Introduction to C-Programming

By
Utpal Saikia
Assistant Professor
Department of Mathematics
Silapathar College

Contents

- Basic programming concept
- Programming approach to solving problem
- Algorithm
- Flowcharts
- Character set
- C tokens, keywords and identifiers, constants, variables
- Data types

Problem solving technique

A problem can be defined as something difficult to deal with or understand. It is something difficult to solve. The following sequence of steps can be performed for problem solving technique

1) Problem definition

(i) First problem is studied thoroughly.

(ii) Next, emphasize is given on ‘what to do‘rather than ‘how to do‘.

(iii) The requirements necessary for solving the problem is find out.

(iv) The constraints if any available in the problem is pointed out.

2) Find a method/formula

(i) Here we looked for a already solved problem, which in general case does not exist but if it exists then it is guaranteed for correctness.

(ii)Then search is made for a similar solved problem .

(iii) If available help is taken for the already solved problem.

(iv) Follow a top down approach .

(v) Bottom up approach.

(vi) Attack is made on the problem from all possible sides.

3) Select the ‘best’ method/formula

After that we select the best method among all if available for solving the problem. Best in the sense of time, cost and labour that is the selected method should minimize the above entities than the other methods available.

4)Prepare a work plan for the method

After selecting the best method an algorithm or flow chart is developed

5)The algorithm is then tested for the known data and if the test results are not correct then process is repeated from beginning.

6) If successful computer code is prepared for the algorithm/flow chart

7)A test run is made on the programme for a set of known data and if the results are not correct the process is repeated from beginning that is step 1

8)If the results are correct in the test run, we get the solution for the actual data.

Algorithm

An algorithm is a finite set of instruction that follow to accomplish a particular task. An algorithm is composed of a finite set of steps, each of which may require one or more operation. It can be defined as a step by step instructions of explicit and unambiguous set of finite statements which when executed for a set of given initial values will produce a definite output in finite time .

The word algorithm comes from the name of a Persian author "Abu Jafar Mohammad Ibu musa Alkhowarizmi", in 825 A.D.

Characteristics of Algorithm

(1) **Input/output :** An algorithm must have some inputs(initial values) and must produce certain outputs. The output must be inserted relation to the input.

(2) **Definiteness:** Every word of an algorithm must be precisely defined.

(3) **Finiteness :** Number of statements as well as time taken in an algorithm must be finite.
(4) **Completeness**: An algorithm must be complete in the sense that it must solve all the problems of the particular type for which the algorithm is designed .for eg- algorithm for quadratic eqn.

Q) Write an algorithm to compute the roots of a quadratic equation .
Sol :-
Step 1: Input a, b, c
Step 2: If a=0 then go to step 12 otherwise continue
Step 3: r= -b/(2*a)
Step 4: d = b*b – 4*a*c
Step 5: If (d<0) go to step 10 otherwise continue
Step 6: If (d>0) go to step 14 otherwise continue
Step 7: Print ‘equal roots’
Step 8: print roots are : r, r
Step 9: Stop
Step 10: Print ‘roots are complex’
Step 11: stop
Step 12: print ‘equation cannot be solved’
Step 13: stop
Step 14: print ‘unequal roots’
Step 15: r1 = sqrt(d)/(2*a)
Step 16: print ‘root 1= r+r1’,’root 2 =r –r1’
Step 17: stop

Flow charts

A flow chart is a pictorial or diagrammatic representation of the steps involve in solving a problem. It also shows the logical sequence in which the steps are to be performed. The first formal flow chart was made by John Von Neumann in 1945. A flow chart is consist of blocks called symbols and arrows called flow fine. There are two types of flow charts:-

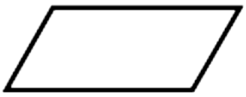
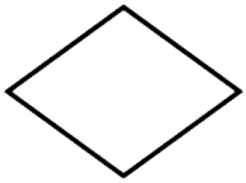
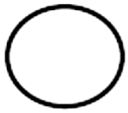
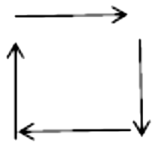
1. System flow chart.
2. Programme flow chart.

System flow chart

A system flow chart shows the path through which the data is passed from one processing unit to another within a company.

Programming flow chart

This flow chart is used to draw the diagram to show how the programme instructions to work the basic symbols used for programme flow charts are:-

Basic Symbol	Function / Purpose
1. 	1. Start / Stop.
2. 	2. Input / Output.
3. 	3. Process / Calculation.
4. 	4. Decision.
5. 	5. Connector.
6. 	6. Flow lines.

1. Start / Stop

The shape of start and stop block is flat oval. It is used at the beginning and the end of a flow chart. Generally, the word **START** and **END/STOP** are given in this block.

2. Input / Output

The shape of input or output is parallelogram. Generally, the words **INPUT**, **READ**, **PRINT** are given in this block. This symbol represents instructions given to an input or output device.

3. Processing

The shape of this processing block is a rectangle. It is used to show an action. All of the processing functions such as addition, subtraction, multiplication, division and assignment, etc. are represented in this block and the words like for, go-to are also used here. It denotes action to be taken by the C.P.U. in the actual processing of data.

4. Decision

The shape of this decision block is diamond. It is used to show a decision to be taken between two paths of action. There is one line going into a decision block and two going out. This is because the block always asks the question and gives two possible path to be taken depending on whether the answer to the question is YES or NO.

5. Connector

The shape of the connector block is a circle. It is used when a diagram has to be continued on another page or when drawing a line between two points on the same page or when drawing a line between two points on the same page would unnecessarily complicated the drawing.

6. Flow lines

The shape of the symbol is a straight line. It is used to link symbol and to indicate the sequence of operation. Flow lines may be cross, from junctions or connect any two symbols so that the symbol flow a logical meaningful flow. Occasionally broken arrow are used to specify a possible flow change in the computer during the programme run depending upon the condition in the process.

C

C is a programming language developed at AT & T's Bell Laboratories of USA in 1972. It was designed and written by a man named Dennis Ritchie. C is popular because because it is reliable, simple and easy to use. Moreover, in an industry where newer languages, tools and technologies emerge and vanish day in and day out, a language that has survived for more than 3 decades has to be really good.

Three features of a Good programming language

1) How it stores data.
2)The way through which input / output handles .
3)Number of operators.

Computer Memory

Bit – 0 or 1
Byte – 8 bits
KB – 2^{10} bytes

MB - 2^{10} KB

GB - 2^{10} MB

TB -2^{10} GB

If in a byte there are n bit then its storing capacity is –

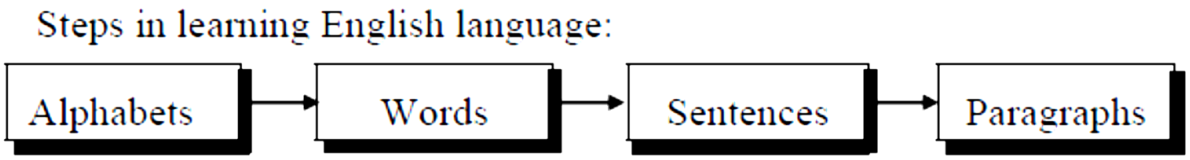
$(2^{n+1}+1)$ to + $(2^{n-1}-1)$

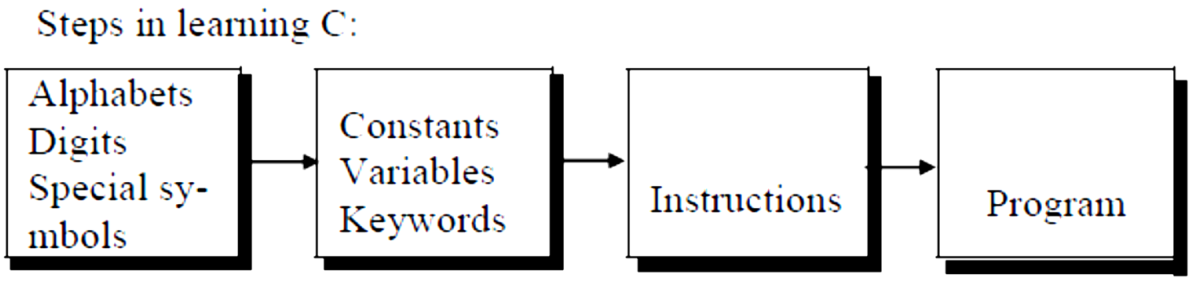
Sign Bit

1 if the number is negative and 0 if the number is positive .

Programming language structure

HLL→Compile→Producing object programme→Executed→Result





The C Character Set

A character denotes any alphabet, digit or special symbol used to represent information. The valid alphabets, numbers and special symbols allowed in C are –

Alphabets	A. B. ..... Y. Z a. b. ..... y. z
Digits	0. 1. 2. 3. 4. 5. 6. 7. 8. 9
Special symbols	~ ‘ ! @ # % ^ & * ( ) _ - + = | \ { } [ ] : ; " ' < > , . ? /

C Tokens

The smallest individual units in C are known as C tokens. C has six types of tokens. They are: - Keywords (float, while), Identifiers (main, amount), Constants (100,-15), Strings (utpal), Special symbols ({},[]) and operators(+ - * /)

Keywords

Keywords are certain reserved words that have standard, predefined meanings in C. These keywords can be used only for their intended purpose , they cannot be used as variables or function names.The standard keywords are –

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

Some compilers may also include some of the following keywords:-

ada asm entry far fortran huge near pascal etc

Constants

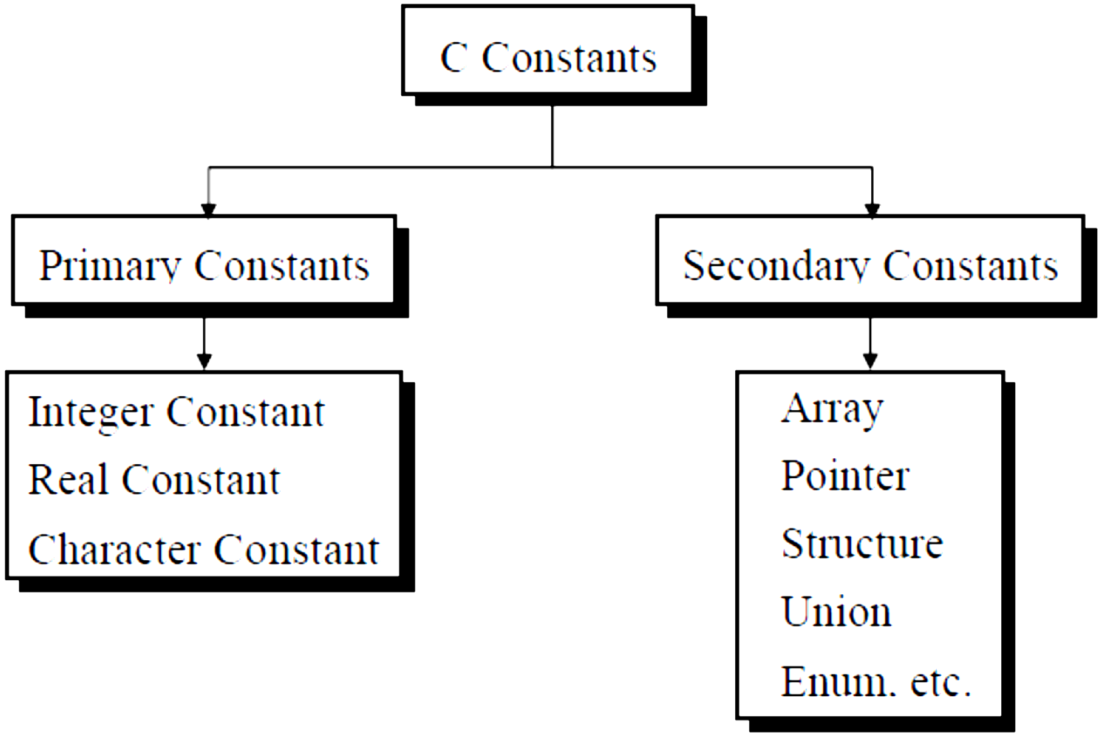
Any unchanged value in a program during the program execution is called constant.

C constants can be divided into two major categories:

(i)Primary Constants

(ii)Secondary Constants

These constants are further categorized as shown in Figure



** But at this stage we would restrict our discussion to only Primary Constants, namely, Integer, Real and Character constants.

1) **Integer Constant:** An integer constant is a signed or unsigned whole number.

Rules for Constructing Integer Constants

(a)An integer constant must have at least one digit.
(b)It must not have a decimal point.
(c)It can be either positive or negative.
(d)If no sign precedes an integer constant it is assumed to be positive.
(e)No commas or blanks are allowed within an integer constant.
(f)The allowable range for integer constants is -32768 to 32767 for 16 bit compiler like turbo C.

Ex.: 426
+782
-8000
-7605

2) **Real or Floating point constant:** Any signed or unsigned number with fractional part is called real or floating point constant. A real constant can be written in decimal or exponential form.

Following rules must be observed while constructing real constants expressed in decimal form:

(a)A real constant must have at least one digit.
(b)It must have a decimal point.
(c)It could be either positive or negative.
(d)Default sign is positive.
(e)No commas or blanks are allowed within a real constant.

Ex.: +325.34
426.0
-32.76
-48.5792

In exponential form of representation, the real constant is represented in two parts. The part appearing before 'e' is called mantissa, whereas the part following 'e' is called exponent.

Following rules must be observed while constructing real constants expressed in exponential form:

(a)The mantissa part and the exponential part should be separated by a letter e.
(b)The mantissa part may have a positive or negative sign.
(c)Default sign of mantissa part is positive.
(d)The exponent must have at least one digit, which must be a positive or negative integer. Default sign is positive.
(e)Range of real constants expressed in exponential form is -3.4e38 to 3.4e38.
Ex.: +3.2e-5
4.1e8
-0.2e+3
-3.2e-5

3) **String or character constant:** Any string of characters enclosed in apostrophes or quotes is called string constant or character constant. There are two types of string constants

(i) **Single Character string constant**: Any letter or character enclosed in single apostrophe is called single character string constant.
Eg – 'y', '$' , '=' etc

(ii) **String of character constant:** Any string of characters consisting of letters , digits and symbols enclosed in double quotesis called string of character constants.
Eg – "Total value is-" , "utpal-505" etc

Variables

A variable is a name or an identifier which is used to refer a value and this value varies or changes during the program execution. Variable names are names given to locations in memory. These locations can contain integer, real or character constants. In any language, the types of variables that it can support depend on the types of constants that it can handle. This is because a particular type of variable can hold only the same type of constant. For example, an integer variable can hold only an integer constant, a real variable can hold only a real constant and a character variable can hold only a character constant.

Rules for Constructing Variable Names

(a) A variable name is any combination of 1 to 31 alphabets, digits or underscores. Some compilers allow variable names whose length could be up to 247 characters. Still, it would be safer to stick to the rule of 31 characters. Do not create unnecessarily long variable names as it adds to your typing effort.

(b) The first character in the variable name must be an alphabet or underscore.

(c) No commas or blanks are allowed within a variable name.

(d) No special symbol other than an underscore (as in **gross_sal**) can be used in a variable name.

(e) Reserved words cannot be used as variables.

(f) All variables used in a C program are declared with appropriate data types before the variable is assigned any value.

Ex.: si_int
m_anjan
pop_e_89

Data type

There are four basic data types in C language and they are—

Data type	Description	Value range	Value range in decimal	Memory
char	single character	-2^7 to 2^7-1	-128 to 127	1 byte
int	integer constant	-2^{15} to $2^{15}-1$	-32768 to 32767	2 bytes
float	floating point number	-2^{31} to $2^{31}-1$	3.4e-38 to 3.4e+38	4 bytes
double	double precision floating point number	-2^{63} to $2^{63}-1$	1.7e-308 to 1.7e+308	8 bytes

Declaration

A declaration associates a group of variables with a specific data type.

For eg : int a,b;

float d,e;

Expression

An expression represents a single data item, such as a number or a character. The expression may consist of a single entity such as a constant, variable, etc or may also consist of some combination of such entities, interconnected by one or more operators. For eg : a+b, x+y, c=a+b, ++e etc.

Statements

A statement causes the computer to carry out some action. There are three different classes of statements in C. They are :

(a) Expression Statements.
(b) Compound Statements.
(c) Control Statements.

An Expression Statement consists of an expression followed by a semicolon.

For eg : a+3; c=a+b; ++i; printf ("Hi"); etc

N.B:- ; is called null statement.

A compound statement consists of several individual statements enclosed within a pair of braces { }.

For eg-

```
{
        Pi=3.14;
        Area=pi*r*r;
}
```

Control statements are used to create special program features , such as logical tests, loops and branches. For eg-

```
while(count<n)
{
    printf("x=");
    scanf("%d",&x);
    sum+=x;
    ++count;
}
```