

Unit – I
PROMMING IN C

Introduction:- C is a programming language developed by **Dennis Ritche** at AT & T Bell lab in USA in the year 1972.

History of C:

- ALGOL(Algorithm Language) is the base for all modern languages introduced in 1960s.
- In 1963, A new programming language CPL(Combined Programming Language) by Cambridge University, London.
- In 1970, Ken Thompson created a language using BCPL features and called **B Language**.
- In 1972 **Dennis Ritche** introduced C Language by using ALGOL, BCPL and B Languages.
- Later ANSL(American National Standard Institute) approved C in 1983.
- Later C85, C89, C95, C99, Turbo C, Borland C Version are developed.

Structure of C Program

A C Program may contains one or more sections as follows,

Documentation Section
Link Section
Definition Section
Global Declaration Section
Main Section { Declaration Part Section Execution Part Section }
Sub Program Section

Documentation Section:-

Documentation section is used to define “*Comments*”. Comments are non-executable statements they are used to understand the program. In “C” comments are specified by using
“/* */”

Link Section:-

Link section is used to link the program with system library files and header file. Link is establish by using “# include” statement.

Definition Section:-

The definition section is used to define all symbolic constants. It is define by using “# define” Statement.

Global Declaration Section:-

These are some variables that are used in one or more functions. Such variables are called “*Global Variables*” and are declare in global section that is outside of Main().

Main() Section:-

Every C Program must have on Main(). It contains two parts.

- ✚ Declaration Part:- The declaration part is used to declaration variable.
- ✚ Execution Par:- This part contains executable statements.

Sub program section:-

If a C program more than one functions i.e., uses defined functions, are generally declared in this section. These functions defined immediately after main().

Variables

A variable is an entity which holds a value that value may changes during the execution of program.

Declaration of Variables:-

Variables are to be declared before they are going to be used.

Syntax: *datatype variabele name;*

Ex: *int a;*

In the above syntax, data type specifies what type of value to be stored.

We can also declare multiple variables in a single line.

Syntax: *data type Var₁, Var₂, Var₃.....Var_n;* **Ex:** *int a,b,c;*

Initializing of Values:-

It means assigning a value to a variable by using “=” symbol.

Syntax: *datatype variable name=value;*

Ex: *int a=10;*

Syntax: *variable name = value;*

Ex: *a=1;*

Syntax: *datatype Var₁=value, Var₂=value.....;*

Ex: *int a=10, b=20;*

Types of Variables:-

There are two types of variables.

1. Local Variables
2. Global Variables

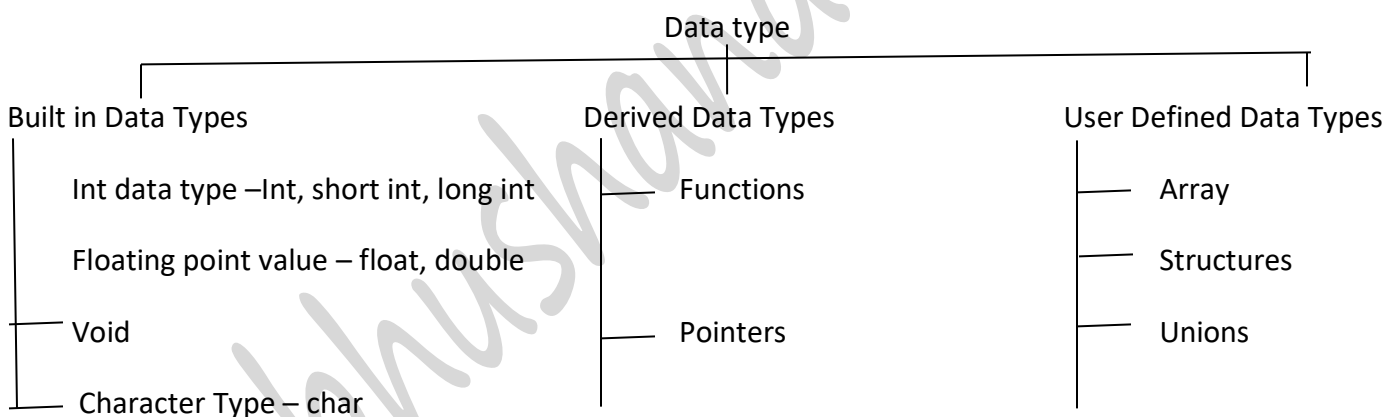
1. **Local Variables:-** Variables that are defined within a function are known as local variables. The scope of these variables is within a function.
2. **Global Variables:-** Variables that are defined outside of the function i.e., outside of Main(). The scope of these variables are everywhere in a program.

Rules:

- A variable name can have alphabets, numbers and underscore.
- The first letter of a variable should be a alphabet or underscore.
- The size of a variable is up to 32 characters.
- No spaces are allowed.
- We should not use a key word as variable name.

Data Type

A Data Type specifies what type of value to be stored or handle by a variable. “C” supports different types of Data types as follows.



Built – in – Data types:

- **Integer DataType:** Integer datatype is used to store whole numbers(0-9). It occupies 2 bytes of memory. Integer data types contains 3 classes namely, Int, short int, long int.

Data Type	Size
Int	2
Short int	1
Long int	4

- **Floating Data Type:** Floating Data type is used to store numbers with decimal values. It occupies 4 bytes of memory. Floating point data types are defined by using a key word “float”. Float point datatypes contains 2 classes namely float and double.

Data Type	Size
Float	4
Double	8

Character Data Type: Character data type is used to store character i.e., a-z or A-Z. It occupies 1 byte of memory. Character data type is defined by using a key word “Char”.

Void: Void data type is introduced in ANSI ‘C’.

- It specifies the return value is nothing.
- It specifies the return value is empty.

Derived Data Types:

- ❖ **Function:** A function is a set of instructions that will perform specific task.

The syntax of a function is

```
return type function – name (list of arguments)
{
    .....
    .....
    Boyd function
    .....
    .....
}
```

In the above syntax, return type specifies that value what type of value that function returns, function name specifies name of the function.

- ❖ **Pointer:** A pointer is a variable which is used to store address of a variable.

```
Ex:   int    a=10;
      Int    *p=&a;
```

User defined DataTypes:

Arrays: An array is collection of similar data values that are store under a single name, each value in the array is identified by using a number that number is called “ Index Number”.

Ex: int a[5];

Structure: A structure is a collection of different data values that are stored under a single name. It is created by using a key word “struct”.

Ex: Struct student

```
{
    Int    a;
    Float  b;
    Char   c;
}
```

Union: A union is definition is a same as structure but union is created by using a key word union. The major difference between a structure and union is in terms of memory.

```
Ex:   Union student
      {
          Int    a;
          Float  b;
          Char   c;
      }
```

Operators

An operator is a symbol, which is used to perform mathematical & logical operations on operand.

c=a+b

c,a,b – operand ; =,+ -- operators

“C” supports different types of operators.

1. Arithmetic Operators
2. Relational Operators
3. Logical Operators
4. Assignment Operators
5. Increment / Decrement Operators
6. Conditional Operators
7. Bitwise Operators and
8. 8. Special Operators

1. **Arithmetic Operators:** Arithmetic operators are used to perform basic mathematical operations like +, -, *, /, %. Etc.

Operator	Meaning	Example
+	Addition	A+b
-	Subtraction	a-b
*	Multiplication	A*b
/	Division	a/b
%	Modulo	a/b

2. **Relational Operators:** Relational operators are used to make comparison between two quantities.
Ex: Comparing age of two persons.

Operator	Meaning	Example
<	Less than	A	Greater than	a>b
<=	Less than or equals	A<=b
>=	Greater than or equals	a>=b
!=	Not equal	A!=b

3. **Logical operators:** Logical operators are used to test two or more conditions at a time.

Operator	Meaning	Example
&&	Logical AND	(a<b)&&(a<c)
	Logical OR	(a<b) !(a<c)
!	Logical NOT	!(a<b)

The working of logical operators are well understood by truth table.

OP1	OP2	OP1&&OP2	OP1 OP2
1	1	1	1
1	0	0	1
0	1	0	1
0	0	0	0

4. **Assignment Operators:** Assignment operator is used to assign a value to variable. It is denoted by “=”.

Ex: a=10;

5. **Increment / Decrement Operators:** Increment operator is used to increase the values of a variable by 1. It is denoted by “++”.

Ex: a++, ++a

Decrement operator is used to decrease the value of a variable by 1. It is denoted by “--”.

Ex: a--, --a

6. **Conditional Operators:** Conditional operator is also called as ternary operator. Because it contains three parts. i.e., a condition, a true block and a false block.

Syntax:
$$\begin{array}{c} \text{True} \\ \text{exp1} \text{ ? exp2 : exp3} \\ \text{False} \end{array}$$

7. **Bitwise Operators:** Bitwise operators are used to perform operations on bits i.e., 0's and 1's.
Bitwise operators are

Operators	Meaning
&	Bitwise AND
	Bitwise OR
^	Bitwise XOR
<<	Left shift
>>	Right shift

8. **Special Operators:**

❖ **Comma Operators:** It is used to separate variable in declaration statement.

Ex: int a,b,c;

- ❖ **Size of Operators:** It is used to find size of a variable or constant. Ex: `int a;`
`int b = size of (a);`

Compiling and Execution of Program

Popular programming languages like C, C++, Java etc., use compiler to convert high level language instructions into low level language instructions. Compiler is a program that converts high level language instruction to low level language instructions. Generally, compiler performs two things first it verifies the program errors, if errors are found, it returns list of errors otherwise it converts the complete code into low level language.

We use the following steps to create and execute C Program on Windows OS

- Step 1 : Create Source Code
 Write program in the editor and save it with “.c” Extension
- Step 2 : Compile Source Code
 Press Alt+F9 to compile
- Step 3 : Run Executable code
 Press Ctrl+F9 to run
- Step 4 : Check Result
 Press Alt+F5 to open user screen

Step 1: Creating Source Code

To create source code, we use any text editor to write the program instructions. The following steps are used to create source code file in Windows O.S.

- Click start button
- Select Run
- Type CMD and press enter
- Type `cd c:\TC\bin` in the command prompt and press enter
- Type TC Press enter
- Click on file → new in C Editor window
- Type the program
- Save it as filename.c

Step 2: Compiling Source Code

Compilation is the process of converting high level language instructions into low level language instructions. We use the shortcut key Alt+F9 to compile a C Program in Turbo C.

Whenever we press Alt+F9, the source file is going to be submitted to the compiler on receiving a source file, the compiler first check for the errors. If there are any errors, then compiler returns the list of errors, if there are no errors then the source code is converted into the object code and store it as file with obj extension. Then the object code is given to the linker. The linker combines both the object code and specified header file code and generates an executable file with exe extension.

Step 3: Executing / running executable file

After completing compilation successfully, an executable file is created with .exe extension. We use a shortcut key Ctrl+F9 to run a C Program. Whenever we press Ctrl+F9, the exe file is submitted to the CPU on receiving exe files; CPU performs the task according to the instructions written in the file.

Step 4: Check result

After running the program the result is placed into user screen. Just we need to open the user screen to check the result of the program execution. We use the shortcut key Alt+F5 to open the user screen and check the result.

Input / Output Statements

These are some library functions which are available for transferring data between computer and input & output devices.

Some of the input and output function are as following

- a. `getchar` & `putchar` functions:

The `getchar()` is used to give input a single character.

Syntax: `ch=getchar()`

The `putchar()` is used to give output a single character.

Syntax: `putchar(ch);`

- b. `getch` & `putch` functions:

getch() is used to give input a single character and it doesn't require an enter key to be pressed.

Syntax: ch=getch();

The putchar() is used to give output a single character & it doesn't require an enter key to be pressed.

Syntax: putchar(ch);

c. gets & puts function:

The gets() is used to give input a string. String is sequence of characters.

Syntax: gets(ch);

The puts() is used to give output a string.

Syntax: puts(ch);

d. Printf & Scanf functions:

The printf() is used for displaying output on the screen i.e., the data is transfer from computer to output device.

Syntax: printf("format string", arg1,arg2,arg3.....);

Ex: ("%d", a);

The scanf() is used for accepting input from the user i.e., the data is transfer from input to the computer.

Syntax:scanf("format string", &arg1, & arg2, &arg3,.....);

Ex: ("%d",&a);

Write a C Program to print your name?

```
#include <stdio.h>
```

```
void main()
```

```
{
```

```
    printf("srnb");
```

```
}
```

Write a C Program to print any 3 names?

```
#include <stdio.h>
```

```
void main()
```

```
{
```

```
    printf("vivekananda");
```

```
    printf("ramakrishna paramahansa");
```

```
    printf("sharadha devi");
```

```
}
```

Write a C Program to accept a value and print that value?

```
#include <stdio.h>
```

```
void main()
```

```
{
```

```
    int a;
```

```
    a=10;
```

```
    printf("%d",a);
```

```
}
```

Write a C Program to accept 2 values and print that values?

```
#include <stdio.h>
```

```
voidmain()
```

```
{
```

```
    int a,b;
```

```
    a=10;
```

```
    b=20;
```

```
    printf("%d",a);
```

```
    printf("%d",b);
```

```
}
```

Write a c program to perform sum of 2 numbers?

```
#include <stdio.h>
void main()
{
    int a,b,c;
    a=5,b=10;
    c=a+b;
    printf("%d",c);
}
```

Write a C program to find subtraction of 2 numbers?

```
#include <stdio.h>
void main()
{
    int a,b,c;
    a=25,b=20;
    c=a-b;
    printf("%d",c);
}
```

Write a C Program to find multiplication of 2 values?

```
#include <stdio.h>
void main()
{
    int a,b,c;
    a=25,b=20;
    c=a*b;
    printf("%d",c);
}
```

Write a C Program to accept value by using scanf function and print that value?

```
#include<stdio.h>
void main()
{
    int a;
    scanf("%d",&a);
    printf("%d",a);
}
```

Write a C program to accept two values by using scanf and print that values?

```
#include<stdio.h>
void main()
{
    int a,b;
    scanf("%d",&a);
    scanf("%d",&b);
    printf("%d,%d",a,b);
}
```

or

```
#include<stdio.h>
void main()
{
    int a,b;
    printf("enter a value");
    scanf("%d",&a);
    printf("enter b value");
    scanf("%d",&b);
    printf("%d,%d",a,b);
}
```

Write a C program to find area of a circle?

```
#include<stdio.h>
void main()
{
    int area,r,pi=3.14;
    printf("enter r value");
    scanf("%d",&r);
    area=pi*r*r;
    printf("%d",area);
}
```

Write a C Program to find circumference of circle

```
#include<stdio.h>
void main()
{
    int circumference;
    float pi=3.14;
    printf("enter r value");
    scanf("%d",&r);
    circumference=2*pi*r;
    printf("%d",circumference);
}
```

write a c program to calculate simple interest

```
#include<stdio.h>
void main()
{
    float si,p,t,r;
    printf("enter p value");
    scanf("%f",&p);
    printf("enter t value");
    scanf("%t",&t);
    printf("enter r value");
    scanf("%f",&r);
    si=(p*t*r)/100;
    printf("%f", si);
}
```

Write a C Program to swap two values without using 3rd value?

```
#include<stdio.h>
void main()
{
    int a,b;
    printf("enter a value");
    scanf("%d",&a);
    printf("enter b value");
    scanf("%d",&b);
    printf("before swap");
    printf("%d,%d",a,b);
    a=a+b;
    b=a-b;
    a=a-b;
    printf("after swap");
    printf("%d,%d",a,b);
}
```


Write a C Program to swap 2 values with using 3rd variable?

```
#include<stdio.h>
void main()
{
    int a,b,c;
    printf("enter a value");
    scanf("%d",&a);
    printf("enter b value");
    scanf("%d",&b);
    printf("before swap");
    printf("%d,%d",a,b);
    c=a;
    a=b;
    b=c;
    printf("after swap");
    printf("%d,%d",a,b);
}
```

Type Casting and Type Conversion

Type Casting is a way to convert a value of a variable from one datatype to another datatype. For example, if we want to store a float value in to integer value then we use *type casting*.

There are 2 types of Type Casting.

They are 1. Implicit Conversion 2. Explicit conversion

1. **Implicit Conversion**: In implicit conversion, compiler automatically converts a value of a variable from one data type to another data type.

Ex: #include<stdio.h>
void main()
{
 int a=10;
 float b;
 b=a//implicit conversion
 printf("%d",b);
}

2. **Explicit Conversion**: In explicit conversion, the user will converts a value of variable from one data type to another data type. Explicit conversion is done by using a "casting()" operator.

Ex: #include<stdio.h>
void main()
{
 int a=10;
 float b;
 b=(float); a//explicit conversion
 printf("%f",b);
}

The major difference between Type casting and Type conversion are as follows.

Type Casting	Type Conversion
1. In this user will convert a value of a variable from one data type to another data type.	1. In this compiler will automatically converts a value of a variable form one data type to anther data type.
2. We use a casting()	2. We no need to use any operator.
3. It is done during program execution.	3. It is done during program compilation.
4. It is also called as Narrow Casting.	4. It is also called as widening casting.

Flow Chart


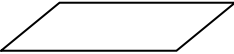



A flow chart is a diagrammatic or graphical representation of algorithm, work flow as process. It illustrates or explains the sequence of steps that must be performed to solve problem.

The first structured method for documenting process flow, the *flow process chart* was introduced by *Frank and Gilbreth*.

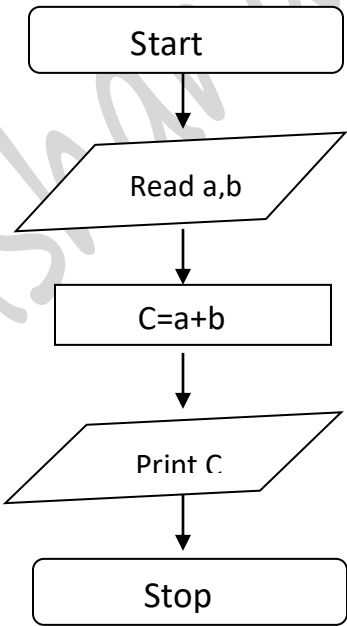
These are four general types of flow charts. They are

- Document flowchart
- Data flowchart
- System flowchart
- Program flowchart

The commonly used symbols for drawing a flow chart are as follows.

Symbol	Name	Function
	Terminal or oval	An oval represents a start point or end point
	I / O symbol or Parallelogram	It represents input or output
	Process or rectangle	A rectangle represents logic or formula
	Decision or Diamond	It represents a decision or condition
	Connector or Circle	It represents to joint two or more flow charts

Let us draw an example flow chart for sum of 2 numbers.



Advantages:

- ✚ Flow charts are better way of communication of logic of a system.
- ✚ It helps effective analysis of a problem.
- ✚ By using charts we can easily develop programs.
- ✚ It helps for debugging process.

Disadvantages:

- ✚ a correction or change in logic requires to redrawing of flow chart.
- ✚ It is very difficult to represent complex logic.
- ✚ It is time consuming.

Algorithm & its features

Definition: An algorithm is a finite sequence of well defined steps for solving a problem.

Or

Algorithm is a step by step process of a program execution.

The word algorithm is derived from algorism; it means *process of doing arithmetic with Arabic numbers*. The word comes from the name of famous Arab mathematician *Al-Khowarizmi* who developed methods for solving problems which used step by step instructions. The word *algorism* was combined with *arithmetic* to become *algorithm*.

Characteristics or features of Algorithm:

An algorithm has 5 important characteristics. They are

1. Input 2. Output 3. Definition 4. Finiteness 5. Effectiveness

1. **Input:** An algorithm may specify the quantities which are to be read.
2. **Output:** An algorithm may specify the quantities which are to be write or display.
3. **Definiteness:** Each instruction must be clear and unambiguous.
4. **Finiteness:** An algorithm should be terminated or stop after finite number of steps.
5. **Effectiveness:** arithmetic must be effective which means that all operations are executable.

Let us write an algorithm to find sum of 2 numbers.

Step 1: start

Step 2: Read a,b values

Step 3: compute $c=a+b$

Step 4: print "c"

Step 5: Stop