Problem Solving and Programming

Nishant Gupta

This reference book can be useful for

BBA, MBA, B.Com, BMS, M.Com, BCA, MCA

and many more courses for Various Universities

NEERAJ
PUBLICATIONS
www.neerajbooks.com

Published by:

NEERAJ PUBLICATIONS

(Publishers of Educational Books)

Sales Office : 1507, 1st Floor,
Nai Sarak, Delhi-110 006
E-mail: info@neerajbooks.com
Website: www.neerajbooks.com

Typesetting by: Competent Computers

CONTENTS

PROBLEM SOLVING AND PROGRAMMING THROUGH C

Problem Solving

1

1.1 INTRODUCTION AND OBJECTIVES

If we think about our daily life, apart from computer, we encounter many problems and also solve them, some of them have a proper solution already defined, for same we have to think ourself and find one. The basic path to problem solving remains that first identify the goal, secondly make use of available resources and finaly use them to reach the goal. But in this chapter we use INS concept of problem solving pertaining to computer programming.

When we talk of problem solving pertaining to computer programming, we take into consideration, the output required as goal and input data available as resource and other parametres as time contraints etc.

There is no standard or universal approach to solve a problem. Same problem can be solved in different ways, we connot say which one is correct. Both ways would we correct if they produce correct output. The problem solving skills are integral part of computer programming. No matter what language we are using.

The approach to problem solving comes with experience and practice. Problem solving skills are equally important in project life-cycles like study, designing, development, testing and implementation.

The computer problem solving requires:

(i) Analysis of problem.
(ii) Identify inputs/outputs.
(iii) Plan a process with inputs available.

Problem solving requires personal creativity, and analytical ability, which comes by practice and exposure.

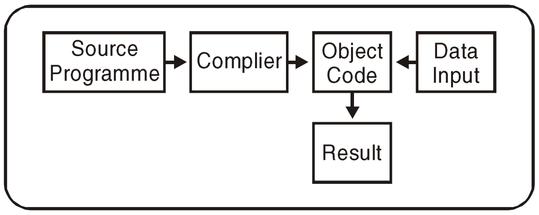
Computer is used to solve the problems. This machine follows the set as stored instructions called program very quickly and have the memory to store data. The computer cannot think, it only executes program instructions very quickly.

Computers do not work like humans. When computer is used to solve a problem, we must specify the input required, the process to be done and the output required. Once these are considered an algorithm and a program is developed. It is not necessary the programme developed gives the correct result. If it does not give the correct result we need to find the errors and then correct them, this process is called Debugging.

All types of computer programs are reffered to as softwere. Physical parts of equipment such as electronic parts, devices, come under hardware. Software works on the hardware and uses it to have the output. Sometimes operations performed by the software are built in hardware.

Set of instructions of the high level language used to code a problem and to find a solution is referred to as source programme. A software is used to read that code and translate into machine language which computer understands is used, which is known as compiler or interpreter.

The following is the path to from making a program and execute it.



1.2 PROBLEM SOSLVING TECHNIQUES

Problem solving comes by experience, it is a creative process which defines the systematic process. The number of steps can be taken to solve the problem is the performance of the algorithm.

1.2.1 Steps for problem solving

***(i)* Problem Definition:** The problem should be defined and the clear concept should be developed that what must be done, rather than how is it done.

***(ii)* Start analysing problem:** There are many ways to solve a problem. What comes in your mind may be correct but the approach should be of best way.

***(iii)* Use specific examples:** We can take the already solved problems to the new one which save times and efforts.

***(iv)* Similarities among problems :** It is important to see if there are any similarities between the current problem and the past problem which we have solved. The more experience a person have in solving problems.

***(v)* Working backwards from the Solution:** Sometimes we already have the solution to the problem and then try to work backwards to the starting point. Even a guess at the solution to the may be enough.

1.2.2 Using computer as a problem solving tool.

The computer is used to solve problem only when the problem's solution is made in step by step manner using algorithm.

follow the steps

(i) Develop Algorithm and a flow chart of the sloved problem.
(ii) write a computer program using programming language.
(iii) Enter the program using editor.
(iv) Compile the program
(v) Test and Debug the program
(vi) Run and Get the Results.

1.3 DESIGN OF ALGORITHMS

The first step in program developement is to describe what you want the computer to do. This plan expressed as a sequence of jobs is called algorithm. An algorithm of a programming language becomes program.

Definition: An algorithm is a set of well-defined rules for the solution of a problem in a finite number of steps. An algorithm can be expressed in pseudo codes–something resembling the programming language.

Features of Algorithm:

***(i)* Proper understanding of the problem:** The problem should be properly understood in order to devlop an effective algorithm.

***(ii)* Use of procedures and Subroutines:** When the problem becomes complex the algorithm written should be in modular manner *i.e.* one main problem to be solved using many small problem.

***(iii)* Choice of variables:** Proper variables names should be used in algorithm so that the name of variable gives the meaning to use it.

***(iv)* Documentation of program:** If the algorithm is complex then it should have comments at appropriate places so that the other person can understand the code.

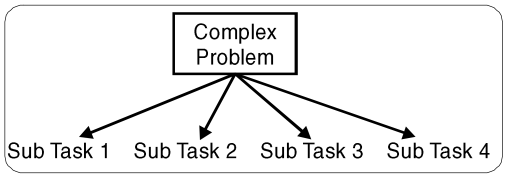
Some examples of algorithm.

1. Algorithm to calculate factorial of a number.

```
1. Start
2. Input numbr n
3. Initialize  $i = 1$, $f = 1$
4. Repeat steps 4 – 6 until $i = n$
5. $f = f * i$
6. $i = i + 1$
7. print f
8. Stop
```

1.3.4 Top Down Design

A technique for algorithm design that tries to accomodate human limitation is known as top down design. Top down design provieds the way of handling the logical complexity and detail encountered in computer algorithms. In this approach the complex problem is divided into many simple sub problems and each of them is worked upon and then the solution is combined.



1.4 ANALYSIS OF ALGORITHM EFFICENCY

Algorithms use some of the computers resources like CPU time, memory, I/O etc. Because of high cost of computing, it is desirable to design algorithms that are economical in the use of CPU, time & memory.

(i) Analysis can be more reliable.

(ii) It helps in choosing a solution from many solutions available to a problem.

1.5 ANALYSIS OF ALGORITHM COMPLEXITY

The following are the qualities of the algorithm:

(i) Easily modifiable.

(ii) They should be correct and clearly defined.

(iii) Require lees time, storage, I/O time.

(iv) They are well documented.

(v) They are not dependent on language.

(vi) They are able to use the sub procedures.

(1) Computational complexity: More computing resources are needed to solve larger problems in the same class. Very small problems can be solved with an algorithm that exhibit exponential behaviour.

(2) The order Notation: The o-notation gives an upper behind to a function within a constant factor. For a given function g (n) we denote by o (g (n)).

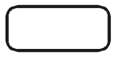
For example a double rested loop strurcture yields o (n^2) upper bound on the worst case time.

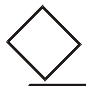
(3) Worst and average case behaviour: In many applications it is important to have a measure of the expected complexity. In linear search the worst case will be to examine n values and average case will be to examine n/2 values.

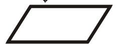
1.6 FLOWCHARTS

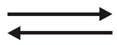
Flowcharts are used in programming to diagram the path in which information is processed through a computer to obtain the desired results. Flowchart is a graphical representation of an algorithm. It makes use of symbols which are connecting among them to indicate the flow of information & processing. It will show the general outline of how to solve a problem or performing a task. It is prepared for better understanding of the altgorithm.

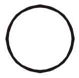
Basic symbol used in flowchart design:

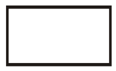

Start/Stop

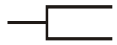

Question Decision (used in branching)


Input/Output

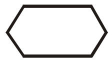

Lines or arrows represent the direction of the flow of control

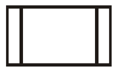

Connector (connect one part of the flowchart to another)


Process, Instruction

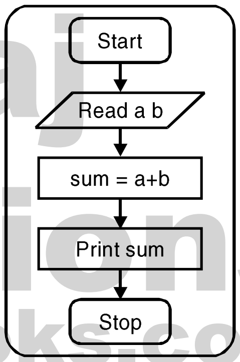

Comments, Explanations, Definitions

Additional symbols related to more advanced programming:

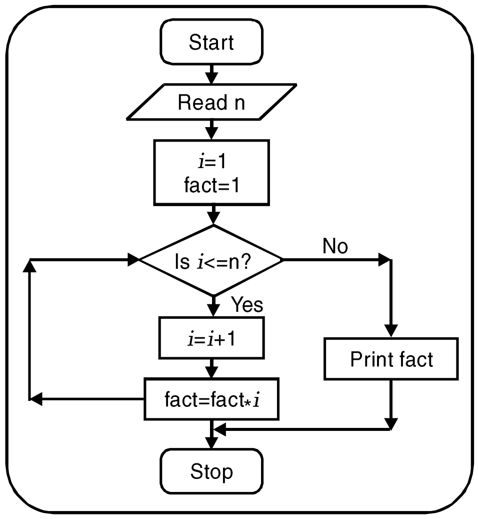

Preparation (may be used with "do loops")


Refers to separate flowchart

Example 1: A flowchart to display the sum of two numbers



Example 2: Flowchart to find factorial of a given numbers.



CHECK YOUR PROGRESS

Q. 1. Differentiate between flowchart and algorithms.

Solution: Algorithms.

1. An algorithm consists of a set of explicit and ambighous finite steps which when caused out for a given set of initial conditions produce the overspending output and terminate in a fixed amout of time. In other words, Algorithm is a process of breaking the given problem into smaller, simpler and interlinked steps arranged sequentially in order to achieve the desired objective.

2. The algorithm is written in English or English like language known as pseudocode. This language is not dependent on any programming language.

Flow chart

1. Flow chart is the pictorial representation of an algorithm.

2. Picture always speak too much than any paragraph. Flowchart gives the clear idea of the process to solve the problem.

3. Some specified geometoric shape boxes are used for the operations and they are connected by across to control the flow. Any error in the logic of the process can be detected easily.

4. A flowchart basically is the plan to be followed when the program is written. It acts like a road map for a programmer and guides him in proceeding from the starting point to the final point, while writing a computer program.

Q. 2. Compute and print the sum of a set of data values.

Solution:

Algorithm to calculate sum of set of values.

1. Start
2. Set the sum variable to zero.
3. Input the data values as long as they exist and add them to sum.
4. Display the value of sum.
5. Stop

Q. 3. Write the steps that are suggested to facilitate the problem solving process using computer.

Solution:

1. Define the problem
2. Analyse the problem for input and output required.
3. Formulate a mathematical and logical model to solve the problem.
4. Develop an algorithm to solve the problem using the model in step 3. Each step should be so simple that it can be converted into equivalent statement in programming language.
5. Design a flowchart.
6. Write a program code of the same.
7. Test the program for errors.

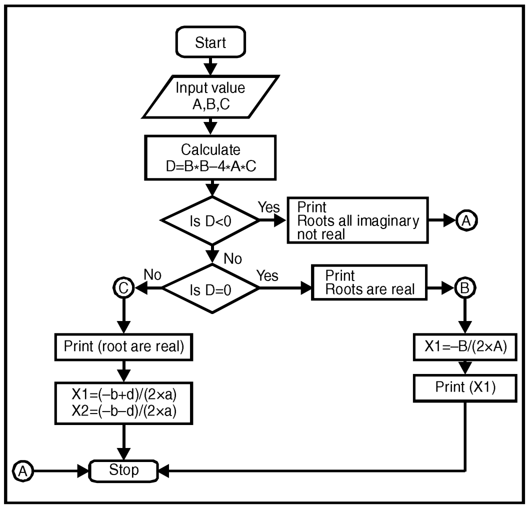
Q. 4. Draw an algorithm and flowchart to calculate the roots of quadratic equation $Ax^2 + Bx+c = o$.

Solution:

Algorithm

```
1. Start
2. Input the value of A, B and C.
3. Calculate D = B * B – 4 * A * C
4. Check if (D < O) print (Roots are imaginary not real)
5. if (D > O) Print+(Roots are Real)
    (a) x1 = - B/(2 * A)
    print (x1)
6. If (D > O) print (Roots are Real and distinct)
    (a) x1 = (– B + D) / (2 * A)]
    (b) x2 = (– B – D) / (2 * A)
    (c) Print (X1, X2)
7. Stop.
```

Flowchart:


Start
Input value A,B,C
Calculate D=B∗B–4∗A∗C
Is D<0
Yes
Print Roots all imaginary not real
A
No
Is D=0
No
C
Yes
Print Roots are real
B
Print (root are real)
X1=(–b+d)/(2×a)
X2=(–b–d)/(2×a)
X1=–B/(2×A)
Print (X1)
A
Stop