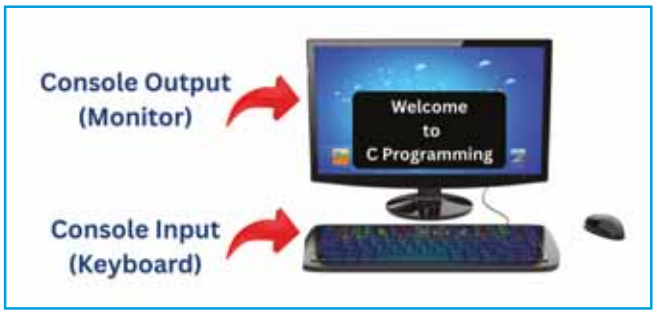
ઈનપુટ આઉટપુટ ઓપરેશન

પરિચય

અગાઉના પ્રકરણમાં, આપણે C પ્રોગ્રામિંગના ઓપરેટરો અને પદાવલીઓના ઉપયોગ વિશે જાણ્યું. આ પ્રકરણ C પ્રોગ્રામિંગમાં ઈનપુટ અને આઉટપુટ (I/O) ઓપરેશનો પર ધ્યાન કેન્દ્રિત કરે છે. C પ્રોગ્રામિંગમાં, ઈનપુટ/આઉટપુટ (I/O) ઓપરેશનો યુઝર અને સિસ્ટમ સાથે સંવાદ કરવા માટે આવશ્યક છે. I/O ઓપરેશનો પ્રોગ્રામને બાહ્ય સ્રોતો (જેમ કે યુઝર ઈનપુટ અથવા ફાઈલ ઈનપુટ) માંથી ડેટા પ્રાપ્ત કરવાની અને પરિણામોને આઉટપુટ સ્ક્રીન પર પ્રદર્શિત કરવાની સવલત આપે છે. C માં પ્રમાણભૂત I/O ફંક્શન, જેમ કે આઉટપુટ માટેનું *printf()* અને ઈનપુટ માટેનું *scanf()*, stdio.h હેડર ફાઈલમાં વ્યાખ્યાયિત થયેલા છે. I/O ઓપરેશનો શીખવા એ C શીખવાનો એક મૂળભૂત તબક્કો છે, કારણ કે તે સંવાદીત પ્રોગ્રામો લખવાનો આધાર પૂરો પાડે છે. આપણે વિવિધ ઈનપુટ ફંક્શનોનો અભ્યાસ કરીશું, જે દરેક યુઝર ઈનપુટને ખાસ પ્રકારે દાખલ કરવા માટે રચાયેલા છે. તે જ રીતે, આપણે આવશ્યક આઉટપુટ ફંક્શનોની ચર્ચા કરીશું જેનો ઉપયોગ પ્રોગ્રામના આઉટપુટને પ્રદર્શિત કરવા માટે થાય છે. આ પ્રકરણ એસ્કેપ સિક્વન્સ (escape sequences) પણ રજૂ કરશે, જે આઉટપુટની વાંચનીયતા વધારે છે. વધુમાં, આપણે ફોર્મેટેડ I/O ફંક્શનો અને ડેટાને વ્યવસ્થિત પ્રદર્શિત કરવા માટે ઉપયોગમાં લેવાતા વિવિધ ફોર્મેટ સ્પેસિફાયર્સ (format specifiers) ના ઉપયોગનો અભ્યાસ કરીશું. આ પ્રકરણના અંત સુધીમાં, તમે ઈન્ટરેક્ટિવ C પ્રોગ્રામ બનાવવા માટે વિવિધ ફંક્શનનો અસરકારક રીતે ઉપયોગ કેવી રીતે કરવો તેની વ્યાપક સમજ મેળવી લેશો.



આકૃતિ 9.1 : કન્સોલ I/O ડિવાઈસીસ

પ્રોગ્રામિંગમાં I/O ઓપરેશન શું છે?

પ્રોગ્રામિંગમાં ઈનપુટ અને આઉટપુટ (I/O) ઓપરેશન એવી પ્રક્રિયાઓ છે જે પ્રોગ્રામને યુઝર પાસેથી ડેટા મેળવવા (ઈનપુટ) અને યુઝરને પરિણામો પ્રિન્ટ કરવાની (આઉટપુટ) સગવડતા પૂરી છે. જુઓ આકૃતિ 9.1.

ઈનપુટ ઓપરેશન (Input Operation) : આમાં કીબોર્ડ, માઉસ, ફાઈલ અથવા નેટવર્ક જેવા ઉપકરણોમાંથી ડેટા મેળવવાનો સમાવેશ થાય છે. ઉદાહરણ તરીકે, કીબોર્ડ દ્વારા એક નંબર દાખલ કરવો.

આઉટપુટ ઓપરેશન (Output Operations) : આમાં સ્ક્રીન, પ્રિન્ટર અથવા ડીસ્ક (ફાઈલ) જેવા ઉપકરણોને ડેટા મોકલવાનો સમાવેશ થાય છે. ઉદાહરણ તરીકે, વિદ્યાર્થીનું પરિણામ સ્ક્રીન પર પ્રિન્ટ કરવું.

C પ્રોગ્રામિંગમાં, I/O ઓપરેશનો સામાન્ય રીતે *stdio.h* (સ્ટાન્ડર્ડ ઈનપુટ/આઉટપુટ) લાઈબ્રેરીમાં વ્યાખ્યાયિત પ્રમાણભૂત ફંક્શનો દ્વારા કરવામાં આવે છે. નીચે લોકપ્રિય I/O ફંક્શનો આપેલા છે:

- scanf(), gets(), getch(), અને fgets() ઈનપુટ માટે
- printf(), puts(), putc(), અને fputs() આઉટપુટ માટે

આ ઓપરેશનો આપણા પ્રોગ્રામને વધુ સંવાદાત્મક (ઈન્ટરેક્ટિવ) અને યુઝર-ફ્રેન્ડલી (યુઝર-અનુકૂળ) બનાવવા માટે આવશ્યક છે. આવા ફંક્શન પ્રોગ્રામના અમલ દરમિયાન યુઝરને સોફ્ટવેર સાથે સંવાદ કરવાની મંજૂરી આપે છે.

Get More Learning Materials Here : CLICK HERE www.studentbro.in

પ્રોગ્રામિંગમાં I/O ઉપકરણોનો ઉપયોગ

I/O (ઇનપુટ/આઉટપુટ) ડિવાઇસ એ હાર્ડવેરના એવા ઘટકો છે જે કમ્પ્યૂટર સિસ્ટમને બાહ્ય વિશ્વ સાથે સંવાદ કરવાની મંજૂરી આપે છે. પ્રોગ્રામ કન્સોલ ઇનપુટ ડિવાઇસમાંથી ઇનપુટ મેળવે છે અને કન્સોલ આઉટપુટ ડિવાઇસ પર આઉટપુટ પ્રદાન કરે છે, જે આકૃતિ 9.1માં દર્શાવેલ છે.

- ઇનપુટ ડિવાઇસનો ઉપયોગ કમ્પ્યૂટર સિસ્ટમમાં ડેટા મોકલવા માટે થાય છે. ઇનપુટ ડિવાઇસના લોકપ્રિય ઉદાહરણો કીબોર્ડ, માઉસ, સ્કેનર અને માઇક્રોફોન છે.
- આઉટપુટ ડિવાઇસનો ઉપયોગ ડેટા પ્રદર્શિત કરવા માટે થાય છે. આઉટપુટ ડિવાઇસના લોકપ્રિય ઉદાહરણો મોનિટર, પ્રિન્ટર, સ્પીકર અને LEDs છે.
- કેટલાક ડિવાઇસીસ, જેમ કે ટચસ્ક્રીન, ઇનપુટ અને આઉટપુટ બંને ડિવાઇસ તરીકે કાર્ય કરે છે.

પ્રોગ્રામિંગમાં, ફાઇલ I/O (File I/O) પણ ખૂબ જ ઉપયોગી છે, જેનો ઉપયોગ હાર્ડ ડિસ્ક જેવા સ્ટોરેજ ડિવાઇસમાંથી વાંચવા અથવા તેના પર લખવા માટે થાય છે. તમે તમારા ઉચ્ચ અભ્યાસમાં ફાઇલ I/O વિશે શીખશો.

ચાલો એક સરળ પ્રોગ્રામથી શરૂઆત કરીએ જે વિદ્યાર્થીનો રોલ નંબર વાંચશે અને પ્રિન્ટ કરશે.

```
/* Program to read and print a roll number of a student. */
#include <stdio.h>
int main() {
    int rollNumber;

    printf("Enter your roll number: ");
    scanf("%d", &rollNumber);

    printf("Your roll number is: %d\n", rollNumber);
    return 0;
}

Result:

Enter your roll number: 101
Your roll number is: 101
```

આ પ્રોગ્રામ C માં પ્રમાણભૂત ઇનપુટ/આઉટપુટ ફંક્શનનો ઉપયોગ કરીને યુઝર પાસેથી ઇનપુટ કેવી રીતે લેવો અને તેને સ્ક્રીન પર કેવી રીતે પ્રિન્ટ કરવો તે દર્શાવે છે. તે યુઝરનો ઇનપુટ સંગ્રહિત કરવા માટે rollNumber નામનો એક પૂર્ણાંક (integer) ચલ ઘોષિત કરીને શરૂ થાય છે. ત્યારબાદ, *printf()* ફંક્શન યુઝરને રોલ નંબર દાખલ કરવા માટેનો સંદેશ પ્રદર્શિત કરે છે. *scanf()* ફંક્શનનો ઉપયોગ પૂર્ણાંક ઇનપુટ વાંચવા અને તેને rollNumber ચલમાં સંગ્રહિત કરવા માટે થાય છે. છેલ્લે, પ્રોગ્રામ યુઝર દ્વારા દાખલ કરવામાં આવેલ મૂલ્ય પહેલા "Your roll number is: " (તમારો રોલ નંબર છે:) સંદેશ પ્રદર્શિત કરવા માટે *printf()* ફંક્શનનો ઉપયોગ કરે છે.

C પ્રોગ્રામિંગમાં <stdio.h> ની ભૂમિકા

<stdio.h> ફાઇલનું પૂરું નામ સ્ટાન્ડર્ડ ઇનપુટ આઉટપુટ હેડર (Standard Input Output Header) છે. તે C કમ્પાઇલરમાં એક સમાવિષ્ટ હેડર ફાઇલ છે જે ઇનપુટ અને આઉટપુટ ઓપરેશન માટે જરૂરી ફંક્શનો પુરા પાડે છે, જેમ કે યુઝર પાસેથી ડેટા વાંચવો અથવા સ્ક્રીન પર આઉટપુટ પ્રદર્શિત કરવો. આપણા પ્રોગ્રામમાં *<stdio.h>* નો સમાવેશ કર્યા વિના, *printf()* અને *scanf()* જેવા ફંક્શનો કમ્પાઇલેશન ત્રુટિ (compilation errors) આપશે. તેમાં વ્યાખ્યાયિત સામાન્ય રીતે ઉપયોગમાં લેવાતા ફંક્શનો નીચે સૂચિબદ્ધ છે :

Get More Learning Materials Here : CLICK HERE www.studentbro.in

- printf() – આઉટપુટ પ્રદર્શિત કરવા
- scanf() – ઈનપુટ લેવા
- getchar(), putchar() – અક્ષર (character) ઈનપુટ/આઉટપુટ માટે
- gets(), fgets(), puts() – સ્ટ્રીંગ (શબ્દો) ઈનપુટ/આઉટપુટ માટે
- fopen(), fclose(), fscanf(), fprintf() – ફાઈલ સંચાલન માટે

ઈનપુટ ફંક્શનો (Input Functions)

C પ્રોગ્રામિંગમાં, ઈનપુટ ફંક્શનનો ઉપયોગ પ્રોગ્રામના અમલ દરમિયાન યુઝર પાસેથી ડેટા મેળવવા માટે થાય છે. *<stdio.h>* હેડર ફાઈલમાં વિવિધ પ્રકારના ઈનપુટ જેમ કે અક્ષરો (characters), સ્ટ્રિંગ્સ (strings) અને સંખ્યાઓ (numbers) વાંચવા માટે અલગ-અલગ ફંક્શનો ઉપલબ્ધ છે. ટેબલ 9.1 સામાન્ય ઈનપુટ ફંક્શનો અને તેનો ઉપયોગ દર્શાવે છે:

ફંક્શન	ઉપયોગ	ઉદાહરણ
`scanf()`	number/char/string વાંચવા	`scanf("%d", &num);`
`getchar()`	એક character વાંચવા	`ch = getchar();`
`gets()`	string (unsafe) વાંચવા	`gets(name);`
`fgets()`	string (safe) વાંચવા	`fgets(name, 50, stdin);`

ટેબલ 9.1 : ઈનપુટ ફંક્શનો

ચાલો લોકપ્રિય ઈનપુટ ફંક્શનોનો ઉપયોગ ઉદાહરણ સાથે સમજીએ.

scanf() ફંક્શન

scanf() ફંક્શન એ સૌથી લોકપ્રિય ઈનપુટ ફંક્શનોમાંનું એક છે, જેનો ઉપયોગ કીબોર્ડમાંથી પૂર્ણાંકો (integers), અપૂર્ણાંકો (floats), અક્ષરો (characters) અને સ્ટ્રિંગ્સ (strings) જેવા ઈનપુટ વાંચવા માટે થાય છે. *scanf()* ફંક્શનની સામાન્ય વાક્યરચના નીચે મુજબ છે:

```
scanf("format_specifier", &variable);
```

જ્યાં,

- **"format_specifier"** (ફોર્મેટ સ્પેસિફાયર) : કયા પ્રકારનો ડેટા વાંચવો છે તે જણાવે છે. જેમ કે, %d પૂર્ણાંક (*int*) માટે, %f અપૂર્ણાંક (*float*) માટે વગેરે.
- **&variable :** એ ચલનું એડ્રેસ છે, જ્યાં વાંચવામાં આવેલ ઈનપુટ ડેટા સંગ્રહ કરવામાં આવશે.

scanf() ના ઉદાહરણો

યુઝર પાસેથી પૂર્ણાંક (integer) સંખ્યા વાંચવા માટે, આપણે નીચે બતાવ્યા પ્રમાણે "%d" ફોર્મેટ સ્પેસિફાયર સાથે *scanf()* ફંક્શનનો ઉપયોગ નીચે મુજબ કરી શકીએ છીએ:

```
int marks;
scanf("%d", &marks);
```

આજ રીતે, યુઝર પાસેથી અપૂર્ણાંક (*float*) સંખ્યા વાંચવા માટે, આપણે *scanf()* ફંક્શન સાથે "%f" ફોર્મેટ સ્પેસિફાયરનો ઉપયોગ નીચે પ્રમાણે કરી શકીએ છીએ:

```
float percentage;
scanf("%f", &percentage);
```

વિદ્યાર્થીનું નામ વાંચવા, આપણે નીચે પ્રમાણે વિધાનો લખી શકીએ:

Get More Learning Materials Here : CLICK HERE www.studentbro.in

```
char name[30];
scanf("%s", name);
```

જુઓ કે *scanf()* માં %s ફોર્મેટ સ્પેસિફાયર એક જ શબ્દ વાંચી શકે છે. *scanf()* ફંક્શન ઈનપુટમાં જ્યાં સુધી પહેલી ખાલી જગ્યા (space) આવે નહીં ત્યાં સુધીના અક્ષરો જ વાંચે છે. આથી, જો આપણે name ચલમાં બે શબ્દો દાખલ કરીએ (જેમ કે "Mudra Patel"), તો તે ફક્ત પહેલો શબ્દ "Mudra" જ સંગ્રહિત કરશે.

વિવિધ ફોર્મેટ સ્પેસિફાયર વિશેની વિગતવાર માહિતી આ અધ્યાયના આગળના ભાગમાં આપવામાં આવી છે.

getchar() ફંક્શન

getchar() ફંક્શનનો ઉપયોગ કીબોર્ડ જેવા સ્ટાન્ડર્ડ ઈનપુટ ડિવાઈસમાંથી એક જ અક્ષર (single character) વાંચવા માટે થાય છે. ઈનપુટ કરેલ અક્ષરને સામાન્ય રીતે આપણે *char* પ્રકારના ચલમાં સંગ્રહિત કરીએ છીએ. જ્યારે યુઝર પાસેથી ચોક્કસ અક્ષર-બાદ-અક્ષર (character-by-character) ઈનપુટ લેવાની જરૂર હોય, ત્યારે getchar() ફંક્શનનો ઉપયોગ કરવો જોઈએ.

getchar() નું ઉદાહરણ

```
char ch;
ch = getchar();     /* This will read one character from keyboard */
```

gets() ફંક્શન

gets() ફંક્શનનો ઉપયોગ ખાલી જગ્યા સહિતની આખી લાઈન (ઘણા શબ્દો ધરાવતી સ્ટ્રિંગ) વાંચવા માટે થાય છે. આ ફંક્શન સ્ટાન્ડર્ડ ઈનપુટ (stdin) માંથી નવી લાઈન (\n) ન મળે ત્યાં સુધી ઈનપુટની લાઈન વાંચે છે. આનો અર્થ એ છે કે, જ્યાં સુધી આપણે કીબોર્ડ પર Enter (નવી લાઈન) દબાવીશું નહીં, ત્યાં સુધી *gets()* ફંક્શન કીબોર્ડમાંથી વાંચવાનું ચાલુ રાખશે. પણ આપણે *gets()* માં વાંચવા માટેના અક્ષરોની સંખ્યા સ્પષ્ટ કરતા નથી, તેથી તે અસુરક્ષિત બને છે કારણ કે તે બફર ઓવરફ્લો (buffer overflow) ની સમસ્યા તરફ દોરી શકે છે. સરળ શબ્દોમાં કહીએ તો, બફર ઓવરફ્લો ત્યારે થાય છે જ્યારે કોઈ પ્રોગ્રામ ચલની મેમરી સ્પેસ (જેને બફર કહેવાય છે) માં તે ખરેખર સમાવી શકે તેના કરતાં વધુ ડેટા સંગ્રહવાનો પ્રયાસ કરે છે.

gets() નું ઉદાહરણ

```
char name[50];
gets(name);          /* This will allow us to enter full name including spaces. */
```

નોંધ : સુરક્ષાના કારણોસર, *gets*() ફંક્શન વાપરવું સલામત નથી. તેના બદલે *fgets()* નો ઉપયોગ કરવો વધુ સુરક્ષિત અને ભલામણપાત્ર છે.

fgets() ફંક્શન

આ ફંક્શનનો ઉપયોગ યુઝર પાસેથી ખાલી જગ્યા સહિતની સ્ટ્રિંગ્સ (ઈનપુટ લાઈન) વાંચવા માટે થાય છે. *fgets()* ફંક્શનમાં, આપણે યુઝર પાસેથી વાંચવા માંગતા હોઈએ તેટલા અક્ષરોની સંખ્યા સ્પષ્ટ કરી શકીએ છીએ, તેથી તેને *gets()* ફંક્શનની તુલનામાં વધુ સુરક્ષિત ફંક્શન ગણવામાં આવે છે. *fgets()* ફંક્શન કાં તો નવી લાઈન (\n) મળે ત્યારે અથવા મર્યાદા (limit) પૂરી થાય ત્યારે વાંચવાનું બંધ કરશે. ફંક્શનમાં બફરનું કદ સ્પષ્ટ કરીને, આપણે બફર ઓવરફ્લો અટકાવી શકીએ છીએ. *fgets()* ફંક્શનની સામાન્ય વાક્યરચના નીચે મુજબ છે:

```
fgets(variable, size, stdin);
```

fgets() નું ઉદાહરણ

```
char name[50];
fgets(name, 50, stdin);
```

આ કોડમાં,

- `char name[50];` વિદ્યાર્થીનું નામ સંગ્રહવા ચલ name ઘોષિત કરે છે.

Get More Learning Materials Here : CLICK HERE www.studentbro.in

- fgets(name, 50, stdin); આ વિધાન સ્ટાન્ડર્ડ ઇનપુટ ડિવાઇસ (કીબોર્ડ) માંથી ખાલી જગ્યા સહિતની આખી લાઇન વાંચશે. તે કુલ 50 અક્ષરો સંગ્રહિત કરી શકે છે. આનો અર્થ છે કે તે 49 જેટલા અક્ષરો ઇનપુટ તરીકે લઈ શકે છે, જેમાં વત્તા એક નલ કેરેક્ટર ('\0') છે, જે સ્ટ્રિંગના અંતને દર્શાવે છે. અહીં "stdin" નો અર્થ સ્ટાન્ડર્ડ ઇનપુટ (કીબોર્ડ) થાય છે.

નોંધ : સુરક્ષાના જોખમોને કારણે *gets*() ફંક્શનને C11 સ્ટાન્ડર્ડમાંથી દૂર કરવામાં આવ્યું છે. સલામત ઇનપુટ મેનેજમેન્ટ માટે *fgets()* નો ઉપયોગ કરો.

મિક્સ ડેટાનું વાંચન (Reading Mixed Data)

આપણે એક અક્ષર વાંચવા માટે *getchar()* અને એક આખી લાઇન વાંચવા માટે *gets()* નો ઉપયોગ કેવી રીતે કરવો તે શીખી લીધું છે. હવે, ચાલો જોઈએ કે એક જ *scanf()* ફંક્શનનો ઉપયોગ કરીને એક જ ઇનપુટ વિધાનમાં *int, char,* અને *string* જેવા વિવિધ ડેટા પ્રકારો કેવી રીતે વાંચી શકાય.

```
/* Program to illustrate mixed data input */
#include <stdio.h>

int main() {
    int age;
    char gender;
    char name[30];

    printf("Enter age, gender (M/F), and name:");
    scanf("%d %c %s", &age, &gender, name);

    printf("\nYou entered:\n");
    printf("Age: %d \n", age);
    printf("Gender: %c \n", gender);
    printf("Name: %s \n", name);

    return 0;
}

Result:

Enter age, gender (M/F), and name:20 F Mudra

You entered:
Age: 20
Gender: F
Name: Mudra
```

આ પ્રોગ્રામમાં, માત્ર એક જ *scanf()* વિધાનનો ઉપયોગ ત્રણ જુદા જુદા યુઝર ઇનપુટ વાંચવા માટે થાય છે. આ *scanf()* વિધાનમાં, %d એક પૂર્ણાંક (age), % c એક અક્ષર (gender), અને %s એક સ્ટ્રિંગ (name, પ્રથમ સ્પેસ પર અટકે છે) વાંચે છે.

એ નોંધવું અગત્યનું છે કે આ પ્રકારના મિક્સ-મોડ ઇનપુટનો ઉપયોગ કરતી વખતે, જો %c નો ઉપયોગ %d પછી તરત જ કરવામાં આવે, તો તે વધેલો ન્યૂલાઇન અક્ષર ('\n') વાંચી શકે છે, અને આપણે પછીના ઇનપુટને યોગ્ય રીતે વાંચી શકીશું નહીં. આ ટાળવા માટે, %c પહેલાં એક સ્પેસનો ઉપયોગ કરવો, જેમ કે નીચેના વિધાનમાં દર્શાવેલ છે:

```
scanf("%d %c", &age, &gender);
```


Get More Learning Materials Here : CLICK HERE www.studentbro.in

આઉટપુટ ફંક્શનો (Output Functions)

C પ્રોગ્રામિંગ કન્સોલ પર આઉટપુટ દર્શાવવા માટે ઘણા ફંક્શનો પૂરાં પાડે છે. આઉટપુટ ફંક્શનોનો મુખ્ય હેતુ પ્રોગ્રામમાંથી ડેટાને પ્રમાણભૂત આઉટપુટ ડિવાઇસ પર મોકલવાનો છે. મોટા ભાગના કિસ્સાઓમાં, આપણે મોનિટરનો ઉપયોગ આપણા પ્રમાણભૂત આઉટપુટ ડિવાઇસ તરીકે કરીએ છીએ. આઉટપુટ ફંક્શનો મુખ્યત્વે <*stdio.h*> હેડર ફાઇલમાં જોવા મળે છે. ચાલો આપણે સૌથી વધુ ઉપયોગમાં લેવાતા આઉટપુટ ફંક્શનોને સરળ ઉદાહરણો સાથે સમજીએ. ટેબલ 9.2 સામાન્ય આઉટપુટ ફંક્શનો અને તેનો ઉપયોગ દર્શાવેલ છે.

ફંક્શન	હેતુ/ઉપયોગ	ઉદાહરણ	નમૂનાનો આઉટપુટ
`printf()`	ફોરમેટ સાથે text, variables, numbers પ્રિન્ટ કરવા	`int age=25;` `printf("Age=%d", age);`	Age = 25
`putchar()`	એક character પ્રિન્ટ કરવો	`putchar('A');`	A
`puts()`	string ને newline સાથે પ્રિન્ટ કરવી	`puts("Hello");`	Hello (ટેક્સ્ટ પછી નવી લાઈન)
`fputs()`	string ને newline વગર પ્રિન્ટ કરવી	`fputs("Hi", stdout);`	Hi

ટેબલ 9.2 : આઉટપુટ ફંક્શનો

ચાલો સામાન્ય આઉટપુટ ફંક્શનના ઉપયોગને ઉદાહરણ સાથે સમજીએ.

printf() ફંક્શન

printf() ફંક્શન C માં ફોર્મેટેડ આઉટપુટ દર્શાવવા માટેનો સૌથી શક્તિશાળી અને લવચીક રસ્તો છે. પ્રોગ્રામમાં *printf()* નો ઉપયોગ કરવા માટે, આપણે #include <stdio.h> વિધાનનો ઉપયોગ કરીને *stdio.h* હેડર ફાઇલને ઉમેરવી પડે છે. *printf()* એ લગભગ દરેક C પ્રોગ્રામમાં સૌથી વધુ ઉપયોગમાં લેવાતું ફંક્શન છે. તે આપણને ટેક્સ્ટ, ચલની કિંમતો, પરિણામો વગેરેને નિશ્ચિત ફોર્મેટમાં પ્રિન્ટ કરવાની સગવડતા આપે છે.

printf() ની વાક્યરચના નીચે પ્રમાણે છે:

```
printf("control string", var1, var2, var3, … varN);
```

જ્યાં, var1, var2, var3 …varN એ ચલ અથવા અચલ અથવા પદાવલી હોઈ શકે છે. આઉટપુટ ફોર્મેટિંગ માટેની સૂચનાઓ કંટ્રોલ સ્ટ્રીંગ (Control String) માં લખવામાં આવે છે. કંટ્રોલ સ્ટ્રીંગમાં નીચેનામાંથી કોઈપણ અથવા બધા ઘટકો સામેલ હોઈ શકે છે:

- મોનિટર પર પ્રદર્શિત થવા માટેનો અક્ષરોનો સમૂહ (string).
- દરેક ચલ માટેનું એક ફોર્મેટ સ્પેસિફાયર (Format Specifier) (જેમ કે %d, %c, %f, વગેરે).
- એસ્કેપ સિક્વન્સ અક્ષરો (Escape Sequence characters) જેમ કે નવી લાઇન (\n), ટેબ (\t), બેકસ્પેસ (\b) વગેરે.

કંટ્રોલ સ્ટ્રિંગ (Control String) ની અંદરના ફોર્મેટ સ્પેસિફાયર (Format Specifiers) ખાલી જગ્યા (space) દ્વારા અલગ પડે છે અને % થી શરૂ થાય છે. *printf()* ફંક્શનનો અનુગામી ભાગ ચલોની સૂચિ ધરાવે છે, જેના મૂલ્યોને આઉટપુટ તરીકે સ્ક્રીન પર પ્રિન્ટ કરવાના હોય છે. આ ચલો કંટ્રોલ સ્ટ્રિંગમાં પ્રદાન કરેલ ફોર્મેટ સ્પેસિફિકેશન સાથે સંખ્યા, ક્રમ અને પ્રકારમાં મેળ ખાતા હોવા જોઈએ, અન્યથા ખોટો અથવા અયોગ્ય આઉટપુટ પ્રદર્શિત થશે.

Get More Learning Materials Here : CLICK HERE www.studentbro.in

printf() ફંક્શન જુદા જુદા પ્રકારના ડેટાને છાપવા માટે %d, %f, %s વગેરે જેવા સામાન્ય ફોર્મેટ સ્પેસિફાયર (Format Specifiers) નો ઉપયોગ કરે છે. આ ફંક્શન ન્યૂલાઈન (newline) આપમેળે ઉમેરતું નથી. આઉટપુટમાં નવી લાઈન લેવા માટે, આપણે *printf()* ફંક્શનમાં \n નો સમાવેશ કરવો પડશે.

printf() ફંક્શનની એક મર્યાદા એ છે કે તેમાં આંતરિક ભૂલ ચકાસણી પદ્ધતિનો અભાવ હોય છે. આનો અર્થ એ છે કે ખોટા ફોર્મેટ સ્પેસિફાયર ઉપયોગ કરવાથી રનટાઈમ સમસ્યાઓ થઈ શકે છે. વધુમાં, *printf()* માત્ર કન્સોલ આઉટપુટ માટે જ કામ કરે છે અને GUI (ગ્રાફિકલ યુઝર ઈન્ટરફેસ) અથવા ફાઈલ આઉટપુટ માટે તેનો ઉપયોગ કરી શકાતો નથી.

printf() નું ઉદાહરણ

```
/* Program to illustrate use of printf() with different format
specifiers. */

#include <stdio.h>
int main() {
    int age = 18;
    float score = 94.50;
    char grade = 'A';
    char name[20] = "Nirva";
    printf("Name:%s, Age:%d, Grade:%c, Score:%.2f", name,
    age, grade, score);
    return 0;
}

Result:

Name:Nirva, Age:18, Grade:A, Score:94.50
```

putchar() ફંક્શન

putchar() એક જ અક્ષરને આઉટપુટ સ્ક્રીન પર પ્રિન્ટ કરે છે. તે અક્ષર-બાદ-અક્ષર (character-by-character) આઉટપુટ માટે સરળ અને કાર્યક્ષમ છે. તે સ્ટ્રિંગ્સ (strings) અથવા એરે (arrays) ને પ્રિન્ટ કરવા માટે લૂપમાં સારી રીતે કામ કરે છે. ઉપરાંત, તેના સરળ માળખાને કારણે, તેને વાંચવું અને સમજવું સરળ છે. નોંધ લો કે માત્ર એક *putchar()* ફંક્શન સ્ટ્રિંગ્સ અથવા એરેને હેન્ડલ કરી શકતું નથી. આને હેન્ડલ કરવા માટે, તમારે તેનો ઉપયોગ અનેક વખત અથવા લૂપની અંદર કરવાની જરૂર પડશે.

putchar() ફંક્શનમાં આઉટપુટ પ્રિન્ટ કરતી વખતે ફોર્મેટિંગ ક્ષમતાઓ મર્યાદિત હોય છે. ફંક્શનમાં ફોર્મેટ સ્પેસિફાયરના અભાવને કારણે, અક્ષર પ્રિન્ટ કર્યા પછી ખાલી જગ્યાઓ (spacing) અથવા ન્યૂલાઈન (newline) ઉમેરવા પર કોઈ નિયંત્રણ રહેતું નથી.

putchar() નું ઉદાહરણ

```
char ch = 'A';

putchar(ch);
```

સ્ક્રીન પર આઉટપુટ A પ્રિન્ટ કરશે.

Get More Learning Materials Here : CLICK HERE www.studentbro.in

puts() ફંક્શન

puts() C માં એક સ્ટાન્ડર્ડ આઉટપુટ ફંક્શન છે, જે *<stdio.h>* હેડર ફાઈલમાં વ્યાખ્યાયિત થયેલ છે. તેનો ઉપયોગ આખી સ્ટ્રિંગને કન્સોલ આઉટપુટ પર પ્રિન્ટ કરવા માટે થાય છે. સ્ટ્રિંગ પ્રિન્ટ કર્યા પછી તે આપમેળે અંતે એક નવી લાઈન ઉમેરે છે. સ્ટ્રિંગ આઉટપુટ માટે તે *printf()* કરતાં વાપરવામાં સરળ છે. પ્રિન્ટ કર્યા પછી તે કર્સરને આપમેળે પછીની લાઈન પર ખસેડે છે.

puts() ફંક્શનના મુખ્ય ફાયદાઓ આ પ્રમાણે છે: તે ફોર્મેટિંગ વગર સ્ટ્રિંગ્સ પ્રિન્ટ કરવા માટે સરળ અને ઝડપી છે. તે આપણને મેન્યુઅલી \n ઉમેરવામાંથી મુક્તિ આપે છે. તે *printf()* ની સરખામણીમાં સરળ છે.

puts() ની કેટલીક મર્યાદાઓ આ પ્રમાણે છે: તેમાં ફોર્મેટિંગ પર કોઈ નિયંત્રણ હોતું નથી. તે સંખ્યાઓ (numbers) અને અક્ષરો (characters) પ્રિન્ટ કરવા માટે યોગ્ય નથી.

puts() નું ઉદાહરણ

```
/* Program to illustrate use of puts() function */

#include <stdio.h>
int main() {
    char name[20] = "Pratham";
    puts("Welcome");
    puts(name);
    return 0;
}

Result:

Welcome
Pratham
```

નોંધ લો કે પ્રથમ *puts()* ફંક્શનમાં \n નો ઉપયોગ કર્યા વિના પણ, વિદ્યાર્થીનું નામ નવી લાઈન પર પ્રિન્ટ થાય છે.

fputs() ફંક્શન

C પ્રોગ્રામિંગમાં *fputs()* ફંક્શનનો ઉપયોગ પ્રોગ્રામના આઉટપુટને કાં તો ફાઈલ પર અથવા સ્ટાન્ડર્ડ આઉટપુટ સ્ટ્રીમ (stdout) પર લખવા માટે થાય છે. તે સ્ટાન્ડર્ડ I/O લાઈબ્રેરી, *<stdio.h>* માં સામેલ છે. *fputs()* વડે, આપણે નક્કી કરી શકીએ છીએ કે આપણે આપણું આઉટપુટ ક્યાં પ્રદર્શિત કરવા માંગીએ છીએ, પછી ભલે તે સ્ક્રીન પર હોય કે ફાઈલની અંદર. તે સ્ટ્રિંગના અંતે નવી લાઈન ઉમેરતું નથી. તે ફોર્મેટિંગ અથવા લાઈન બ્રેક વિના સ્ટ્રિંગ્સ છાપવા માટે ઉપયોગી છે.

fputs() નું ઉદાહરણ

```
    char state[20] = "Gujarat";
    fputs(state, stdout);
```

આ કોડ સ્ક્રીન પર Gujarat પ્રિન્ટ કરશે.

Get More Learning Materials Here : CLICK HERE www.studentbro.in

ધ્યાન રાખો કે *fputs()* નો ઉપયોગ આપણા પ્રોગ્રામના આઉટપુટને ડિસ્ક પર સેવ કરેલી ફાઇલમાં લખવા માટે પણ થઈ શકે છે. જો કે, ફાઇલ મેનેજમેન્ટની કામગીરી અહીં આવરી લેવામાં આવી નથી, કારણ કે તે આ પ્રકરણના કાર્યક્ષેત્રની બહાર છે.

આઉટપુટ ફંક્શનમાં એસ્કેપ સિક્વન્સ (Escape Sequences) નો ઉપયોગ

C પ્રોગ્રામિંગમાં એસ્કેપ સિક્વન્સ (Escape Sequences) એ ખાસ પ્રકારના અક્ષર સંયોજનો છે, જેનો ઉપયોગ *printf()* જેવા આઉટપુટ ફંક્શનમાં આઉટપુટનું ફોર્મેટ નિયંત્રિત કરવા માટે થાય છે. તેનો ઉપયોગ ટેબ (Tab), નવી લાઇન (Newline) અને કેરેજ રિટર્ન (Carriage Return – લાઇનની શરૂઆત પર જવું) જેવા અદૃશ્ય (નોન-પ્રિન્ટેબલ) અક્ષરો દર્શાવવા માટે પણ થાય છે. દરેક એસ્કેપ સિક્વન્સ હંમેશા બેકસ્લેશ (\) થી શરૂ થાય છે અને તેની પાછળ એક અથવા વધુ અક્ષરો આવે છે.

જ્યારે આપણે આઉટપુટને એક નિશ્ચિત માળખામાં (structured text - સ્ટ્રક્ચર્ડ ટેક્સ્ટ) પ્રિન્ટ કરવું હોય, ડેટા ગોઠવવું (align data) હોય અથવા સ્પેશિયલ અક્ષરો (special characters) બતાવવા હોય ત્યારે એસ્કેપ સિક્વન્સ ખૂબ ઉપયોગી સાબિત થાય છે. કલ્પના કરો કે આપણે આપણી સ્ટ્રિંગમાં નવી લાઇન, ટેબ સ્પેસ અથવા ડબલ ક્વોટ (“) પ્રિન્ટ કરવા માંગીએ છીએ. આપણે આ અક્ષરો સીધા ટાઇપ કરી શકતા નથી, કારણ કે C ભાષામાં તેમના ખાસ અર્થ હોય છે. એસ્કેપ સિક્વન્સ આપણને આ અક્ષરોના સામાન્ય અર્થમાંથી “પલાયન” (escape) કરવાની તક આપે છે — એટલે કે, આપણે આ અક્ષરોને શાબ્દિક (literal) રૂપમાં પ્રિન્ટ કરી શકીએ અથવા ચોક્કસ ક્રિયા કરાવી શકીએ. ટેબલ 9.3 સૌથી વધુ ઉપયોગમાં લેવાતા એસ્કેપ સિક્વન્સ અને તેમના અર્થ દર્શાવે છે.

એસ્કેપ સિક્વન્સ	અર્થ
\n	**Newline :** કર્સરને નવી લાઈનની શરૂઆતમાં ખસેડે છે.
\t	**Horizontal Tab :** ટેબ સ્પેસ દાખલ કરે છે.
\b	**Backspace :** કર્સરને એક જગ્યા પાછળ ખસેડે છે.
\r	**Carriage Return (start of line) :** કર્સરને તે જ લાઈનની શરૂઆતમાં ખસેડે છે (જો એ લાઈનમાં અક્ષરો હશે તો તેના પર લખશે).
\\	**Backslash :** બેકસ્લેશ (\) પ્રિન્ટ કરશે.
\"	**Double Quote :** ડબલ ક્વોટ (“) પ્રિન્ટ કરશે.
\'	**Single Quote :** સિંગલ ક્વોટ (‘) પ્રિન્ટ કરશે.
\0	**Null Character :** સ્ટ્રીંગનો અંત દર્શાવે છે, અને *printf()* પ્રિન્ટ નહીં કરે.

ટેબલ 9.3 : એસ્કેપ સિક્વન્સ

ચાલો, એસ્કેપ સિક્વન્સનો ઉપયોગ નીચે આપેલા C પ્રોગ્રામ દ્વારા સમજીએ:

```
/* Program to illustrate use of escape sequence characters */

#include <stdio.h>
int main() {

    printf("Hello, World! \n");            /* Use \n for a new line*/
```


Get More Learning Materials Here : CLICK HERE www.studentbro.in

```
    printf("This is on a new line.\n\n");   /* Two newlines */

    printf("Name\t:\tChirag Patel\n");   /* \t for horizontal tab */
    printf("Age \t:\t30\n");

    /* Use \" to print double quoes */
    printf("Teacher said: \"Welcome Students!\"\n");

    /* Use \\ to print a backslash */
    printf("System path is C:\\Users\\Documents\n");

    /* Use \b for backspace. 3 will no be printed */
    printf("123\b456\n");

    /* Use of carriage return (Line 2 will overwrite Line 1) */
    printf("Line 1.\rLine 2.\n");

    /* Combining multiple escape sequences */
    printf("\n\t--- Program Output ---\n");
    return 0;
}
```

Result:

```
Hello, World!
This is on a new line.

Name    :       Chirag Patel
Age     :       30
Teacher said: "Welcome Students!"
System path is C:\Users\Documents
12456
Line 2.

        --- Program Output ---
```

આગળના કોડમાં, `printf("Line  1.\rLine  2.\n");` વિધાન રસપ્રદ છે. આ પ્રોગ્રામ પ્રથમ "Line 1." પ્રિન્ટ કરે છે. ત્યારબાદ \r (કેરેજ રિટર્ન)ના કારણે કર્સર હાલની લાઈનની શરૂઆતમાં પાછો જાય છે. પછી "Line 2." પ્રિન્ટ થાય છે, જે અગાઉના "Line 1." ને ભૂસીને તેના પર લખે છે.

અંતમાં, \n કર્સરને નવી લાઈન પર લઈ જાય છે, અને પ્રોગ્રામનો બાકીનો ભાગ આગળ ચલાવવામાં આવે છે.

ફોર્મેટેડ ઈનપુટ/આઉટપુટ ફંક્શનો (Formatted I/O Functions)

અગાઉના વિભાગોમાં આપણે C પ્રોગ્રામિંગમાં સામાન્ય રીતે ઉપયોગમાં લેવાતા I/O ફંક્શનોનો અભ્યાસ કર્યો છે. ફોર્મેટેડ ઈનપુટ/આઉટપુટ ફંક્શનો એ એવા ફંક્શનો છે જે આપણને ડેટાને નિર્ધારિત, વાંચવા યોગ્ય ફોર્મેટમાં વાંચવા અને લખવાની સવલત આપે છે.

Get More Learning Materials Here : CLICK HERE www.studentbro.in

printf() અને *scanf()* જેવા ફોર્મેટેડ I/O ફંક્શનોનો ઉપયોગ નિર્ધારિત ફોર્મેટમાં ડેટા દર્શાવવા અને વાંચવા માટે થાય છે. આ ફંક્શનો ફોર્મેટ સ્પેસિફાયર (Format Specifiers) અને કંટ્રોલ સ્ટ્રિંગ (Control String) નો ઉપયોગ કરીને ડેટા કેવી રીતે દર્શાવવો કે સ્વીકારવો તે નિયંત્રિત કરે છે.

ફોર્મેટ સ્પેસિફાયર એ ખાસ પ્રકારની ક્રમબદ્ધ અક્ષર શ્રેણી છે, જે પ્રતિશત ચિહ્ન (%) થી શરૂ થાય છે અને તેની પાછળ એક અથવા વધુ અક્ષરો આવે છે. આ અક્ષરો જણાવે છે કે કયા પ્રકારનો ડેટા વાંચવો કે લખવો છે, તેમજ વૈકલ્પિક રીતે ફોર્મેટિંગ માટેની સૂચનાઓ પણ આપી શકાય છે.

કંટ્રોલ સ્ટ્રિંગ એ I/O ફંક્શનનો પહેલો પેરામીટર હોય છે. તે એક સ્ટ્રિંગ શાબ્દિક (લિટરલ) હોય છે, જેમાં સામાન્ય અક્ષરો (જે એ રીતે જ પ્રિન્ટ/સ્કેન થાય છે) અને એક કે એકથી વધુ ફોર્મેટ સ્પેસિફાયર સામેલ હોય છે.

ફોર્મેટ સ્પેસિફાયર્સ (Format specifiers)ના ચાવીરૂપ પાસાઓ

તેની ચર્ચા નીચે પ્રમાણે છે :

- **ડેટા ટાઇપ સૂચન (Data Type Indication) :** ફોર્મેટ સ્પેસિફાયરનું મુખ્ય કાર્ય I/O ફંક્શનને ઈનપુટ માટેના સંભવિત ડેટા ટાઇપ વિશે જણાવવાનું છે, અથવા આઉટપુટ માટે કયા પ્રકારનો ડેટા પ્રિન્ટ થવાનો છે તે સૂચવવાનું છે. ટેબલ 9.4 માં આપેલા વિવિધ ફોર્મેટ સ્પેસિફાયર્સનો ઉપયોગ કરીને, આપણે I/O ફંક્શનો દ્વારા અલગ-અલગ પ્રકારના ડેટાને અસરકારક રીતે સંભાળી શકીએ છીએ.

ફોર્મેટ સ્પેસિફાયર	ઉપયોગ/વર્ણન
%d	પૂર્ણાંક (decimal - *int*)
%f	અપૂર્ણાંક સંખ્યા (*float*)
%lf	ડબલ- પ્રિસિઝન પૂર્ણાંક (*double*)
%c	એક અક્ષર (*char*)
%s	સ્ટ્રિંગ (અક્ષરોની હરોળ)
%u	ધન પૂર્ણાંક (Unsigned integer)
%x	હેક્ષાડેસીમલ પૂર્ણાંક (lowercase)
%X	હેક્ષાડેસીમલ પૂર્ણાંક (uppercase)
%o	ઓક્ટલ નંબર
%ld	લાંબો પૂર્ણાંક (Long integer)
%lu	ધન લાંબો પૂર્ણાંક (Unsigned long integer)
%p	પોઈન્ટર એડ્રેસ (Pointer address)

ટેબલ 9.4 : ફોર્મેટ સ્પેસિફાયર

- **ફોર્મેટિંગ વિકલ્પો (Formatting Options - Flags, Width, Precision) :** *printf()* અને *scanf()* ની કંટ્રોલ સ્ટ્રિંગમાં ફ્લેગ્સ (flags), પહોળાઈ (width), પ્રિસિઝન (precision) અને લેન્થ મોડિફાયર્સ (length modifiers) સામેલ હોઈ શકે છે, જેનાથી ડેટા કેવી રીતે ઈનપુટ લેવાશે અથવા પ્રિન્ટ થશે તે નિયંત્રિત કરી શકાય છે. નીચે ફોર્મેટ સ્પેસિફાયરનાં ઘટકો કેવી રીતે કાર્ય કરે છે તેની સામાન્ય ફોર્મેટ બતાવવામાં આવી છે :

```
%[flags][width][.precision]specifier
```


Get More Learning Materials Here : CLICK HERE www.studentbro.in

ફોર્મેટ સ્પેસિફાયર્સમાં આઉટપુટના દેખાવને નિયંત્રિત કરવા માટે વૈકલ્પિક ઘટકો (optional components) પણ હોઈ શકે છે:

- **Flags (ફ્લેગ્સ) :** આઉટપુટના વર્તનને બદલવા માટે તેનો ઉપયોગ થાય છે. ઉદાહરણ તરીકે, - લેફ્ટ-જસ્ટિફિકેશન માટે, + હંમેશા સાઈન બતાવવા માટે, 0 ઝીરો-પેડિંગ માટે.
- **Width (પહોળાઈ) :** આઉટપુટ માટે ઓછામાં ઓછી ફિલ્ડ પહોળાઈ નિર્ધારિત કરે છે. જો ડેટા ટૂંકો છે તો પેડિંગ થાય છે; જો લાંબો છે તો ટ્રંકેશન (truncate) નહીં થાય.
- **Precision (પ્રિસિઝન) :** ફ્લોટિંગ-પોઈન્ટ નંબર માટે, ડેસિમલ પોઈન્ટ પછીના અંકોની સંખ્યા નિયંત્રિત કરે છે. સ્ટ્રિંગ માટે, પ્રિન્ટ કરવાં માટે વધુમાં વધુ અક્ષરોની સંખ્યા નિર્ધારિત કરે છે.

- **ઈનપુટ માટે મેચિંગ (Matching for Input) :** જ્યારે *scanf()* જેવા ઈનપુટ ફંક્શન સાથે ઉપયોગ થાય છે, ત્યારે કંટ્રોલ સ્ટ્રિંગમાં ફોર્મેટ સ્પેસિફાયર્સને પેરામીટર લિસ્ટમાં રહેલા સંબંધિત ચલના ડેટા ટાઈપ સાથે ચોક્કસ મેચિંગ રાખવું જરૂરી છે. જો મેચ ન થાય, તો અણધાર્યું વર્તન અથવા ખોટો ડેટા પ્રિન્ટ થઈ શકે છે.

ફોર્મેટેડ (Formated) ઈનપુટ/આઉટપુટ ફંક્શનનું ઉદાહરણ

ચાલો ઉદાહરણ દ્વારા વિવિધ ફોર્મેટિંગ વિકલ્પોને સમજીએ.

```
/* Program to illustrate various formatting options. */

#include <stdio.h>
int main() {
    int num = 123;
    float price = 45.6789;
    char state[20] = "GUJARAT";

    printf("num = %5d \n", num);
    printf("num = %05d \n", num);

    printf("Price = %.3f \n", price);
    printf("Price = %8.2f \n", price);
    printf("Price = %-8.2f \n", price);

    printf("State = %9.3s \n", state);
    printf("State = %-9.3s \n", state);

    return 0;
}

Result:

num =   123
num = 00123
Price = 45.679
Price =    45.68
Price = 45.68
State =       GUJ
State = GUJ
```


Get More Learning Materials Here : CLICK HERE www.studentbro.in

ઉપરના કોડમાં ઉપયોગ કરાયેલા ફોર્મેટિંગ સ્પેસિફાયર્સની વિવિધ અસરનો ટેબલ 9.5માં સારાંશ આપેલો છે.

વિધાન	ફોર્મેટ સ્પેસિફાયર	આઉટપુટ	અસર
printf("%5d \n", num);	%5d	123	num ને 5-અક્ષર પહોળા ક્ષેત્રમાં જમણી બાજુએ પ્રિન્ટ કરશે
printf("%05d \n", num);	%05d	00123	5-અક્ષર પહોળા ક્ષેત્રમાં જમણી બાજુએ 0 પેડીંગ સાથે પ્રિન્ટ કરશે
printf("%.3f \n", price);	%.3f	45.679	3 ડેસીમલ ડીઝીટ સાથે જમણી બાજુએ પ્રિન્ટ કરશે
printf("%8.2f \n", price);	%8.2f	45.68	કુલ પહોળાઈ 8, જેમાં 2 ડેસીમલ ડીઝીટ અને જમણી બાજુએ પ્રિન્ટ કરશે
printf("-8.2f \n", price);	%-8.2f	45.68	કુલ પહોળાઈ 8, જેમાં 2 ડેસીમલ ડીઝીટ અને ડાબી બાજુએ પ્રિન્ટ કરશે
printf("%9.3s \n", state);	%9.3s	GUJ	કુલ 9 અક્ષરની પહોળાઈમાં પહેલાં 3 અક્ષરો જમણી બાજુએ પ્રિન્ટ કરશે
printf("%-9.3s \n", state);	%-9.3s	GUJ	કુલ 9 અક્ષરોની પહોળાઈમાં પહેલાં 3 અક્ષરો ડાબી બાજુએ પ્રિન્ટ કરશે

ટેબલ 9.5 : ફોર્મેટ સ્પેસિફાયર્સ અને તેની અસરનો સારાંશ

સારાંશ

આ પ્રકરણમાં, આપણે C પ્રોગ્રામિંગમાં ઈનપુટ અને આઉટપુટ ઓપરેશનના મૂળભૂત ખ્યાલોનો અભ્યાસ કર્યો. આપણે શીખ્યા કે કેવી રીતે સ્ટાન્ડર્ડ લાઈબ્રેરી <stdio.h> કીબોર્ડમાંથી ઈનપુટ વાંચવા અને સ્ક્રીન પર આઉટપુટ દર્શાવવા સક્ષમ બનાવે છે. મુખ્ય ઈનપુટ ફંક્શનો જેમ કે scanf(), getchar() અને fgets() યુઝરને વિવિધ પ્રકારના ડેટા દાખલ કરવા માટે વપરાય છે, જ્યારે આઉટપુટ ફંક્શનો જેમ કે printf(), putchar(), puts() અને fputs() મૂલ્યો યોગ્ય ફોર્મેટમાં આઉટપુટ દર્શાવવામાં મદદ કરે છે. આપણે %d, %f, %c, %s જેવા ફોર્મેટ સ્પેસિફાયર્સ અને ફીલ્ડ પહોળાઈ, પ્રિસિઝન, અને એલાઈનમેન્ટ જેવા ફોર્મેટિંગ નિયંત્રણોની ભૂમિકા પણ શીખી. પ્રકરણ દરમિયાન આપણે વિવિધ એસ્કેપ સિક્વેન્સ (જેમ કે \n, \t, અને \\) શીખ્યા, જે આપણી ફોર્મેટિંગ જરૂરિયાત મુજબ આઉટપુટ બતાવવા ખૂબ ઉપયોગી છે. પ્રકરણમાં આવરી લેવામાં આવેલા વિવિધ I/O ફંક્શનો અને ઓપરેશનો C પ્રોગ્રામમાં યુઝર સાથે સંવાદ કરવાની માળખાકીય ભૂમિકા રચે છે, જે તેને વાસ્તવિક જગતની એપ્લિકેશન માટે અનિવાર્ય બનાવે છે.

સ્વાધ્યાય

1. પ્રોગ્રામિંગમાં I/O ઓપરેશનનો અર્થ શું છે?
2. C પ્રોગ્રામિંગમાં ફોર્મેટ સ્પેસિફાયર્સનો ઉપયોગ જણાવો. scanf() અને printf() માં ફોર્મેટ સ્પેસિફાયર્સ શું છે? %d, %f, %c, અને %s માટે ઉદાહરણ આપો.
3. કીબોર્ડનો ઉપયોગ કરીને યુઝર પાસેથી ઈનપુટ વાંચવા માટે C માં ઉપયોગમાં લેવાતા કોઈ ત્રણ લોકપ્રિય ફંક્શનો દર્શાવો.

Get More Learning Materials Here : CLICK HERE www.studentbro.in

4. C પ્રોગ્રામિંગમાં એવા ઈનપુટ ફંક્શનનું નામ જણાવો, જે યુઝરને સંપૂર્ણ ટેક્સ્ટ-લાઈન દાખલ કરવાની મંજૂરી આપે, જેમાં શબ્દોની વચ્ચે ખાલી જગ્યા પણ હોય. જ્યારે પૂર્ણ વાક્યો અથવા બહુ-શબ્દ ઈનપુટ સાથે કામ કરતા હોય ત્યારે તે scanf() કરતાં કેમ વધારે યોગ્ય છે તે સમજાવો. જરૂરી હેડર ફાઈલ અને આ ફંક્શન ઉપયોગ કરતી વખતે કોઈ સલામતી બાબતનો ખ્યાલ રાખવો પડશે કે કેમ તે બાબતે જણાવો.

5. આપેલા એસ્કેપ સિક્વેન્સ અક્ષરોનો ઉપયોગ યોગ્ય ઉદાહરણ સાથે જણાવો,: \n, \t અને \b.

6. printf() અને scanf() ફંક્શનનો ઉપયોગ યોગ્ય C પ્રોગ્રામ સાથે સમજાવો.

7. લોકપ્રિય ઈનપુટ અને આઉટપુટ ડિવાઈસના કોઈ ત્રણ નામ યાદીબદ્ધ કરો.

8. C પ્રોગ્રામિંગમાં <stdio.h> ની ભૂમિકા અંગે સંક્ષિપ્ત નોંધ લખો.

9. printf() અને putchar() ફંક્શન વચ્ચેનો તફાવત જણાવો.

10. સંખ્યાઓને ગોઠવવા કે ડેસીમલ પ્રિસિઝન નિયંત્રિત કરવા printf() ના ઉપયોગથી આઉટપુટ કેવી રીતે ફોર્મેટ કરી શકાય? %5d, %2f નો ઉપયોગ દર્શાવતા ઉદાહરણ આપો.

11. **સાચું કે ખોટું જણાવો.**

(1) scanf() ફંક્શનનો ઉપયોગ સ્ક્રીન પર ફોર્મેટેડ આઉટપુટ દર્શાવવા માટે થાય છે.

(2) એસ્કેપ સિક્વન્સ \t નો ઉપયોગ આઉટપુટમાં નવી લાઈન દાખલ કરવા માટે થાય છે.

(3) getchar() ફંક્શનનો ઉપયોગ કીબોર્ડમાંથી એક જ અક્ષરનો ઈનપુટ વાંચવા માટે થાય છે.

(4) gets() ફંક્શનનો ઉપયોગ ખાલી જગ્યા સહિતની સંપૂર્ણ લાઈન યુઝર ઈનપુટ તરીકે વાંચવા માટે કરી શકાય છે.

(5) puts() અને fputs() જેવા ફંક્શનનો ઉપયોગ C માં ઈનપુટ ઓપરેશન માટે થાય છે.

12. **ખાલી જગ્યાઓ પૂરો.**

(1) ફોર્મેટ સ્પેસિફાયર %f નો ઉપયોગ _____ ડેટા ટાઈપ (મૂલ્ય) પ્રિન્ટ કરવા માટે થાય છે.

(2) printf() નો ઉપયોગ કરીને અક્ષરોની સ્ટ્રિંગ પ્રિન્ટ કરવા માટે, ફોર્મેટ સ્પેસિફાયર _____ નો ઉપયોગ કરીએ છીએ.

(3) એસ્કેપ સિક્વન્સ \n નો ઉપયોગ આઉટપુટમાં _____ દાખલ કરવા માટે થાય છે.

(4) ડબલ ક્વોટ ચિહ્ન (") પ્રિન્ટ કરવા માટે, સ્ટ્રિંગની અંદર એસ્કેપ સિક્વન્સ _____ નો ઉપયોગ કરીએ છીએ.

(5) _____ ઈનપુટ ફંક્શન C પ્રોગ્રામિંગમાં કન્સોલમાંથી પૂર્ણ લાઈન (બહુ-શબ્દ) વાંચવા માટે ઉપયોગ કરી શકાય છે.

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

(1) નીચેનામાંથી કયું ફંક્શન એક જ અક્ષર પ્રિન્ટ કરવા માટે વપરાશે?

(a) puts() (b) fputs() (c) printf() (d) putchar()

(2) નીચેનાનું આઉટપુટ શું હશે?

```
printf("Welcome %d", 2025);
```

(a) Welcome (b) Welcome %d

(c) Welcome 2025 (d) Welcome %d, 2025

Get More Learning Materials Here : CLICK HERE www.studentbro.in

(3) નીચેના C વિધાનોનું આઉટપુટ શું હશે?

```
putchar('A');
putchar('\n');
```

(a) Nothing (b) A

(c) 'A\n' (d) A પછી નવી લાઈન

(4) નીચેનામાંથી કયું વિધાન Hello "World" પ્રિન્ટ કરશે?

(a) printf("Hello "World"); (b) printf("Hello \"World\"");

(c) printf('Hello "World"'); (d) printf("Hello World);

(5) C માં કેરેક્ટર ચલને પ્રિન્ટ કરવા માટેનું સાચું ફોર્મેટ સ્પેસિફાયર કયું છે?

(a) %ch (b) %c (c) %s (d) %char

(6) નીચેના વિધાનનું આઉટપુટ ધારો:

```
printf("ABC\bD");
```

(a) ABCD (b) ABD (c) ABC\bD (d) AB CD

(7) એસ્કેપ સિક્વન્સ \t નો હેતુ શું છે?

(a) newline દાખલ કરવાનો (b) space દાખલ કરવાનો

(c) tab space દાખલ કરવાનો (d) અક્ષરને ભૂંસવાનો

(8) નીચેના C પ્રોગ્રામનો આઉટપુટ શું હશે?

```
#include <stdio.h>
int main() {
     printf("C:\\Program Files\\");
    return 0;
}
```

(a) C:\Program Files\ (b) C:Program Files

(c) C://Program Files// (d) Error

(9) નીચેના C પ્રોગ્રામનો આઉટપુટ શું હશે?

```
#include <stdio.h>
int main() {
    char str[ ] = "End\0Start";
    printf("%s", str);
    return 0;
}
```

(a) End (b) End\0Start (c) Start (d) End\0

(10) એસ્કેપ સિક્વેન્સ \0 અર્થ શો છે?

(a) 0 પ્રિન્ટ કરે છે (b) સ્ટ્રીંગનો અંત દર્શાવે છે

(c) ખાલી જગ્યા દાખલ કરે છે (d) કઈ જ પ્રિન્ટ કરતુ નથી

Get More Learning Materials Here : CLICK HERE www.studentbro.in

પ્રાયોગિક સ્વાધ્યાય

1. એક એવો C પ્રોગ્રામ લખો જે કીબોર્ડમાંથી એક અક્ષર વાંચે અને તેને putchar() ફંક્શનનો ઉપયોગ કરીને પ્રિન્ટ કરે.

2. એક એવો C પ્રોગ્રામ લખો જે યુઝરનો રોલ નંબર અને ટકા વાંચે અને પ્રદર્શિત કરે. તમારો પ્રોગ્રામ અમલ કરવા માટે int અને float ચલોનો ઉપયોગ કરો.

3. એક એવો C પ્રોગ્રામ લખો જે વિદ્યાર્થીને તેમનું સંપૂર્ણ સરનામું ટાઇપ કરવા માટે પૂછશે, જેમાં સોસાયટીનું નામ, શહેર, રાજ્ય અને પિનકોડ સામેલ હોય. જરૂરી અલગ ચલો ઘોષિત કરો. છેલ્લે, તે સરનામું સ્ક્રીન પર પ્રિન્ટ કરે, અને વાંચન સુવિધા માટે \t (ટેબ) અને \n (નવી લાઇન) કેરેક્ટરનો ઉપયોગ કરીને તેને અનેક લાઇનોમાં વિભાજિત કરે.

4. એક એવો C પ્રોગ્રામ લખો જે વ્યક્તિનું પ્રથમ નામ અને છેલ્લું નામ બે અલગ scanf() ફંક્શનનો ઉપયોગ કરીને વાંચે, અને પછી એક જ printf() ફંક્શનનો ઉપયોગ કરીને પૂર્ણ નામ પ્રિન્ટ કરે.

5. એક એવો C પ્રોગ્રામ લખો જે વ્યક્તિનું નામ (સ્ટ્રિંગ), ઉંમર (પૂર્ણાંક), ઊંચાઈ (અપૂર્ણાંક), અને શહેર (સ્ટ્રિંગ) માટે ચલ ઘોષણા કરે. આ ચલોને કેટલીક નમૂનાની માહિતી સાથે આરંભ કરો. પછી, printf() નો ઉપયોગ યોગ્ય ફોર્મેટ સ્પેસિફાયર્સ (જેમ કે %s, %d, %.2f) ઉપયોગ કરીને આ માહિતી સુવ્યવસ્થિત અને વાંચનીય રીતે પ્રદર્શિત કરો, ઉદાહરણ તરીકે: "Name: [Name], Age: [Age] years, Height: [Height]m, City: [City]."

6. એક એવો C પ્રોગ્રામ લખો જે યુઝર પાસેથી લગભગ 100 અક્ષરોનો પેરાગ્રાફ વાંચે. યુઝર દ્વારા દાખલ કરેલી સામગ્રી વાંચવા અને પ્રિન્ટ કરવા માટે યોગ્ય ઈનપુટ અને આઉટપુટ ફંક્શનનો ઉપયોગ કરો.

7. એક એવો C પ્રોગ્રામ લખો જે લંબાઈ અને પહોળાઈ ઈનપુટ તરીકે વાંચે. આ લંબચોરસ આકારનું ક્ષેત્રફળ (લંબાઈ × પહોળાઈ) અને પરિઘ (2 × (લંબાઈ + પહોળાઈ)) ગણાવીને પ્રદર્શિત કરો. જવાબ નીચે દર્શાવેલા ફોર્મેટમાં પ્રસ્તુત કરો:

```
Length entered: 10.5 units
Breadth entered: 5.0 units

Calculated Area: 52.50 square units
Calculated Perimeter: 31.00 units
```

8. એક એવો C પ્રોગ્રામ લખો જે યુઝર પાસેથી તાપમાન સેલ્સિયસમાં ઈનપુટ તરીકે વાંચે અને તેને નીચેના સૂત્રનો ઉપયોગ કરીને ફેરનહીટમાં રૂપાંતરિત કરે:

$$F = (C \times 9/5) + 32$$

અહીં

F: ફેરનહીટમાં તાપમાન દર્શાવે છે.

C: સેલ્સિયસમાં તાપમાન દર્શાવે છે.

પરિણામ %.2f નો ઉપયોગ કરીને બે ડેસીમલ સ્થાન સાથે પ્રદર્શિત કરો.

Get More Learning Materials Here : CLICK HERE www.studentbro.in

9. નીચેનો C પ્રોગ્રામ વિદ્યાર્થીનું નામ વાંચે અને પ્રિન્ટ કરે છે

```
#include <stdio.h>
int main() {
     char name[30];
     printf("Enter your name: ");
     scanf("%s", name);

     printf("Your name is: ");
     puts(name);
     return 0;
}
```

પરંતુ જ્યારે આપણે કોડનો અમલ કરીએ અને નામ તરીકે "Jia Patel" દાખલ કરીએ, ત્યારે તે માત્ર પ્રથમ નામ "Jia" પ્રિન્ટ કરે છે. આ કોડથી સંપૂર્ણ નામ પ્રિન્ટ ન કરી શકવાનું કારણ ઓળખો.

10. એવો C પ્રોગ્રામ લખો જે મૂળ રકમ (P), વ્યાજ દર (R) અને વર્ષોની સંખ્યા (N) ના આપેલા મૂલ્યો આધારે સાદુ વ્યાજ ગણે અને પ્રિન્ટ કરે. સાદું વ્યાજ શોધવા માટે નીચેનું સૂત્ર વાપરોઃ

```
SI = P*R*N/100
```


Get More Learning Materials Here : CLICK HERE www.studentbro.in