આઈડેન્ટિફાયર્સ (identifiers) જેવા માન્ય C કોડ લખવા માટેના આવશ્યક ઘટકોનો સમાવેશ થાય છે. વધુમાં, આપણે C ના ઈતિહાસ વિશે પણ જાણકારી મેળવી, જે ડેનિસ રિચી દ્વારા વિકસાવવામાં આવી હતી, અને આધુનિક સોફ્ટવેર ડેવલપમેન્ટમાં તેની સતત પ્રાસંગિકતા (relevance) વિશે ચર્ચા કરી. GCC કમ્પાઈલર અને ટેક્સ્ટ એડિટર્સ જેવા મુખ્ય સોફ્ટવેર C પ્રોગ્રામ લખવા અને ચલાવવા કેવી રીતે વપરાય તે પણ જોયું. લીનક્ષ સિસ્ટમ પર C કોડને કેવી રીતે કમ્પાઈલ અને એક્ઝિક્યુટ કરવો તે દર્શાવવા માટે વ્યવહારુ ઉદાહરણો પૂરા પાડવામાં આવ્યા.

સ્વાધ્યાય

1. C પ્રોગ્રામમાં #include ડાયરેક્ટીવનો ઉપયોગ શું છે?
2. C માં main() ફંક્શન શા માટે મહત્વનું છે?
3. C પ્રોગ્રામિંગમાં return 0; સ્ટેટમેન્ટ શું દર્શાવે છે?

4. C પ્રોગ્રામમાં કર્લી બ્રેસિસ {} ની ભૂમિકા શું છે?

5. C પ્રોગ્રામિંગમાં તમે આઉટપુટ કેવી રીતે દર્શાવો છો?

6. C પ્રોગ્રામિંગના ઇતિહાસ વિશે જણાવો.

7. C પ્રોગ્રામિંગમાં કેરેક્ટર સેટ (character set) ની યાદી બનાવો.

8. C પ્રોગ્રામિંગમાં કેસ સેન્સિટિવિટી (Case Sensitivity) વિશે ઉદાહરણ સાથે વિગતવાર સમજાવો.

9. લીનક્ષમાં C પ્રોગ્રામ ચલાવવા માટેના પગલાંની યાદી બનાવો.

10. C પ્રોગ્રામ ફાઇલને સેવ કરવા માટેના પગલાંની યાદી બનાવો.

11. **સાચું કે ખોટું જણાવો.**

(1) C પ્રોગ્રામમાં main() ફંક્શન વૈકલ્પિક છે.

(2) int અને float જેવા કીવર્ડ્સનો ઉપયોગ ચલનામ (variable names) તરીકે કરી શકાય છે.

(3) #include ડાયરેક્ટીવનો ઉપયોગ પ્રોગ્રામમાં લાઇબ્રેરીઓનો સમાવેશ કરવા માટે થાય છે.

(4) C એ કેસ-સેન્સિટિવ (case-sensitive) પ્રોગ્રામિંગ ભાષા નથી.

(5) C માં આઉટપુટ દર્શાવવા માટે printf() નો ઉપયોગ થાય છે.

12. **ખાલી જગ્યાઓ પૂરો.**

(1) C પ્રોગ્રામની શરૂઆત _____ ફંક્શનથી થાય છે.

(2) C પ્રોગ્રામમાં _____ ડાયરેક્ટીવ હેડર ફાઇલનો સમાવેશ કરે છે.

(3) C માં વિધાનનો અંત લાવવા માટે _____ પ્રતીકનો ઉપયોગ થાય છે.

(4) સ્ક્રીન પર લખાણ પ્રિન્ટ કરવા માટે _____ નો ઉપયોગ થાય છે.

(5) C પ્રોગ્રામિંગમાં કીવર્ડ્સનો ઉપયોગ _____ તરીકે કરી શકાતો નથી.

13. **બહુવિકલ્પી પ્રશ્નો. સૌથી યોગ્ય જવાબ પસંદ કરો.**

(1) નીચનામાંથી કયું પ્રતિક C માં હેડર ફાઇલ સામેલ કરવા ઉપયોગી છે?

(a) $ (b) @ (c) # (d) %

(2) printf("Hello \t World!"); નું આઉટપુટ શું હશે?

(a) Hello World? (b) Hello World

(c) Hello World! (d) Error

(3) નીચેનામાંથી ક્યો કીવર્ડ પૂર્ણાંક ચલ ઘોષિત કરવા વપરાય છે?

(a) int (b) float (c) char (d) include

(4) નીચાનામાંથી ક્યો માન્ય આઇડેન્ટિફાયર નથી?

(a) totalMarks (b) 1value (c) _value (d) value1

(5) C માં return 0; વિધાનનો હેતુ શું છે?

(a) પ્રોગ્રામ શરુ કરવો (b) main() ફંક્શનનો અંત કરવો

(c) હેડર સામેલ કરવા (d) આઉટપુટ પ્રિન્ટ કરવા

(6) નીચેનામાંથી કયું ચલના નામ તરીકે માન્ય છે?

(a) 123abc (b) _count (c) float (d) -value

(7) નીચેનામાંથી ક્યો કીવર્ડ અપૂર્ણાંક નંબર ઘોષિત કરવા વપરાય છે?

(a) int (b) float (c) char (d) None

(8) નીચેનામાંથી કઈ એસ્કેપ સીકવન્સ નવી લાઈન માટે વપરાય છે?

(a) \t (b) \n (c) \ (d) \r

(9) નીચેનામાંથો ક્યો કોડ બ્લોક બનાવવા વપરાય છે?

(a) () (b) [] (c) {} (d) <>

(10) નીચેનામાંથી ક્યો C માં કીવર્ડ છે?

(a) loop (b) int (c) print (d) name

પ્રાયોગિક સ્વાધ્યાય

1. printf() ફંક્શનનો ઉપયોગ કરીને તમારું પૂરું નામ પ્રિન્ટ કરતો એક C પ્રોગ્રામ લખો.
2. "Hello, India!" પ્રિન્ટ કરતો એક C પ્રોગ્રામ લખો.
3. નીચે પ્રમાણે મેસેજ પ્રિન્ટ કરતો એક C પ્રોગ્રામ લખો.

Hello,
World!