

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

© Reserved with the Publishers only.

Typesetting by: Competent Computers

Terms & Conditions for Buying E-Book

- The User must Read & Accept the Terms and Conditions (T&C) carefully before clicking on the accept option for Buying the Online Soft Copy of E-books. Under this Particular Facility you may buy only the Online Soft Copy of E-books, no Hard Copy or Printed Copy shall be provided under this facility.
- These E-Books are valid for 365 days online reading only (From the Date of Purchase) and no kind of Downloading, Printing, Copying, etc. are allowed in this facility as these products are just for Online Reading in your Mobile / Tablet / Computers.
- All the online soft copy E-books given in this website shall contain a diffused watermark on nearly every page to protect the material from being pirated / copy / misused, etc.
- This is a Chargeable Facility / Provision to Buy the Online Soft Copy of E-books available online through our Website Which a Subscriber / Buyer may Read Online on his or her Mobile / Tablet / Computer. The E-books content and their answer given in these Soft Copy provides you just the approximate pattern of the actual Answer. However, the actual Content / Study Material / Assignments / Question Papers might somewhat vary in its contents, distribution of marks and their level of difficulty.
- These E-Books are prepared by the author for the help, guidance and reference of the student to get an idea of how he/she can study easily in a short time duration. Content matter & Sample answers given in this E-Book may be Seen as the Guide/Reference Material only. Neither the publisher nor the author or seller will be responsible for any damage or loss due to any mistake, error or discrepancy as we do not claim the Accuracy of these solution / Answers. Any Omission or Error is highly regretted though every care has been taken while preparing these E-Books. Any mistake, error or discrepancy noted may be brought to the publishers notice which shall be taken care of in the next edition. Please consult your Teacher/Tutor or refer to the prescribed & recommended study material of the university / board / institute / Govt. of India Publication or notification if you have any doubts or confusions before you appear in the exam or Prepare your Assignments before submitting to the University/Board/Institute.
- Publisher / Study Badshah / shall remain the custodian of the Contents right / Copy Right of the Content of these reference E-books given / being offered at the website www.studybadshah.com.
- The User agrees Not to reproduce, duplicate, copy, sell, resell or exploit for any commercial purposes, any portion of these Services / Facilities, use of the Service / Facility, or access to the Service / Facility.
- The Price of these E-books may be Revised / Changed without any Prior Notice.
- The time duration of providing this online reading facility of 365 days may be alter or change by studybadshah.com without any Prior Notice.
- The Right to accept the order or reject the order of any E-books made by any customer is reserved with www.studybadshah.com only.
- All material prewritten or custom written is intended for the sole purpose of research and exemplary purposes only. We encourage you to use our material as a research and study aid only. Plagiarism is a crime, and we condone such behaviour. Please use our material responsibly.
- In any Dispute What so ever Maximum Anyone can Claim is the Cost of a particular E-book which he had paid to Study Badshah company / website.
- If In case any Reader/Student has paid for any E-Book and is unable to Access the same at our Website for Online Reading Due to any Technical Error/ Web Admin Issue / Server Blockage at our Website www.studybadshah.com then He will be send a New Link for that Particular E-Book to Access the same and if Still the Issue is Not Resolved Because of Technical Error/ Web Admin Issue / Server Blockage at our website then His Amount for that Particular Purchase will be refunded by our website via PayTM.
- All the Terms, Matters & Disputes are Subjected to "Delhi" Jurisdiction Only.

CONTENTS

S.No.	Page
1. Problem Solving	1
2. Basics of C	5
3. Variables and Constants	11
4. Expressions and Operators	16
5. Decision and Loop Control Statements	22
6. Arrays	40
7. Strings	49
8. Functions	59
9. Structures and Unions	72
10. Pointers	77
11. The C - Preprocessor	92
12. Files	97
Some More Important Questions	108
Short Answer Type Questions Miscellaneous	125
Important 'C' Programs	130
Appendix-I : ASCII Values of Characters	150
Appendix-II : ANSI 'C' Library Functions	151
Appendix-III : Glossary	155
<i>Sample Question Paper</i>	159

Sample Preview of The Chapter

Published by:



**NEERAJ
PUBLICATIONS**

www.neerajbooks.com

PROBLEM SOLVING AND PROGRAMMING THROUGH C

Problem Solving



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 finally 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 constraints etc.

There is no standard or universal approach to solve a problem. Same problem can be solved in different ways, we cannot 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 experiance 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.

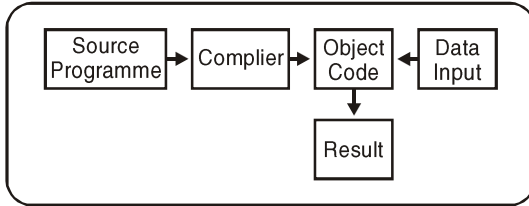
Computers do not work like humans. When computer is used to solve a problem, we must specify the input required, the process to be done and the output required. Once these are considered an algorithm and a program is developed. It is not necessary the programme developed gives the correct result. If 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 softwre. 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.

2 / NEERAJ : PROBLEM SOLVING AND PROGRAMMING THROUGH C

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 SOLVING 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 development 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 develop 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.

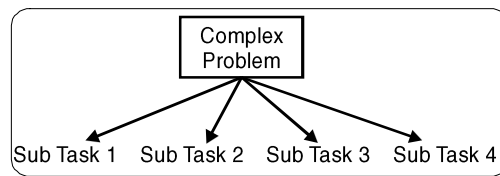
Some examples of algorithm.

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 less 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 bound to a function within a constant factor. For a given function $g(n)$ we denote by $o(g(n))$.



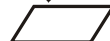
For example a double nested loop structure yields $o(n^2)$ upper bound on the worst case time.


(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 algorithm.

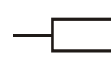
Basic symbol used in flowchart design:

-  Start/Stop
-  Question Decision (used in branching)
-  Input/Output

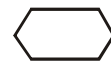
 Lines or arrows represent the direction of the flow of control


 Connector (connect one part of the flowchart to another)

 Process, Instruction

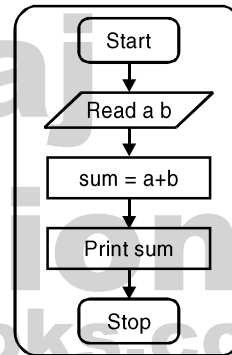
 Comments, Explanations, Definitions

Additional symbols related to more advanced programming:

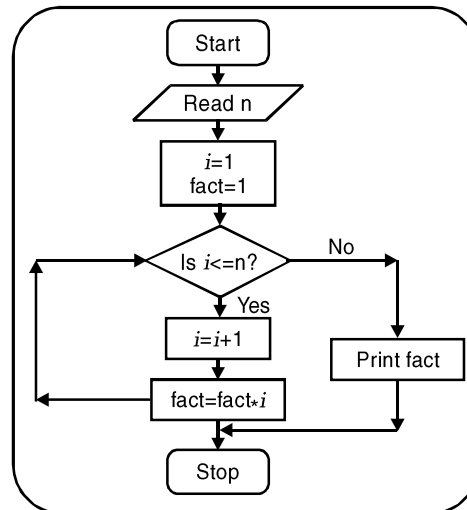
 Preparation (may be used with "do loops")

 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 unambiguous finite steps which when carried out for a given set of initial conditions produce the desired output and terminate in a fixed amount 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. Pictures always speak too much than any paragraph. Flowchart gives the clear idea of the process to solve the problem.

3. Some specified geometric shape boxes are used for the operations and they are connected by arrows 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 $Ax^2 + Bx + c = 0$.

Solution:

Algorithm

1. Start
2. Input the value of A, B and C.
3. Calculate $D = B * B - 4 * A * C$
4. Check if $(D < 0)$ print (Roots are imaginary not real)
5. if $(D > 0)$ Print+(Roots are Real)
 - (a) $x_1 = -B / (2 * A)$
 - print (x_1)
6. If $(D > 0)$ print (Roots are Real and distinct)
 - (a) $x_1 = (-B + D) / (2 * A)$
 - (b) $x_2 = (-B - D) / (2 * A)$
 - (c) Print (X_1, X_2)
7. Stop.

Flowchart:

