AMRITA VISHWA VIDYAPEETHAM CHENNAI CAMPUS

Second Year. B.Tech (Computer Science and Engineering)

Problem Solving using ‘C’ programming

Work Book

(From Academic year 2025-26 ODD)

Name: S DEEPAK BHARATHWAJ

College Name: Amrita Vishwa Vidyapeetham, Chennai

Roll No.: CH.SC.U4CSE24015 Department: CSE

Academic Year: 2025-26

Preface to the First Edition

The new syllabus for First Year B.Tech Computer Science is implemented from the academic year 2022.

It is absolutely necessary and essential that all the Computer Science practicals be conducted on Open Source Operating System like Linux or windows. All the practical's related to C needs to be conducted using GCC compiler.

.

It is mandatory to carry the completed and duly signed lab Activity sheets for the practical examination.

Lab Incharge: Dr J.Umamageswaran

Introduction

1. About the work book

This workbook is intended to be used by First Year B.Tech. (Computer Science) students for the Computer Science laboratory courses in their curriculum. In Computer Science, hands-on laboratory experience is critical to the understanding of theoretical concepts studied in the theory courses. This workbook provides the requisite background material as well as numerous computing problems covering all difficulty levels.

The objectives of this book are

1) Defining clearly the scope of the course
2) Bringing uniformity in the way the course is conducted across different colleges
3) Continuous assessment of the course
4) Bring in variation and variety in the experiments carried out by different students in a batch
5) Providing ready reference for students while working in the lab
6) Catering to the need of slow paced as well as fast paced learners

2. How to use this work book?

This workbook is mandatory for the completion of the laboratory course. It is a measure of the performance of the student in the laboratory for the entire duration of the course.

2.1 Instructions to the students

Please read the following instructions carefully and follow them

1) Carry this book every time you come to the lab for computer science practical's
2) You should prepare yourself beforehand for the Exercise by reading the material mentioned under.
3) If the self-activity exercise or assessment work contains any blanks such as _____, get them filled by your instructor.
4) Instructor will specify which problems you are to solve by ticking box
5) You will be assessed for each exercise on a scale of 5

i) Not done	0	ii) Incomplete	1
iii) Late Complete	2	iv) Needs improvement	3
v) Complete	4	vi) Well Done	5

2.2. Instruction to the Instructors

1) Explain the assignment and related concepts in around ten minutes using white board if required or by demonstrating the software
2) Fill in the blanks with different values for each student
3) Choose appropriate problems to be solved by student by ticking box
4) Make sure that students follow the instruction as given above
5) After a student completes a specific set, the instructor has to verify the outputs and sign in the provided space after the activity.
6) You should evaluate each assignment carried out by a student on a scale of 5 as specified above by ticking appropriate box.
7) The value should also be entered on assignment completion page of the respective Lab course

2.3. Instructions to the Lab administrator

You have to ensure appropriate hardware and software is made available to each student.

The operating system and software requirements on server side and also client side are as given below

1) Server Side (Operating System)
    a. * Fedora Core Linux *
    b. Servers Side (software's to be installed)

    In Linux – C, C++, awk, shell, perl, postgresql/Mysql

2) Client Side (Operating System)

    * Red Hat Linux and Fedora Core *

    Client Side (software's to be installed)

    In Linux – C, C++, awk, shell, perl, postgresql/mysql

Information about installation and configuring of the server and client are provided in appendix A

Activity Completion Sheet

Problem Solving and C programming		
Sr.No	**Assignment Name**	**Marks**
1	Problem Solving using Pseudo code and Flowchart, Simple programs, Understanding errors and error handling.	
2	Decision Making Control Structures.	
3	Loop Control Structures	
4	Functions (User Defined functions, Library functions and Recursion).	
5	Arrays (1-D and 2-D)	
6	Strings	
7	Pointers	

CERTIFICATE

This is to certify that, Mr./Miss._____Roll No._____of I year B.Tech. (Computer Science and Engineering) has successfully completed __out of ___ practical's satisfactorily during the academic year 20 - 20 .

Practical In-charge

Chairperson

Problem Solving using ‘C’ programming

You should read following topics before starting this exercise

1. **UNIX and LINUX operating system**

About UNIX and LINUX

The success story of UNIX starts with the failure of the MULTICS project. The project failed and the powerful GE-645 machine was withdrawn by GE. Two scientists at Bell Labs, Ken Thompson and Dennis Ritchie, who were part of the MULTICS team, continued to work and succeeded and named their Operating system UNIX, a pun on MULTICS.

The machine available at Bell Labs was a DEC PDP-7 with only 64 k memory while the Operating system they were developing was meant for a larger machine. The problematic situation was handled with an innovative solution. They developed most part of the software in a higher-level language, C, which helped them in porting their Operating system from one hardware to another.

With the growing popularity of UNIX, it was available on a variety of machines, from personal computers to mainframes. The most popular amongst them was UNIX System V from AT&T.

Each big player in the market came up with their own versions of UNIX. IBM had its own version of UNIX called AIX, which was used on high-end servers. Sun's version of UNIX called Solaris was used on Sun workstations. Novell marketed UnixWare along with Netware, its Network operating system.

LINUX is a version of UNIX, which though it resembles UNIX in looks and feels but differs from other versions in the way it was developed and distributed. In contrast to large proprietary UNIX versions, Linux was developed by Linus Torvalds, a Finnish student. He made the source code available and invited partners via the internet in his development effort. He got professional help from all quarters and Linux evolved rapidly. It was made freely available for everyone to use. Linux that was initially meant for Personal computers is now available for a variety of hardware platforms, from mainframes to handheld computers

Linux supports multiple users. Every user needs to have an account in order to use the system. One of the users called system administrator (root) is given the charge of creating user accounts and managing the system normally works on the "#" prompt.

You will be given a username and password, using which you can login into Linux operating system.

For computer users, the operating system provides a user-command interface that is easy to use, usually called the Shell. The user can type commands at the shell prompt and get the services of the operating system. Linux operating system shell has the "$" prompt.

You can open a system terminal that gives you a $ prompt where you can type in various shell commands.

LINUX system will usually offer a variety of shell types:

- sh or Bourne Shell: the original shell still used on UNIX systems and in UNIX-related environments. It is available on every Linux system for compatibility with UNIX programs.
- bash or Bourne Again shell: the standard GNU shell, is the standard shell for common users on Linux and is a superset of the Bourne shell.
- csh or C shell: the syntax of this shell resembles that of the C programming language.
- tcsh or Turbo C shell: a superset of the common C shell, enhancing user-friendliness and speed.
- ksh or the Korn shell: A superset of the Bourne shell

Activity Sheet 1 **Date:**

Problem Solving using Pseudo code and Flowchart, Simple programs, understanding errors and error handling.

Objective-To demonstrate the use of data types, simple operators and expressions

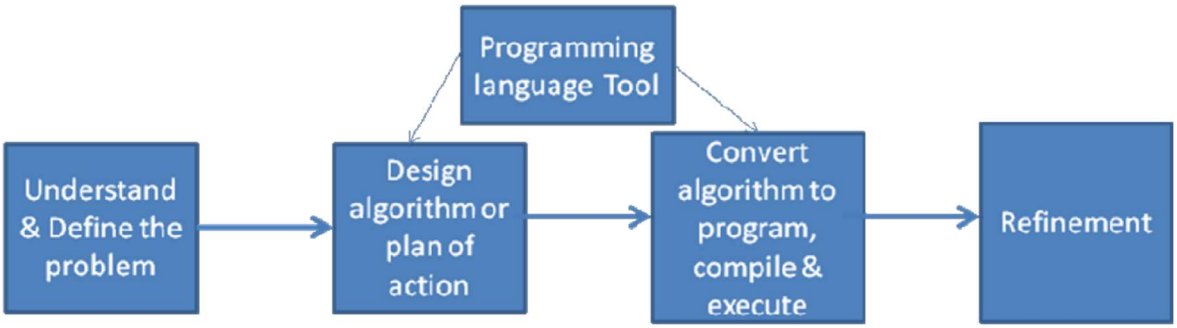
Reading-

You should read following topics before starting this exercise

1. Problem solving steps- writing algorithms and flowcharts
2. Different basic data types in C and rules of declaring variables in C
3. Different operators and operator symbols in C
4. How to construct expressions in C, operator precedence

Program solving using Computers

The steps in solving a problem using computers are shown below



Pseudo Language conventions

1. Appropriate names should be given to variables – the algorithm itself is given a name

   Algorithm FindingMaximum

2. Organized sequence of data values is given a name and each data value is indicated by giving the index in brackets

   Values[position]

3. Expressions containing Arithmetic operators +, -, * , / relational operators <, ≤ , >, ≥, ≠, = and logical and, or , not can be used

   Example: length * breadth, previous < next

4. Assignment statement variable = expression is used to assign data values to variables

5. Conditional constructs for making decisions

   if condition then statements

   Statements are to be carried out only if condition is true

   if condition then statements1 else statements2

   Statements1 are to be carried out only if condition is true and statements2 are to be carried out if condition is false

6. Loops

   i. while condition do statements

      Statements are to be repeated while condition is true

   ii. for variable = value1 to value2 do statements

      Statements are to be repeated moving from value1 to value2, with an increment one step.

   iii. for variable = value1 downto value2 do { statements }

      Statements are to be repeated moving from value1 down to value2, with a decrement one step.

   iv. repeat { statements } until condition

      Statements are to be repeated until condition is true

7. Input /output statements

   i. read value For taking value as input for the algorithm

   ii. write value For printing value as output of the algorithm

8. Algorithm name(value1 , valu2) – taking input for the algorithm

9. return value1 , value2 – for giving output of the algorithm

Examples:

Problem Statement: Accept radius and calculate area and circumference of a circle

```
Algorithm AreaCircumference
Begin
        Input radius pi=3.142
        area = piXradiusXradius
        circum= 2XpiXradius  Output
area
        Output circum
End
```

Problem Statement: Check if a number is even

```
Algorithm Even
Begin Input m if m mod 2 = 0 then
        output “m is even”
End
```

Problem Statement: Find maximum of two numbers

```
Algorithm Maximum
Begin
        Input m Input n if
        m > n then  output
        m
        else  output n
End
```

Problem Statement: Give a discount of 15 % when purchase amount exceeds 5000, otherwise give a discount of 10%

```
Algorithm Discount Input
amount
if amount exceeds  5000 then  compute discount as 25 times
        amount divided by 100
else compute discount as 15 times amount divided by 100
Subtract discount from amount
Output amount
End
```

Problem Statement: Given a set of 100 values representing marks of students, count the total students that have passed. (A score of 40 is required for passing.)

```
Algorithm PassCount1
Begin
         count = 0
        n = 1
         While n <= 100 do
                          Input marks
                        If marks >= 40 then
                                  increment count by 1
                           n = n+1
        Output count
 End
```

The same can be done using another loop construct as shown below:

```
Algorithm PassCount1
Begin
         count = 0
        for n = 1 to 100 do
                          Input marks
                        If marks >= 40 then  increment
        count by 1 Output count
 End
```

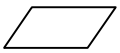
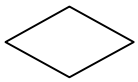
Problem Statement: Accept characters till a * is entered from the keyboard and count the number of characters entered.

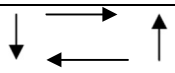
```
Algorithm CharacterCount
Begin
        count = 0
        Input char
        while char ≠ * do
count = count + 1
                        Input char
        Output count
 End
```

Problem Statement: Accept a number and calculate the sum of its digits.

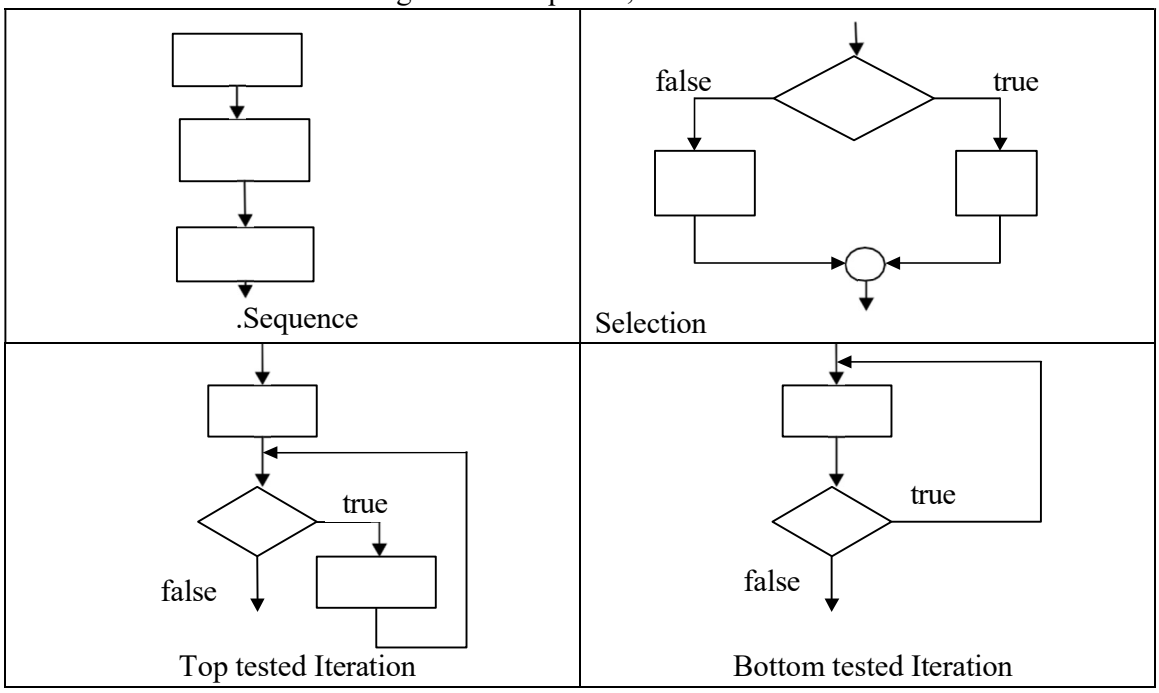
```
Algorithm SumDigits
Begin
        Input num
        sum=0
        repeat
                        digit = num mod 10
                        sum=sum+digit
        num = num/10
        until num>0 Output
        sum
 End
```

FLOWCHART SYMBOLS

Start Statement	
Input Statement	
Statement or procedure	
Decision, choice or Selection	
Connectors	

Control flow	

Basic control structures in an algorithm: Sequence, Selection and Iteration are shown below



1. Data type Table

Data	Data Format	C Data Type	C Variable declaration	Input Statement	Output statement

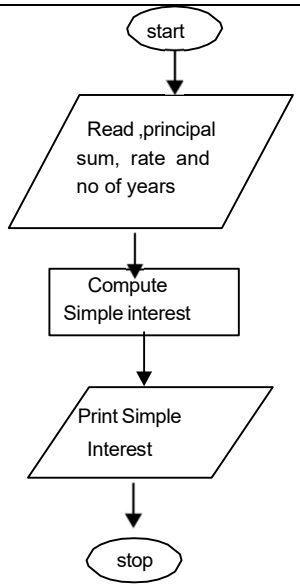
quantity
month
creditcard number
	Numeric	int
		Short int
		long int
			int quantity;
			short month;
			long ccno;
				scanf("%d",&quantity);
				scanf("%d",&month);
				scanf("%ld", &ccno);
					printf("The quantity is %d", quantity);
					printf("Credit card number is %ld, ccno);

price π	real	float double	float price; const double pi=3.141593 ;	scanf("%f",&price);	printf("The price is %5.2f", price);
grade	character		char grade;	scanf("%c",&grade)	printf("The grade is %c",grade);

2. Expression Examples

Expression	C expression
Increment by a 3	a = a + 3
Decrement b by 1	b = b-1 or b--
2 a^2 + 5 b/2	2*a*a + 5*b/2
7/13(x-5)	(float)7/13*(x-5)
5% of 56	(float)5/100*56
n is between 12 to 70	n>=12 && n<=70
$\pi r^2 h$	Pi*r*r*h
n is not divisible by 7	n % 7 != 0
n is even	n%2== 0
ch is an alphabet	ch>='A' && ch<='Z' || ch>='a' && ch<='z'

Step 1: Writing the Pseudocode	Step 2: Draw the flowchart	Step 3: Write the Program

Algorithm SimpleInterest
Begin
Input amount, rate, years
si =amountXyearsXrate / 100
Output si
End
		#include <stdio.h>
		void main( )
		{
		/* variable declarations */
		float amount, rateOfInterest, simpleInterest;
		int noOfYears;
		/* prompting and accepting input */
		printf("Give the Principal Sum");
		scanf("%f",&amount);
		printf("Give the Rate of Interest");
		scanf("%f",&rateOfInterest);
		printf("Give the Number of years");
		scanf("%d",&noOfYears);
		/* Compute the simple Interest*/
		simpleInterest=amount*noOfYears*rateOfInterest / 100 ;
		/* Print the result*/
		printf("The simple Interest on amount %7.2f for %d years at the rate %4.2f is %6.2f", amount, noOfYears, rateOfInterest, simpleInterest);
		}

Self-Activity

1. Type the sample program given above. Execute it for the different values as given below and fill the last column from the output given by the program. Follow the following guidelines

a. At $ prompt type gedit followed by filename. The filename should have .c as extension for example $gedit pnr.c

b. Type the sample program and save it. Compile the program using cc compiler available in Linux
$cc pnr.c
It will give errors if any or it will give back the $ prompt if there are no errors
A executable file a.out is created by the compiler. The program can be executed by typing name of the file as follow $ ./a.out
Alternatively, the executable file can be given name by using –o option while compiling as follows $cc pnr.c –o pnrexec
$./pnrexec

Sr. No	Principal sum	No of years	Rate of interest	Simple Interest
1	2000	3	____	
2	4500	___	4.5	
3	_____	6	8.3	