Programming in C

- Basic input output in C
- Storage Classes
- Types of Errors

Contents

- Input in C
- Printf()
- Output in C
- Scanf()
- Storage Classes in C
- Syntax and Logical Error in C

Input/ output in C

- In C Language input and output function are available as C compiler function or C library provided with each C compiler implementation. These all functions are collectively known as **Standard I/O Library function.**
- Console Input/Output function access the three major files before the execution of a C Program. These are as:
  - **stdin:** This file is used to receive the input (usually is keyborad file, but can also take input from the disk file).
  - **stdout:** This file is used to send or direct the output (usually is a monitor file, but can also send the output to a disk file or any other device).
  - **stderr:** This file is used to display or store error messages.

Getchar and Putchar

A) getchar(): This function is an Input function. It is used for reading a single character from the keyboard. It is buffered function. Buffered functions get the input from the keyboard and store it in the memory buffer temporally until you press the Enter key.

- The general syntax is as:
  v = getchar();
  where v is the variable of character type.

B) putchar(): This function is an output function. It is used to display a single character on the screen.

The general syntax is as:

putchar(v);
where v is the variable of character type.

EXAMPLE

```
#include <stdio.h>
void main( )
{
    int c;
   printf( "Enter a value :");
    c = getchar( );
   printf( "\nYou entered: ");
   putchar( c );
}
```

GETS AND PUTS

```
#include <stdio.h>
void main( )
{
    char str[100];
   printf( "Enter a value :");
   gets( str );
    printf( "\nYou entered: ");
    puts( str );
}
```

GETCH ( )

getch() is a nonstandard function and is present in conio.h header file.

- it reads a single character from keyboard. But it does not use any buffer, so the entered character is immediately returned without waiting for the enter key.

Example:

```
// Example for getch() in C
#include <stdio.h>
#include <conio.h>
void main()
{
  printf("%c", getch());

}
```

C INTEGER INPUT/OUTPUT USING PRINTF AND SCANF

```
#include <stdio.h>
int main()
{
    int num;
   printf("Enter an integer: ");
   scanf("%d",&num);
    printf("Number = %d",num);
   return 0;
   }
```

- **Output**

Enter an integer: 4
Number = 4

C Floats Input/Output

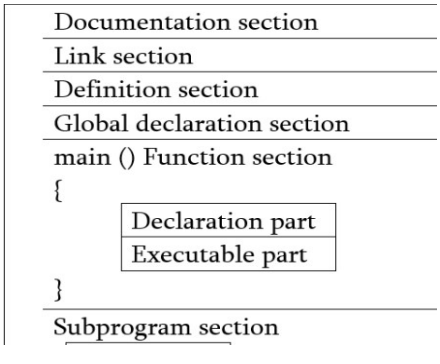
```
#include <stdio.h>
int main()
{
 float f;
printf("Enter a number: "); // %f format string is used in case
   of floats
scanf("%f",&f);
 printf("Value = %f", f);
return 0;
}
```

- **Output**
- Enter a number: 23.45 Value = 23.450000

Format String in C

- You must be wondering what is the purpose of %d inside the scanf() or printf() functions. It is known as **format string** and this informs the scanf() function, what type of input to expect and in printf() it is used to give a heads up to the compiler, **what type of output to expect.**
- Format String Meaning
- %d Scan or print an integer as signed decimal number
- %f Scan or print a floating point number
- %c To scan or print a character
- %s To scan or print a character string

Basic Structure of C program



Storage Class in C

- Every variable in C programming has two properties: **data type and storage class.**
- Type refers to the data type of a variable. And, **storage class determines the scope and lifetime of a variable**.
- There are 4 types of storage class:
  - automatic
  - external
  - static
  - register

The auto Storage Class

- The **auto** storage class is the default storage class for all local variables.
- void main( )
- {
- int mount;
- auto int month;
- }

The register Storage Class

- It is stored in the CPU registers.
- Its scope is limited to the block where it is defined.
- The initial value that it contains(if not already assigned) is any garbage value.
- It remains active till the control is in the block where the variable is declared.
- The keyword used to define a variable with the register storage class is ***'register'***.

```
void main( )
{
register int miles;
}
```

The static Storage Class

- It is stored in the memory.
- Its scope is limited to the block where it is defined.
- The initial value that it contains(if not already assigned) is zero.
- It remains active between different function calls.
- The keyword used to define a variable with the static storage class is ***'static'***.

Example

```
void change( )
{
   auto int a=1;
   static int b=1;
   printf("%d   %d
  \n",a,b);
   a++;
 b++;
}
```

```
void main( )
{
   static int c;
   int i=1;
   printf("%d\n",c);
   while(i<=5)
   {
      change();
      i++;
   }
}
```

THE EXTERN STORAGE CLASS

- The **extern** storage class is used to give a reference of a global variable that is visible to ALL the program files.
- When we use 'extern', the variable cannot be initialized however, it points the variable name at a storage location that has been previously defined.

EXAMPLE

First File: main.c

```
#include <stdio.h>
 int count ;
extern void write_extern();
void main()
{
count = 5;
write_extern();
}
```

Second File: support.c

```
#include <stdio.h>
 extern int count;
void write_extern( )
{
 printf("count is %d\n", count);
}
```

SYNTAX AND LOGICAL ERROR

- **Syntax errors -- Errors in spelling and grammar.**
  - **You can use the compiler or interpreter to uncover syntax errors.**

  Example:
  - **Missing Parenthesis (})**
  - **Printing the value of variable without declaring it**
  - **Missing semicolon**
- **Logical errors -- Errors that indicate the logic used when coding the program failed to solve the problem. Program may execute but results are wrong.**
  - **You do not get error messages with logic errors.**
  - **production of wrong solutions.**

  Example:
  - **sum=a-b;**
  - **for(i=0;i<10;i++);**

ASSIGNMENT

1. Explain various storage classes in C with example.
2. Explain the basic structure of C program in detail.

REFERENCES

- https://www.programiz.com/article/algorithm-programming
- https://www.tutorialspoint.com/cprogramming/c_operators.htm
- LET US C, YASHWANT KANETKAR