Parul University

FACULTY OF ENGINEERING & TECHNOLOGY

BACHELOR OF TECHNOLOGY

Computational Thinking for Structure Design -1
(303105104)

1st SEMESTER

COMPUTER SCIENCE & ENGINEERING DEPARTMENT

Laboratory Manual

PREFACE

It gives us immense pleasure to present the first edition of ***Computational Thinking for Structure Design -1*** for the B.Tech. 1st year students for PARUL UNIVERSITY.

The Fundamental of Programming theory and laboratory courses at **PARUL UNIVERSITY, WAGHODIA, VADODARA** are designed in such a way that students develop the basic understanding of the subject in the theory classes and then try their hands on the computer learnt during the theoretical sessions.

This book is emphatically not focused on "the syntax of C". Understanding the fundamental ideals, principals, and techniques is the essence of a good programmer. Only well-designed code has a chance of becoming part of a correct, reliable, and maintainable system. Also, "the fundamentals" are what last: they will still be essential after today's language and tools have evolved or been replaced.

We acknowledge the authors and publishers of all the books which we have consulted while developing this Practical book. Hopefully this *Computational Thinking for Structure Design -1* will serve the purpose for which it has been developed.

Instructions to students

1. Every student should obtain a copy of laboratory Manual.
2. Dress Code: Students must come to the laboratory wearing.
   i. Trousers,
   ii. half-sleeve tops and
   iii. Leather shoes. Half pants, loosely hanging garments and slippers are not allowed.
3. To avoid injury, the student must take the permission of the laboratory staff before handling any machine.
4. Students must ensure that their work areas are clean and dry to avoid slipping.
5. Do not eat or drink in the laboratory.
6. Do not remove anything from the computer laboratory without permission.
7. Do not touch, connect or disconnect any plug or cable without your lecturer/laboratory technician's permission.
8. All students need to perform the practical/program.

CERTIFICATE

This is to certify that

Mr./Ms ..... with enrolment no. ..... has successfully completed his/her laboratory experiments in the ***Computational Thinking for Structure Design -1 (303105104)*** *from the department of ..... during the academic year .....*

Date of Submission:..... Staff In charge:.....

Head of Department:.....

INDEX

Class: 1st Sem **Subject: - Computational Thinking for Structured Design-1**
A.Y. 20__-20__ **Subject Code: 303105104**

Sr. No.	Experiment Title	Page No.		Date of Performance	Date of Assessment	Marks out 0f 10	Sign
	Practical Set – 1	**To**	**From**				
1	Installation C IDE, Basic Structure of C program.Format Specifiers, Escape Character. Run time input/Output Programs.						
2	1. Write a c program to calculate Area of Rectangle,Perimeter of a Rectangle and Diagonal of a Rectangle. 2. Write a c program to calculate Area of square,Perimeter of a square and Diagonal of a square. 3. Write a c program to calculate total area ofCylinder and volume of a cylinder.						
3	1. The total distance traveled by vehicle in ?t seconds is given by distance s = ut+1/2at? where ?u??and ?a??are the initial velocity (m/sec.) and acceleration(m/sec?). Write a C program to find the distance traveled at regular intervals of time given the values of ?u??and ?a???The program should provide the flexibility to the user to select his own time intervals and repeat the calculations fordifferent values of ?u??and ?a?? 2. Write a C program, which takes two integer operands and one operator from the user, performs the operation and then prints the result. (Consider the operators ??????????????and use Switch Statement)						
4	1. Write a C program to find the sum of individual digits of a positive integer. 2. A Fibonacci sequence is defined as follows: the first and second terms in the sequences are 0 and 1. Subsequent terms are						

	found by adding the preceding two terms in the sequence. Write a C program to generate the first n terms of the sequence. 3. Write a C program to generate all the prime numbers between 1 and n, where n is a value supplied by the user.						
5	1. Write a C program to calculate the following Sum: Sum=1-x^2/2! +x^4/4!-x^6/6!+x^8/8!-x^10/10!. 2. Write a C program to find the roots of a quadratic equation.						
6	Write C programs that use both recursive and non-recursive functions. 1. To find the factorial of a given integer. 2. To find the GCD (greatest common divisor) of two given integers.						
7	1. Write a C program to find the largest integer in a list of integers, 2. Write a C program that uses functions to perform the following: 1. Addition of Two Matrices 2. Multiplication of Two Matrices						
8	1. Write a C program that uses functions to perform the following operation; 1. To insert a sub-string into a given main string from a given position. 2. To delete n Characters from a given position in a given string, 2. Write a C program to determine if the given string is a palindrome or not.						
9	1. Write a C program that displays the position or index in the string S where the string T begins, or -1 if S doesn□t contain T. **2.** Write a C program to count the lines, words and characters in a given text.						
10	1. Write a C program to generate Pascal's triangle. 2. Write a C program to construct a pyramid of numbers.						
11	Write a C program to read in two numbers, x and n, and then compute the sum of this geometric progression: 1+x+x^2+x^3+... x^n. For example: if n is 3 and x is 5, then the program computes 1+5+25+125.Print x, n, the sum. Perform error checking. For example, the formula does not make sense for negative exponents □□if n is less than 0. Have						

	your program print an error - message if n<0, then go back and read in the next pair of numbers without computing the sum. Are any values of x also illegal? If so, test for them too.						
12	1. 2's complement of a number is obtained by scanning it from right to left and complementing all the bits after the first appearance of a 1. Thus 2□s complement of 11100 is 00100. Write a C program to find the 2's complement of a binary number. 2. Write a C program to convert a Roman numeral to its decimal Equivalent.						
13	1. Write a c program on Given an unsorted array arr[] of size N. Rotate the array to the left (counter-clockwise direction) by D steps, where D is a positive integer. 2. Write a c Program on given two sorted arrays arr1 and arr2 of size N and M respectively and an element K. The task is to find the element that would be at the k□th position of the final sorted array.Explanation: Input : Array 1 - 1 4 2 3 5 Array 2 - 7 8 6 k = 5 Output : 5 Because The final sorted array would be -1, 2, 3, 4, 5, 6, 7, 8, The 5th element of this array is 6.						
14	1. Write a c program to take multiline string input and print individual string length . 2. Write a c program to reverse the individual word of a given string Explanation:input : Welcome To Bytexl output: emocleW oT lxetyB.						

Practical 1

1. **Installation C IDE, Basic Structure of C program. Format Specifiers, Escape Character. Run time input/Output Programs.**

The basic structure of a C program is divided into 6 parts which makes it easy to read, modify, document, and understand in a particular format. C program must follow the below-mentioned outline in order to successfully compile and execute. Debugging is easier in a well-structured C program.

Sections of the C Program
There are 6 basic sections responsible for the proper execution of a program. Sections are mentioned below:

a. Documentation
b. Preprocessor Section
c. Definition
d. Global Declaration
e. Main() Function
f. Sub Programs

1. Documentation
This section consists of the description of the program, the name of the program, and the creation date and time of the program. It is specified at the start of the program in the form of comments. Documentation can be represented as:

```
// description, name of the program, programmer name, date, time etc.
/*
   description, name of the program, programmer name, date, time etc.
*/
```

Anything written as comments will be treated as documentation of the program and this will not interfere with the given code. Basically, it gives an overview to the reader of the program.

2. Preprocessor Section
All the header files of the program will be declared in the preprocessor section of the program. Header files help us to access other's improved code into our code. A copy of these multiple files is inserted into our program before the process of compilation.

Example:

```
#include<stdio.h>
#include<math.h>
```

3. Definition

Preprocessors are the programs that process our source code before the process of compilation. There are multiple steps which are involved in the writing and execution of the program. Preprocessor directives start with the '#' symbol. The #define preprocessor is used to create a constant throughout the program. Whenever this name is encountered by the compiler, it is replaced by the actual piece of defined code.

Example:

```
#define long long ll
```

4. Global Declaration

The global declaration section contains global variables, function declaration, and static variables. Variables and functions which are declared in this scope can be used anywhere in the program.

Example:

```
int num = 18;
```

5. Main() Function

Every C program must have a main function. The main() function of the program is written in this section. Operations like declaration and execution are performed inside the curly braces of the main program. The return type of the main() function can be int as well as void too. void() main tells the compiler that the program will not return any value. The int main() tells the compiler that the program will return an integer value.

Example:

```
void main()
or
int main()
```

6. Sub Programs

User-defined functions are called in this section of the program. The control of the program is shifted to the called function whenever they are called from the main or outside the main() function. These are specified as per the requirements of the programmer.

Example:

```
int sum(int x, int y)
{
   return x+y;
}
```

List of Format Specifiers in C

The below table contains the most commonly used format specifiers in C

Format Specifier	Description
%c	For b type.
%d	For signed integer type.
%e or %E	For scientific notation of floats.
%f	For float type.
%g or %G	For float type with the current precision.
%i	Unsigned integer
%ld or %li	Long
%lf	Double
%Lf	Long double
%lu	Unsigned int or unsigned long
%lli or %lld	Long long
%llu	Unsigned long long
%o	Octal representation
%p	Pointer
%s	String
%u	Unsigned int

Format Specifier	Description
%x or %X	Hexadecimal representation
%n	Prints nothing
%%	Prints % character

Escape Sequence List

The table below lists some common escape sequences in C language.

Escape Sequence	Name	Description
\a	Alarm or Beep	It is used to generate a bell sound in the C program.
\b	Backspace	It is used to move the cursor one place backward.
\f	Form Feed	It is used to move the cursor to the start of the next logical page.
\n	New Line	It moves the cursor to the start of the next line.
\r	Carriage Return	It moves the cursor to the start of the current line.
\t	Horizontal Tab	It inserts some whitespace to the left of the cursor and moves the cursor accordingly.
\v	Vertical Tab	It is used to insert vertical space.
\\	Backlash	Use to insert backslash character.
\’	Single Quote	It is used to display a single quotation mark.
\”	Double Quote	It is used to display double quotation marks.
\?	Question Mark	It is used to display a question mark.

Escape Sequence	Name	Description
\ooo	Octal Number	It is used to represent an octal number.
\xhh	Hexadecimal Number	It represents the hexadecimal number.
\0	NULL	It represents the NULL character.

Example:

```
#include <stdio.h>
int main() {
  // printf() displays the string inside quotation
  printf("Hello, World!");
  return 0;
}
```

Practical 2

1. Write a c program to calculate Area of Rectangle,Perimeter of a Rectangle and Diagonal of a Rectangle.

```
#include <stdio.h>
#include <math.h>
int main()
{
   int l,b,a,p,d;
   printf("Enter the value of l=");
   scanf("%d",&l);
   printf("Enter the value of b=");
   scanf("%d",&b);
   a= l*b;
   printf("\n the area of the rectangle is %d",a);
   p= 2*(l+b);
   printf("\n the perimeter of the rectangle is %d",p);
   d= sqrt(l*l +b*b);
   printf("\n the diagonal of the rectangle is %d",d);
  return 0;
}
```

OUTPUT

Enter the value of l=20

Enter the value of b=20

the area of the rectangle is 400

the perimeter of the rectangle is 80

the diagonal of the rectangle is 28.2

2. Write a c program to calculate Area of square,Perimeter of a square and Diagonal of a square.

```
#include <stdio.h>
#include <math.h>
int main ()
{
   float A,a,D,P;
   printf("ENTER THE SIDE OF THE SQUARE   ");
    scanf("%f",&a);
    A=a*a;
    P=4*a;
    D=sqrt(2)*a;
    printf("AREA OF THE SQUARE  %f cmsq \n ",A);
    printf("PERIMETER OF THE SQUARE  %f cmsq \n ",P);
    printf("DIAGONAL OF THE SQUARE  %f cmsq \n ",D);
    return 0;
}
```

OUTPUT

```
ENTER THE SIDE OF THE SQUARE   20
AREA OF THE SQUARE  400.000000 cmsq
 PERIMETER OF THE SQUARE  80.000000 cmsq
 DIAGONAL OF THE SQUARE  28.284271 cmsq
```

3. Write a c program to calculate total area of Cylinder and volume of a cylinder.

```
#include <stdio.h>
#include <math.h>
```

```
{
   float A,r,h,V, pi=3.14;
    printf("ENTER THE HEIGHT OF THE CYLINDER   ");
    scanf("%f",&h);
    printf("ENTER THE RADIUS OF THE CYLINDER   ");
    scanf("%f",&r);
    A=2*pi*r(h+r);
    V=pi*pow(r,2)*h;
    printf("AREA OF THE CYLINDER  %f cmsq \n ",A);
     printf("VOLUME OF THE CYLINDER  %f cmsq \n ",V);
    return 0;
    }
```

OUTPUT

```
ENTER THE HEIGHT OF THE CYLINDER   15
ENTER THE RADIUS OF THE CYLINDER   10
AREA OF THE CYLINDER  1570.79 cmsq
 VOLUME OF THE CYLINDER  4712.390 cmsq
```

Practical 3

1. The total distance traveled by vehicle in ‘t seconds is given by distance s = ut+1/2at? where u and a are the initial velocity (m/sec.) and acceleration(m/sec?). Write a C program to find the distance traveled at regular intervals of time given the values of ‘u’ and ‘a’. The program should provide the flexibility to the user to select his own time intervals and repeat the calculations for different values of ‘u’ and ‘a’.

```
#include<stdio.h>
#include<math.h>

int main()

{

  int i, n, sec;

  float d, u, a;

  printf("Enter the no. of intervals\n");

  scanf("%d", &n);

  for(i = 1; i <= n; i++)

  {

    printf("interval: %d \n", i);

    printf("Enter the time in seconds \n");

    scanf("%d",&sec);

    printf("Enter the velocity \n");

    scanf("%f", &u);

    printf("Enter the acceleration \n");

    scanf("%f", &a);

    d= d + (u * sec + (a * (pow(sec, 2))) / 2);

  }

  printf("Total distance travelled is  %.2f", d);

  return 0;
```

```
}
```

OUTPUT

```
Enter the no. of intervals 3
interval: 1
Enter the time in seconds 30
Enter the velocity 30
Enter the acceleration 30
interval: 2
Enter the time in seconds 60
Enter the velocity 60
Enter the acceleration 60
interval: 3
Enter the time in seconds 60
Enter the velocity 30
Enter the acceleration 60
Total distance travelled is  235800.00
```

2. Write a C program, which takes two integer operands and one operator from the user, performs the operation and then
prints the result. (Consider the operators and use Switch Statement)

```
#include<stdio.h>
#include<math.h>
int main(){
    int a,b,op,i=0;
   for(int i;i<=5;i++)
     {
      printf(" \n enter the 2 values ");
    scanf("%d %d",&a,&b);
    printf("what operation do you want to do 1 for +,2 for -,3 for /,4 for *, 5 for %");
    scanf("%d",&op);
     switch (op)
```

```
      {
      case 1:
      printf("addition of no. is: %d",a+b);
      break;
      case 2:
      printf("subtraction of no. is: %d",a-b);
      break;
       case 3:
      printf("division of no. is: %d",a/b);
      break;
       case 4:
      printf("multiplication of no. is: %d",a*b);
      break;
       case 5:
      printf("modulus of no. is: %d",a%b);
      break;
      default:
      printf("invalid input");
      break;
   }

 }
   return 0;
}
```

OUTPUT

```
enter the 2 values 12 2
what operation do you want to do 1 for +,2 for -,3 for /,4 for *, 5 for  2
subtraction of no. is: 10
 enter the 2 values 12 2
 what operation do you want to do 1 for +,2 for -,3 for /,4 for *, 5 for 1
 addition of no. is: 14
 enter the 2 values 12 2
 what operation do you want to do 1 for +,2 for -,3 for /,4 for *, 5 for 3
 division of no. is: 6
 enter the 2 values 12 2
 what operation do you want to do 1 for +,2 for -,3 for /,4 for *, 5 for 4
 multiplication of no. is: 24
 enter the 2 values 12 2
 what operation do you want to do 1 for +,2 for -,3 for /,4 for *, 5 for 5
```

subtraction of no. is: 0
enter the 2 values 12 2
what operation do you want to do 1 for +,2 for -,3 for /,4 for *, 5 for 6
invalid input

Practical 4

1. Write a C program to find the sum of individual digits of a positive integer.

```
#include <stdio.h>
int main()
{
   int a,b,c;
   printf("Enter the value of a=");
   scanf("%d",&a);
   printf("Enter the value of b=");
   scanf("%d",&b);
   c= a+b;
   printf("\n the addition of two number = %d",c);
   return 0;
}
```

OUTPUT

```
Enter the value of a=15
Enter the value of b=15
the addition of two number = 30
```

2. A Fibonacci sequence is defined as follows: the first and second terms in the sequences are 0 and 1. Subsequent terms are found by adding the preceding two terms in the sequence. Write a C program to generate the first n terms of the sequence.

```
#include<stdio.h>
int main()
  {
 int a=0,b=1,c,i,n;
 printf("Enter the values of n :");
 scanf("%d",&n);
 printf("\n%d %d",a,b);
 for(i=2;i<n;++i)
   {
 c=a+b;
 printf(" %d",c);
 a=b;
 b=c;
```

```
        }
  return 0;  }
```

OUTPUT

```
Enter the values of n :6
0 1 1 2 3 5
```

3. Write a C program to generate all the prime numbers between 1 and n, where n is a value supplied by the user

```
#include <stdio.h>

int main() {

  int n, i, j;

  int isPrime;

  printf("Enter a positive integer: ");

  scanf("%d", &n);

  printf("Prime numbers between 1 and %d are: ", n);

  for (i = 2; i <= n; ++i) {

    isPrime = 1;

    for (j = 2; j <= i / 2; ++j) {

      if (i % j == 0) {

        isPrime = 0;

        break;    }  }

    if (isPrime) {

      printf("%d ", i);    }  }

  return 0;}
```

OUTPUT

```
Enter a positive integer: 15
Prime numbers between 1 and 15 are: 2 3 5 7 11 13
```

Practical 5

1.Write a program to calculate the following Sum:
Sum=1-x^2/2!+x^4/4!-x^6/6!+x^8/8!-x^10/10!

```
#include <stdio.h>
#include <math.h>
int factorial(int n) {
    int fact = 1;
    for (int i = 1; i <= n; i++) {
        fact *= i;
    }
    return fact;}
int main() {
    double x;
    int n;
    double sum = 1.0;
    printf("Enter the value of x: ");
    scanf("%lf", &x);
    printf("Enter the number of terms (n): ");
    scanf("%d", &n);
    int sign = -1;
    for (int i = 2; i <= 2 * n; i += 2) {
        sum += (sign * pow(x, i) / factorial(i));
        sign *= -1; }
    printf("Sum = %.6f\n", sum);
    return 0;}
```

OUTPUT**:**

Enter the value of x: 1

Enter the number of terms (n): 1

Sum = 0.500000

2.Write a C program to find the roots of a quadratic equation.

```
#include <stdio.h>
#include <math.h>
int main() {
    double a, b, c;
    double discriminant, realPart, imaginaryPart;
    printf("Enter coefficients a, b, and c: ");
    scanf("%lf %lf %lf", &a, &b, &c);
    discriminant = b * b - 4 * a * c;
    if (discriminant > 0) {
        double root1 = (-b + sqrt(discriminant)) / (2 * a);
        double root2 = (-b - sqrt(discriminant)) / (2 * a);
        printf("Two distinct real roots: root1 = %.2f and root2 = %.2f\n", root1, root2);
    } else if (discriminant == 0) {
        double root = -b / (2 * a);
        printf("One real root: root = %.2f\n", root);
    } else {
        realPart = -b / (2 * a);
        imaginaryPart = sqrt(-discriminant) / (2 * a);
        printf("Complex roots: root1 = %.2f + %.2fi and root2 = %.2f - %.2fi\n", realPart, imaginaryPart,
realPart, imaginaryPart);    }
return 0;}
```

OUTPUT:

coefficients a, b, and c: 4 5

Complex roots: root1 = -0.62 + 1.05% and root2 = -0.6 2 - 1.051

Practical 6

Write C programs that use both recursive and non-recursive functions.

1. To find the factorial of a given integer.

RECURSIVE FUN:

```
#include <stdio.h>
// Recursive function to find the factorial of a number
unsigned long long int factorial_recursive(int n) {
    if (n == 0 || n == 1) {
        return 1;    }
    return n * factorial_recursive(n - 1);
}
int main() {
    int num;
    printf("Enter a non-negative integer: ");
    scanf("%d", &num);
    if (num < 0) {
        printf("Factorial is not defined for negative numbers.\n");
    } else {
        unsigned long long int result = factorial_recursive(num);
        printf("Factorial of %d is %llu\n", num, result);
    }
    return 0;
}
```

NON-RECURSIVE FUN:

```
#include <stdio.h>
// Non-recursive function to find the factorial of a number
unsigned long long int factorial_non_recursive(int n) {
    if (n < 0) {
        return 0;
    }
    unsigned long long int result = 1;
    for (int i = 1; i <= n; i++) {
        result *= i;
    }
    return result;
}
int main() {
    int num;
    printf("Enter a non-negative integer: ");
    scanf("%d", &num);
    if (num < 0) {
        printf("Factorial is not defined for negative numbers.\n");
    } else {
        unsigned long long int result = factorial_non_recursive(num);
        printf("Factorial of %d is %llu\n", num, result);
    }
    return 0; }
```

OUTPUT:

```
Enter a non-negative integer: 7
```

Factorial of 7 is 5040

2. To find the GCD (greatest common divisor) of two given integers:

RECURSIVE FUN:

```
#include <stdio.h>
// Recursive function to find the greatest common divisor (GCD)
int gcd_recursive(int a, int b) {
    if (b == 0) {
        return a;
    }
    return gcd_recursive(b, a % b);
}
int main() {
    int num1, num2;
    printf("Enter two positive integers: ");
    scanf("%d %d", &num1, &num2);
    if (num1 <= 0 || num2 <= 0) {
        printf("GCD is not defined for non-positive numbers.\n");
    } else {
        int result = gcd_recursive(num1, num2);
        printf("GCD of %d and %d is %d\n", num1, num2, result);
    }
    return 0; }
```

NON-RECURSIVE FUN:

```
#include <stdio.h>
// Non-recursive function to find the greatest common divisor (GCD)
```

```
int gcd_non_recursive(int a, int b) {
    int temp;
    while (b != 0) {
        temp = b;
        b = a % b;
       a = temp;  }
    return a;}
int main() {
    int num1, num2;
    printf("Enter two positive integers: ");
    scanf("%d %d", &num1, &num2);
    if (num1 <= 0 || num2 <= 0) {
        printf("GCD is not defined for non-positive numbers.\n");
    } else {
        int result = gcd_non_recursive(num1, num2);
        printf("GCD of %d and %d is %d\n", num1, num2, result);
    }
    return 0;
}
```

OUTPUT:

```
Enter two positive integers: 3 4
 GCD of 3 and 4 is 1
```