Mathematical Method Lab-II

Program in C- Language

In this lecture note I am trying to teach my students what is the basic structure of a program in C. What are the preliminary important knowledge required to write a program code in C.

Dr. Saumen Charaborty

Introduction:

C is a high level, powerful programming language. It is developed by Dennis Ritchie at Bell Laboratory, USA (1970).
Basic Features of C-language:

1. Machine independence- Program written in C normally run on many different computers without alteration.
2. Structured programming- Program are written in systematic structured programming concept.
3. Flexibility-C language has flexible features to write program for numerical, commercial and graphical applications.
4. Integrity-This refers to the accuracy of calculation.
5. Efficiency- Efficiency is measured on the basis of execution speed and memory utilization. C is quite efficient in this regard.

Computer program is a set of precise sequential instruction to process data and find a useful output. These instructions are formed using certain symbols and words according to some rigid rules known as syntax rules (or grammar). C-language has its own syntax rules (i.e. grammar and vocabulary). We will study those rules after knowing the basic structure of program which is followed by any programmer of C-language.

Basic Program structure:

Structure	Description	Example (program on area calculation)
Documentation Section →	A set of comment lines giving the name of the program, the author, date etc	**/*Calculation of area of a rectangle/SaumenC/29.03.2020*/**
Link Section →	Provides instructions to complier to link functions from the system library.	**#include<stdio.h>** **#include<math.h>** **#include<conio.h>**
Definition Section →	Defines all symbolic constants	**#define f(l,b)<space>l*b**(optional)
Global declaration Section →	Global variables (that are used more than one functions) are declared	
main() →	It is the special function used by C, to tell computer where the program starts. **Note: Every program must have exactly one main function.**	
{ →	Marks the beginning of the program	
Declaration part →	Declare all the variables that are used in the executable part of the main function. (may called local variable)	**int**<space> **l, b, A;** (here l means length, b means breadth A means area)
Input →	The values of some declared variables are assigned.	**l=7; b=5;** **scanf("%d%d",&l,&b);**
Executable part →	Data processing occur as instructed	**A=l*b;** **A=f(l,b);**
Output →	The value/output after processing of said variable is stated to show.	**printf("%d%d%d\n,l,b,A");**
} →	Marks the ending of the program.	
Subprogram →	It contains all user defined functions that are used in main function. These are generally placed just after the main function.	

Link Section i.e. Library Function:

C-programs are divided into two function modules.

1. Some functions are stored in the C-library. These are called Library function. Library functions are grouped category wise and are stored in different files known as header files. To acess them we have to write **#include<file name.h>;** .h refers to header file extension.
   Some example of header files : stdio.h- standard input output (it contains **scanf, printf** etc.)
   math.h- mathmatical operation (it contains **sqrt, arctan** etc.)
   conio.h- console input output (it contains **getch, clrscr** etc.)
   Note: These are called preprocessor directives. These are placed at the beginning of a program.
2. Some functions are defined by the users, begin with **#define**. **"#define"** instruction defines value to a symbolic constant for use in the program. Whenever a symbolic name is encounter, the compiler substitutes the value associated with the name automatically. The defined function remains same throughout the execution of the program.

Global declaration Section and Local declaration Section:

It comes at the beginning of the program and they are visible to all parts of the program. But in declaration section within the main (), describes the variable that are used within the main only. These types of variables are called local variables. Every program or subprogram may have local variables.

Some preliminary points regarding declaration of variables-

1. In C all variables must be declared to tell the compiler what the variable name is? and what type of data it holds?
   For example if we write, **int** a=5; it means 'a' is integer type variable and 5 value is assigned to it. Here, **int** is called keyword.
   In this regard it is to be mention that data are of following types,
   a. Integer(**int**) [e.g. 2, 5 etc]
   b. Character(**char**) [e.g. X, Hello etc]
   c. Floating point(**float**) [2 bit, e.g. 2.63, 5.14 etc]
   d. Double precision floating point (**double**) [64 bit]
   e. void (no data)
2. In assigning variables we have to take care of following points-
   a. It must begin with letter.
   b. Length of variable should not be greater than eight characters.
      If we write average_height and average_weight, both mean same to computer. Such name then may be written as avg_height and avg_weight. (they may other options also).
   c. Uppercase and lowercase are significant. For example: TOTAL, Total, total all are different from each other.
   d. Take care that it should not be the keyword of C.
   e. Blank space is not allowed within the variable name.

Input(section):

There are two ways to put the values of the variables (data input).

i) In first way the values of variables are directly assigned. As for example, **int a=5;**, But in such cases we have to change the values of variables every time as per need entering into the program.

ii) In second way of giving data to the variables through keyboard (no need of entering into the program) using the **scanf** function. It is a general input function available in C. The general format of **scanf** is as follows,

scanf("control string",&variable1,&variabl2,….);

the control string contains the format of data being received. The ampersand symbol before variable name specifies the address of variable. We consider the program of area calculation of a rectangle where length (l) and breadth (b) are input as follows,

scanf("%d%d",&l,&b);

Here **%d** value we specifies that integer value is to be read from terminal (or indirect mean is that variables are integer type). For floating will write **%f.**

Output(section):

To get the output value after processing at the terminal, we use the **printf** function in C. The general format of **printf** is as follows,

printf("control string",variable1,variabl2,….);

For example if we consider the program of area calculation of a rectangle where the calculated area (A) is to be printed including length (l), breadth (b), the format will be as follows,

printf("%d%d%d",l,b,A);

Note: Backslash character constant: C supports some special bacslash character constants. That are used in the input or output functions. For example **\n -**stands for newline character.

Process Section:

When data will be input, the computer has to process it. Computer is instructed to execute the process (perform some mathematical or logical manipulations) by different operators and expressions. Now we will go through some primary important operators and expressions.

1. Arithmetic Operators: We use + for addition, - for subtraction, * for multiplication, / for division and % for modulo division. Here it is to be noted that results will be obtained depending upon the variables types (as declared). If the result is integer type then it always truncates the fractional part.
   Example: 15/10=1 (integer type); 15.0/10.0=1.5 (float type);
   The modulo division operation produces the remainder of an integer division.
   Example: 14%3=2; -14%3=-2;-14%-3=-2;14%-3=2
2. Relation Operators: <(less than), >(greater than),<=(less or equal),>=(greater or equal), ==(equal), !=(not equal).
3. Logical Operators: &&(AND), ||(OR), !(NOT) are used when more than one conditions are used and make decision.
   Example:a>b&&x==10
4. Assignment Operators: =(equal) sign is used to assign values to variable.
5. Increment and decrement Operators: ++ for increment and -- for decrement, They have the following form
   ++m or m++ and –m orm--
   Note: ++m(prefix)means the variable is incremented by unit first and then expression is evaluated.
   m++(postfix)means the variable is incremented by unit after the evaluation of the expression.
6. Conditional Operators: "? :"
   Example: x=(a>b)?a:b; means if a>b then x=a otherwise x=b.
7. Coma Operator: The coma (,) operator can be used to lin the related expressions together.
   Example: value=(x=10,y=5,x+y)
   For loop, for(n=1,m=10,n<=m,n++)
8. Arithmetic Expression: Arithmetic expressions are combination of variables, operators and constants arranged as per syntax of language. e.g. aXb-c is written as a*b-c. Expressions are evaluated using an assignment statement form i.e. variable=expression. For example, x= a*b-c. In arithmetic expression without parentheses will be evaluated from left to right using the priority rule as follows, high priority *,/,% and low priority +.-.

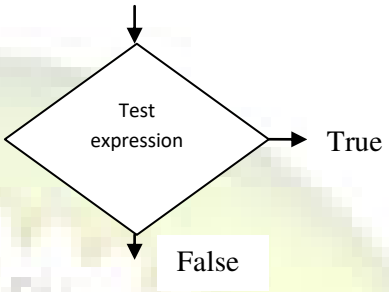
Example: x=9-12/3+3*2-1
=9-4+3*2-1
=9-4+6-1
=5+6-1
=11-1
=10

However, the order of evaluation can be changed by introducing the parentheses.

Conditional Structure with IF statement in C

1. Simple IF statement:

```
if(test expression)
{
 Statement block (if true)
 }
 Statement-x (if false)
```



2. Simple IF…..ELSE statement:

```
if(test expression)
{
True block statement
}
else
{
False block statement
}
Statement-x
```

(There are nested IF statements and ELSE IF ladder structure but are not explained here)

Loop Structure:

Loop program has two segments, one known as the **body of the loop** and the other known as the **control statement**.

While statement:

This is an entry control loop statement. It means if the control statement is true then it enters into the loop.

```
while(test condition)
{
Body of the loop
}
```

Do-While statement:

This is exit controlled loop statement.

```
do
{
Body of the loop
}
while(test condition)
```

For loop statement:

It is an entry control loop statement.

```
for(initialization;test condition;increment)
{
Body of the loop
}
```

Note: More than one variable can be initialized at a time in for statement.
Increment section may also have more than one part.
Test condition may have any compound relation.

Examples of some C-Program

1. **Program code that is use to convert the temperature from Celsius scale to Fahrenheit scale.**

```
/* conversion of temperature from Celsius scale to Fahrenheit scale, Satyaprakash,30.03.2020*/
#include<stdio.h>
#include<conio.h>
main()
{
float C,F;
prinf("Enter Celsius scale value ");
scanf("%f",&C);
F=1.8*C+32;
printf("The temperature in Fahrenheit scale is %f",F);
getch();
}
```

2. **Program code that is use to find the largest number among three given number.**

```
/*Determination of the largest number among three given number, Kahakshan, 30.03.2020*/
#include<stdio.h>
#include<math.h>
#include<conio.h>
main()
{
int n1,n2,n3,max;
prinf("Enter three numbers ");
scanf("%d%d%d",&n1, &n2, &n3);
max=n1;
if(n2>max)
{
max=n2;
}
if(n3>max)
{
max=n3;
```

```
}
printf("The largest number is %d",max);
getch();
}
```

3. **Program code that is use to calculate the factorial of a given number.**

```
/*Determination of factorial of a given number, Abhishek, 30.03.2020*/
#include<stdio.h>
#include<math.h>
#include<conio.h>
main()
{
int n,C,F;
prinf("Enter the number whose factorial is to be determined ");
scanf("%d",&n);
C=1;
F=1;
for(C=1,C<=n,C++)
{
F=F*C;
}
printf("The factorial of %d is %d",n,F);
getch();
}
```