



**HYDERABAD INSTITUTE OF TECHNOLOGY AND
MANAGEMENT**

Computer Science and Engineering Department

Programming for Problem Solving

LAB MANUAL

YEAR: 2021-2022

COURSE CODE:

REGULATIONS: H21

CLASS: B.TECH

BRANCH: CSE/CSO/CSD/CSM/CSC/EEE/MECH/ECE

SECTION: I-I SEM

TEAM OF INSTRUCTORS (OPTIONAL)

PREPARED BY: Ms.P.Swathy/Mr Subhani Ali/Mr Nagoor Meeravali



HYDERABAD INSTITUTE OF TECHNOLOGY AND MANAGEMENT
Program Outcomes

PO	Statement
PO1	Engineering knowledge: Apply the knowledge of mathematics, science, Engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
PO2	Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO3	Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
PO4	Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
PO5	Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
PO6	The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
PO7	Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
PO8	Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
PO9	Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO	Statement
PO10	Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
PO11	Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to ones own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
PO12	Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

List of Experiments

1	Week-1: OPERATORS AND EVALUATION OF EXPRESSIONS	7
1.1	Design and develop a flowchart and algorithm to read a number and implement using a C program to check whether the given number is even or odd using ternary operator.	7
1.2	Design and develop a flowchart and algorithm to read two integers and implement using a C program to perform the addition of two numbers without using + operator	7
1.3	c. Develop a C program to evaluate the following arithmetic expressions by reading appropriate input from the standard input device. Understand the priority of operators while evaluating expressions. i. $6*2/(2+1*2/3+6)+8*(8/4)$ ii. $17-8/4*2+3-++z$ iii. $!(x > 10) \&\& (y == 2)$	8
1.4	d. Develop a C program to display the size of various built-in data types in C language	10
2	Week – 2: CONTROL STRUCTURES	12
2.1	a. Design and develop a flowchart and algorithm to read a year as an input and find whether it is leap year or not. Implement a C program for the same and execute for all possible inputs with appropriate messages. Also consider end of the centuries	12
2.2	Design and develop a flowchart and algorithm to find the square root of a given number N. Implement a C program for the same and execute for all possible inputs with appropriate messages. (Note: Don't use library function sqrt(n), Hint: Use Newton-Raphson method to find the square root).	13
2.3	c. Design and develop a flowchart and algorithm to generate a Fibonacci sequence up to a given number N. A Fibonacci sequence is defined as follows: The first and second terms in the sequence are 0 and 1. Subsequent terms are found by adding the preceding two terms in the sequence. Implement a C program for the developed flowchart/algorithm and execute the same to generate the first N terms of the sequence	14
2.4	d. Design and develop a flowchart and algorithm that takes three coefficients (a, b, and c) of a Quadratic equation ($ax^2+bx+c=0$) as input and compute all possible roots. Implement a C program for the developed flowchart/algorithm and execute the same to output the possible roots for a given set of coefficients with appropriate messages.	16
3	Week-3: CONTROL STRUCTURES	20

3.1	a.Design and develop an algorithm to find the reverse of an integer number N and check whether it is PALINDROME or NOT. Implement a C program for the developed algorithm that takes an integer number as input and output the reverse of the same with suitable messages. Ex: N: 2020, Reverse: 0202, Not a Palindrome.	20
3.2	b.Draw the flowchart and write C Program to compute $\sin(x)$ using Taylor series approximation given by $\sin(x) = x - (x^3/3!) + (x^5/5!) - (x^7/7!) + \dots$. Compare the result with the built- in Library function and print both the results with appropriate messages.	21
3.3	c.Design and develop an algorithm and flowchart to read a three digit number and check whether the given number is Armstrong number or not. Write a C program to implement the same and also display the Armstrong numbers between the ranges 1 to 1000.	22
3.4	d. Design and develop an algorithm for evaluating the polynomial $f(x) = a_4x^4 + a_3x^3 + a_2x^2 + a_1x^1 + a_0$, for a given value of x and its coefficients using Horner's method. Implement a C program for the same and execute the program for different sets of values of coefficients and x.	23
4	Arrays	26
4.1	a.Develop, implement and execute a C program to read a list of integers and store it in a single dimensional array. Write a C program to print the second largest integer in a list of integers.	26
4.2	Develop, implement and execute a C program to read a list of integers and store it in a single dimensional array. Write a C program to count and display positive, negative, odd and even numbers in an array	27
4.3	Develop, implement and execute a C program to read a list of integers and store it in a single dimensional array. Write a C program to find the frequency of a particular number in a list of integers	29
4.4	d. Develop, implement and execute a C program that reads two matrices A (m x n) and B (p x q) and Compute the product A and B. Read matrix A and matrix B in row major order respectively. Print both the input matrices and resultant matrix with suitable headings and output should be in matrix format only. Program must check the compatibility of orders of the matrices for multiplication. Report appropriate message in case of incompatibility.	32
5	STRINGS	35

5.1	Develop a user-defined function STRCOPY (str1, str2) to simulate the built-in library function strcpy (str1, str2) that copies a string str2 to another string str1. Write a C program that invokes this function to perform string copying. Also perform the same operation using built-in function	35
5.2	Develop a user-defined function STRCONCT (str1, str2) to simulate the built-in library function strcat (str1, str2) that takes two arguments str1 and str2, concatenates str2 and str1 and stores the result in str1. Write a C program that invokes this function to perform string concatenation. Also perform the same operation using built-in function	37
5.3	Develop a C program that returns a pointer to the first occurrence of the string in a given string using built-in library function strstr(). Example: strstr() function is used to locate first occurrence of the string "test" in the string "This is a test string for testing". Pointer is returned at first occurrence of the string "test".	39
5.4	Develop a C program using the library function strcmp (str1, str2) that compares the string pointed to by str1 to the string pointed to by str2 and returns an integer. Display appropriate messages based on the return values of this function as follows if return value < 0 then it indicates str1 is less than str2. if return value > 0 then it indicates str2 is less than str1. if return value = 0 then it indicates str1 is equal to str2.	40
6	Functions	42
6.1	Design and develop a recursive and non-recursive function FACT (num) to find the factorial of a number, n!, defined by FACT(n) = 1, if n = 0. Otherwise FACT (n) = n * FACT(n-1). Using this function, write a C program to compute the binomial coefficient. Tabulate the results for different values of n and r with suitable messages	42
6.2	Design and develop a recursive function GCD (num1, num2) that accepts two integer arguments. Write a C program that invokes this function to find the greatest common divisor of two given integers.	44
6.3	C.Design and develop a recursive function FIBO (num) that accepts an integer argument. Write a C program that invokes this function to generate the Fibonacci sequence up to num.	46
6.4	d.Design and develop a C function ISPRIME (num) that accepts an integer argument and returns 1 if the argument is prime, a 0 otherwise. Write a C program that invokes this function to generate prime numbers between the given ranges.	48
6.5	e.Design and develop a function REVERSE (str) that accepts a string arguments. Write a C program that invokes this function to find the reverse of a given string.	50

7	Pointers	52
7.1	a. Develop a C program using pointers to compute the sum, mean and standard deviation of all elements stored in an array of n real numbers.	52
7.2	b. Develop a C program to read a list of integers and store it in an array. Then read the array elements using a pointer and print the value along with the memory addresses	53
7.3	Design and develop non-recursive functions input_matrix (matrix, rows, cols) and print_matrix(matrix, rows, cols) that stores integers into a twodimensional array and displays the integers in matrix form. Write a C program to input and print elements of a two dimensional array using pointers and functions.	55
7.4	Develop a C program to store a list of integers in a single dimensional array using dynamic memory allocation (limit will be at run time) using malloc () function. Write a C program to read the elements and print the sum of all elements along with the entered elements. Also use free () function to release the memory.	57
8	Structures and Unions	60
8.1	a. Write a C program that uses functions to perform the following operations: i. Reading a complex number ii. Writing a complex number iii. Addition and subtraction of two complex numbers Note: represent complex number using a structure	60
8.2	b. Write a C program to compute the monthly pay of 100 employees using each employees name, basic pay. The DA is computed as 52% of the basic pay. Gross.salary (basic pay + DA). Print the employees name and gross salary	62
8.3	c. Create a Book structure containing book_id, title, author name and price. Write a C program to pass a structure as a function argument and print the book details.	63
8.4	d. Create a union containing 6 strings: name, home_address, hostel_address, city, state and zip. Write a C program to display your present address	64
9	Additional Programs	66
9.1	Write a C program to read in two numbers, x and n, and then compute the sum of this geometric progression: $1+x+x^2+x^3+\dots+x^n$. 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 $n \leq 0$, then go back and read in the next pair of numbers of without computing the sum. Are any values of x also illegal? If so, test for them too.	66

9.2	b.Develop a C program to find the 2's complement of a given binary number. 2's complement 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.	67
9.3	c.Develop a C program to convert a Roman numeral to its decimal equivalent. E.g. check for the inputs - Roman number IX is equivalent to 9 and Roman number XI is equivalent to 11	69
10 Preprocessor Directives		71
10.1	a.Define a macro with one parameter to compute the volume of a sphere. Write a C program using this macro to compute the volume for spheres of radius 5, 10 and 15meters	71
10.2	b.Define a macro that receives an array and the number of elements in the array as arguments. Write a C program for using this macro to print the elements of the array.	72
10.3	Write symbolic constants for the binary arithmetic operators +, -, *, and /. Write a C program to illustrate the use of these symbolic constants	73
11 Files		75
11.1	a.Create an employee file employee.txt and write 5 records having employee name, designation, salary, branch and city. Develop a C program to display the contents of employee.txt file.	75
11.2	b.Create a studentolddata.txt file containing student name, roll no, branch, section, address. Develop a C program to copy the contents of studentolddata.txt file to another file studentnew-data.txt.	76
11.3	c. Develop a C program to create a text file info.txt to store the information given below. Implement using a C program to count the number of words and characters in the file info.txt.	77
11.4	d.Given two university information files "studentname.txt" and "roll_number.txt" that contains students Name and Roll numbers respectively. Write a C program to create a new file called "output.txt" and copy the content of files "studentname.txt" and "roll_number.txt" into output file. Display the contents of output file "output.txt" on to the screen.	78
12 Command Line Arguments		80
12.1	a.Develop a C program to read a set of arguments and display all arguments given through command line.	80
12.2	b.Develop a C program to read a file at command line argument and display the contents of the file.	80
12.3	Develop a C program to read N integers and find the sum of N integer numbers using command line arguments.	81

12.4 d. Develop a C program to read three integers and find the largest
integer among three using command line argument. 82

1 Week-1:OPERATORS AND EVALUATION OF EXPRESSIONS

- 1.1 Design and develop a flowchart and algorithm to read a number and implement using a C program to check whether the given number is even or odd using ternary operator.

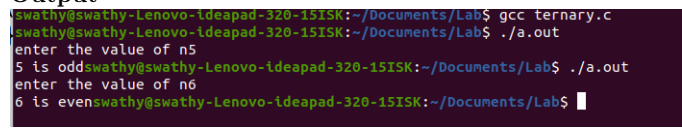
Algorithm

1. Start the program.
2. Read / input the number n.
3. $n \% 2 == 0$? number is even.: number is odd.
4. display the output.
5. Stop the program.

Source Code

```
#include <stdio.h>
int main()
{
    int n;
    printf("enter the value of n");
    scanf("%d",&n);
    (n%2==0)?printf("%d is even",n):printf("%d is odd",n);
    return 0;
}
```

Output



```
swathy@swathy-Lenovo-Ideapad-320-15ISK:~/Documents/Lab$ gcc ternary.c
swathy@swathy-Lenovo-Ideapad-320-15ISK:~/Documents/Lab$ ./a.out
enter the value of n5
5 is oddswathy@swathy-Lenovo-Ideapad-320-15ISK:~/Documents/Lab$ ./a.out
enter the value of n6
6 is evenswathy@swathy-Lenovo-Ideapad-320-15ISK:~/Documents/Lab$
```

- 1.2 Design and develop a flowchart and algorithm to read two integers and implement using a C program to perform the addition of two numbers without using + operator

Algorithm

1. start

2. read two integers a,b
3. $c=a-b-1$
4. Display c
5. stop

Source Code

```
#include <stdio.h>
int main()
{
    int x,y,sum;
    printf("Enter the value for x and y");
    scanf("%d%d",&x,&y);
    sum=x-y-1;
    printf("Sum=%d",sum);
    return 0;
}
```

Output

```
swathy@swathy-Lenovo-ideapad-320-15ISK:~$ cd Documents
swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents$ cd Lab
swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab$ gcc sum.c
swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab$ ./a.out
Enter the value for x and y 4 5
Sum=9swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab$ █
```

- 1.3 c. Develop a C program to evaluate the following arithmetic expressions by reading appropriate input from the standard input device. Understand the priority of operators while evaluating expressions.
- i. $6*2/(2+1*2/3+6)+8*(8/4)$
 - ii. $17-8/4*2+3-++z$
 - iii. $!(x > 10) \&\& (y == 2)$

Evaluation

i. $6*2/(2+2/3+6)+8*(8/4)$
 $6*2/(2+0+6)+8*(8/4)$
 $6*2/8+8*(8/4)$
 $6*2/8+8*2$
 $12/8+8*2$
 $1+8*2$

1+16
17

ii. $17 - 8 / 4 * 2 + 3 - ++z$
 $17 - 8 / 4 * 2 + 3 - ++z$ let $z=2$ $z=z+1$
 $17 - 2 * 2 + 3 - 3$
 $17 - 4 + 3 - 3$
 $13 + 3 - 23$ 16-3
13

iii. $! (x > 10) \&\& (y == 2)$
first it evaluates $(x > 10)$
then $!$ is applied to the evaluated condition
if it is true then it evaluates $(y == 2)$
if both are true then the equation is true
first it evaluates $(x > 10)$
then $!$ is applied to the evaluated condition
if it is false then it evaluates to false without verifying the condition $(y == 2)$

Source Code

```
#include <stdio.h>
int main()
{
    int res,res1,res2,x,y,z;
    res=6*2/(2+1 * 2/3 +6) +8 * (8/4);
    printf("res=%d",res);
    printf("enter the value of z");
    scanf("%d",&z);
    res1=17-8/4*2+3-++z;
    printf("res1=%d",res1);
    printf("enter value for x and y");
    scanf("%d%d",&x,&y);
    res2=! (x > 10) && (y==2);
    printf("res2=%d",res2);
    return 0;
}
```

Output

```
swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week1$ gcc evaluation.c
swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week1$ ./a.out
res=17enter the value of z2
res1=13enter value for x and y11 2
res2=0swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week1$ █
swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week1$ ./a.out
res=17enter the value of z2
res1=13enter value for x and y2 2
res2=1swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week1$ █
```

1.4 d. Develop a C program to display the size of various built-in data types in C language

Source Code

```
#include <stdio.h>
int main()
{
    printf("integer=%lu", sizeof(int));
    printf("\nunsigned integer=%lu", sizeof(unsigned int));
    printf("\nshort integer=%lu", sizeof(short int));
    printf("\nlong integer=%lu", sizeof(long int));
    printf("\n float=%lu", sizeof(float));
    printf("\n double=%lu", sizeof(double));
    printf("\n long double=%lu", sizeof(long double));
    printf("\n character=%lu", sizeof(char));
    return 0;
}
```

Output

```
swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week1$ gcc sizeofvariables.c
swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week1$ ./a.out
integer=4
unsigned integer=4
short integer=2
long integer=8
float=4
double=8
long double=16
character=1swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week1$
```

Viva Questions

1. Find the output of the following code?

```
void main()
{
    int i=10;
    i = !i > 14;
    printf("i=%d", i);
}
```

2. Find the output of the following code?

```
void main()
{
```

```

    int i=0, j=1, k=2,m;
    m = i++ || j++ || k++;
    printf("%3d%3d%3d%3d", m, i, j, k);
}

```

3. Find the output of the following code?

```

void main()
{
    int main=3;
    printf("%d", main);
}

```

4. Find the output of the following code?

```

void main()
{
    int k;
    k = 'a' > 60;
    printf("%d", k);
}

```

5. Find the output of the following code?

```

void main()
{
    int const k = 5;
    k++;
    printf("k is %d", k);
}

```

2 Week – 2: CONTROL STRUCTURES

- 2.1 a.Design and develop a flowchart and algorithm to read a year as an input and find whether it is leap year or not. Implement a C program for the same and execute for all possible inputs with appropriate messages. Also consider end of the centuries

Algorithm

1. Start
2. Read year
3. If the year is divisible by 4 then go to Step 4 else go to Step 7
4. If the year is divisible by 100 then go to Step 5 else go to Step 6
5. If the year is divisible by 400 then go to Step 6 else go to Step 7
6. Print "Leap year"
7. Print "Not a leap year"
8. Stop

Source Code

```
#include <stdio.h>
int main() {
    int year;
    printf("Enter a year: ");
    scanf("%d", &year);
    if (year % 400 == 0) {
        printf("%d is a leap year.", year);
    }
    else if (year % 100 == 0) {
        printf("%d is not a leap year.", year);
    }
    else if (year % 4 == 0) {
        printf("%d is a leap year.", year);
    }
    else {
        printf("%d is not a leap year.", year);
    }
}
```

Output

```
swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week2$ gcc leapyear.c
swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week2$ ./a.out
Enter a year: 2000
2000 is a leap year. swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week2$ ./a.out
Enter a year: 2022
2022 is not a leap year. swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week2$ █
```

2.2 Design and develop a flowchart and algorithm to find the square root of a given number N. Implement a C program for the same and execute for all possible inputs with appropriate messages. (Note: Don't use library function `sqrt(n)`, Hint: Use Newton-Raphson method to find the square root).

Algorithm

Input: initial x, func(x), derivFunc(x)

Output: Root of Func()

1. Assign X to the N itself.
2. Now, start a loop and keep calculating the root which will surely move towards the correct square root of N.
3. Check for the difference between the assumed X and calculated root, if not yet inside tolerance then update root and continue.
4. If the calculated root comes inside the tolerance allowed then break out of the loop.
5. Print the root

Source Code

```
#include <stdio.h>
float absolute(float num)
{
    if(num < 0){
        num = -num;
    }
    return num;
}
float square_root(int x)
{
    const float difference = 0.00001;
    float guess = 1.0;
    while(absolute(guess * guess - x) >= difference){
        guess = (x/guess + guess)/2.0;
    }
}
```



```

    }
    return guess;
}

int main()
{
    int number;
    float root;
    printf("Enter a number: ");
    scanf("%i", &number);
    root = square_root(number);
    printf("The square root of %i is: %f", number, root);
    return 0;
}

```

Output

```

swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week2$ gcc Nraphson.c
swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week2$ ./a.out
Enter a number: 16
The square root of 16 is: 4.000000swathy@swathy-Lenovo-ideapad-320-15ISK:~/Do

```

- 2.3 c.Design and develop a flowchart and algorithm to generate a Fibonacci sequence up to a given number N. A Fibonacci sequence is defined as follows: The first and second terms in the sequence are 0 and 1. Subsequent terms are found by adding the preceding two terms in the sequence. Implement a C program for the developed flowchart/algorithm and execute the same to generate the first N terms of the sequence

Algorithm

1. Start.
2. Declare variables i, a,b , show.
3. Initialize the variables, a=0, b=1, and show =0.
4. Enter the number of terms of Fibonacci series to be printed.
5. Print First two terms of series.
6. Use loop for the following steps. -i show=a+b. -i a=b. -i b=show. -i increase value of i each time by 1. ...
7. End.

Source Code

```
include<stdio.h>
int main()
{
    int a=0,b=1,c,n;
    printf("enter the value of n to print fibonacci series upto n");
    scanf("%d",&n);
    printf("%d\t%d",a,b);
    do
    {
        c=a+b;
        a=b;
        b=c;
        printf("\t%d\t",c);
    }while(c<=n);
    return 0;
}
```

Output

```
swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week2$ ./a.out
enter the value of n to print fibonacci series upto n6
0      1      1      2      3      5      8
swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week2$
```

- 2.4 d.Design and develop a flowchart and algorithm that takes three coefficients (a, b, and c) of a Quadratic equation ($ax^2+bx+c=0$) as input and compute all possible roots. Implement a C program for the developed flowchart/algorithm and execute the same to output the possible roots for a given set of coefficients with appropriate messages.

Algorithm

1. Start
2. Read the coefficients of the equation, a, b and c from the user.
3. Calculate discriminant = $(b * b) - (4 * a * c)$
4. If discriminant ≥ 0 :
 - (a) Calculate root1 = $(-b + \text{sqrt}(\text{discriminant})) / (2 * a)$
 - (b) Calculate root2 = $(-b - \text{sqrt}(\text{discriminant})) / (2 * a)$
 - (c) Display "Roots are real and different"
 - (d) Display root1 and root2
5. Else if discriminant = 0:
 - (a) Calculate root1 = $-b / (2 * a)$
 - (b) root2 = root1
 - (c) Display "Root are real and equal"
 - (d) Display root1 and root2
6. Else:
 - (a) Calculate real = $-b / (2 * a)$
 - (b) Calculate imaginary = $\text{sqrt}(-\text{discriminant}) / (2 * a)$
 - (c) Display "Roots are imaginary"
 - (d) Display real, "±", imaginary, "i"
7. Stop

Source Code

```
#include <math.h>
#include <stdio.h>
int main() {
    double a, b, c, discriminant, root1, root2, realPart, imagPart;
    printf("Enter coefficients a, b and c: ");
    scanf("%lf %lf %lf", &a, &b, &c);
```

```

discriminant = b * b - 4 * a * c;

if (discriminant > 0) {
    root1 = (-b + sqrt(discriminant)) / (2 * a);
    root2 = (-b - sqrt(discriminant)) / (2 * a);
    printf("root1 = %.2lf and root2 = %.2lf", root1, root2);
}

else if (discriminant == 0) {
    root1 = root2 = -b / (2 * a);
    printf("root1 = root2 = %.2lf;", root1);
}

else {
    realPart = -b / (2 * a);
    imagPart = sqrt(-discriminant) / (2 * a);
    printf("root1 = %.2lf+%.2lfi and root2 = %.2f-%.2fi", realPart, imagPart, realPart,
    imagPart);
}

return 0;
}

```

Output

```

swathy@swathy-Lenovo-ideapad-320-15ISK:~/Do
Enter coefficients a, b and c: 1 2 1
root1 = root2 = -1.00;swathy@swathy-Lenovo-
Enter coefficients a, b and c: 2 3 1
root1 = -0.50 and root2 = -1.00swathy@swath

```

Viva Questions

What is the output of the following code?

```

#include <stdio.h>
void main()
{
    int i=1;
    for(;;i++) printf("%d",i);
}

```

Can you write for loop without body, if yes then what is the output of the following code?

```
#include <stdio.h>
int main()
{
    int i;
    for(i=0;i<=10;i++);
    printf("%d",i);
    return 0;
}
```

What is the output of the following program ?

```
#include <stdio.h>
int main()
{
    int i,j=2;
    for(i=0;j>=0,i<=5;i++)
    {
        printf("%d ",i+j);
        j--;
    }
    return 0;
}
```

What is the output of the following program?

```
#include <stdio.h>
int main()
{
    int i;
    for(;i=0,i<=3 ;i++)
    {
        printf("%d ",i);
    }
    return 0;
}
```

What is the output of the following program?

```
#include <stdio.h>
int main()
{
    int i;
    for(i=0;-5 ;i++)
    {
        printf("%d ",i);
        if(i==3)
            break;
    }
}
```

```
break;  
}  
return 0;  
}
```

3 Week-3: CONTROL STRUCTURES

- 3.1 a.Design and develop an algorithm to find the reverse of an integer number N and check whether it is PALINDROME or NOT. Implement a C program for the developed algorithm that takes an integer number as input and output the reverse of the same with suitable messages. Ex: N: 2020, Reverse: 0202, Not a Palindrome.

Algorithm

1. Take the number which you have to reverse as the input.
2. Obtain its quotient and remainder.
3. Multiply the separate variable with 10 and add the obtained remainder to it.
4. Do step 2 again for the quotient and step 3 for the remainder obtained in step 4.
5. Repeat the process until quotient becomes zero.
6. When it becomes zero, check if the reversed number is equal to original number or not.
7. Print the output and exit.

Source Code

```
#include <stdio.h>
void main()
{
    int num, temp, remainder, reverse = 0;
    printf("Enter an integer \n");
    scanf("%d", &num);
    temp = num;
    while (num > 0)
    {
        remainder = num % 10;
        reverse = reverse * 10 + remainder;
        num /= 10;
    }
    printf("Given number is = %d\n", temp);
    printf("Its reverse is = %d\n", reverse);
    if (temp == reverse)
        printf("Number is a palindrome \n");
}
```

```

        else
            printf("Number is not a palindrome \n");
    }

```

Output

```

swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week3$ ./a.out
Enter an integer
1001
Given number is = 1001
Its reverse is = 1001
Number is a palindrome
swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week3$ ./a.out
Enter an integer
1021
Given number is = 1021
Its reverse is = 1201
Number is not a palindrome

```

- 3.2 b. Draw the flowchart and write C Program to compute $\sin(x)$ using Taylor series approximation given by $\sin(x) = x - (x^3/3!) + (x^5/5!) - (x^7/7!) + \dots$. Compare the result with the built-in Library function and print both the results with appropriate messages.

Source Code

```

#include <stdio.h>
#include <math.h>
int fac(int x);

void main()
{
    float x,Q,sum=0;
    int i,j,limit;
    printf("Enter the value of x of sinx series: ");
    scanf("%f",&x);

    printf("Enter the limit upto which you want to expand the series: ");
    scanf("%d",&limit);

    Q=x;
    x = x*(3.1415/180);

    for(i=1,j=1;i<=limit;i++,j=j+2)
    {
        if(i%2!=0)

```



```

{
sum=sum+pow(x,j)/fac(j);
}
else
sum=sum-pow(x,j)/fac(j);
}
printf("Sin(%0.1f): %f",Q,sum);
}

int fac(int x)
{
int i,fac=1;
for(i=1;i<=x;i++)
fac=fac*i;
return fac;
}

```

Output

```

swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week3$ gcc taylor.c -lm
swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week3$ ./a.out
Enter the value of x of sinx series: 0.2
Enter the limit upto which you want to expand the series: 7
Sin(0.2): 0.003491swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week3$

```

3.3 c.Design and develop an algorithm and flowchart to read a three digit number and check whether the given number is Armstrong number or not. Write a C program to implement the same and also display the Armstrong numbers between the ranges 1 to 1000.

Algorithm

1. Start
2. Declare Variable sum, temp, num
3. Read num from User
4. Initialize Variable sum=0 and temp=num
5. Repeat Until num_i=0
 - (a) sum=sum + cube of last digit i.e [(num
 - (b) num=num/10
6. IF sum==temp Print "Armstrong Number" ELSE Print "Not Armstrong Number"
7. Stop

Source Code

```
#include <stdio.h>
int main() {
    int num, originalNum, remainder, result = 0;
    printf("Enter a three-digit integer: ");
    scanf("%d", &num);
    originalNum = num;
    while (originalNum != 0) {
        // remainder contains the last digit
        remainder = originalNum % 10;
        result += remainder * remainder * remainder;
        // removing last digit from the original number
        originalNum /= 10;
    }
    if (result == num)
        printf("%d is an Armstrong number.", num);
    else
        printf("%d is not an Armstrong number.", num);
    return 0;
}
```

Output

```
swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week3$ gcc armstrong.c
swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week3$ ./a.out
Enter a three-digit integer: 345
345 is not an Armstrong number.swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week3$ ./a.out
Enter a three-digit integer: 153
153 is an Armstrong number.swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week3$
```

- 3.4 d. Design and develop an algorithm for evaluating the polynomial $f(x) = a_4x^4 + a_3x^3 + a_2x^2 + a_1x^1 + a_0$, for a given value of x and its coefficients using Horner's method. Implement a C program for the same and execute the program for different sets of values of coefficients and x .

Algorithm

1. Start
2. Input the degree and value of x
3. Input Coefficients , upper limit is provided by the value of degree
4. $i = \text{degree}$, $\text{sum} = 0$

5. $sum = (sum + a_i) * x$
6. decrement i by 1
7. if ($i < 0$) goto step 5
8. add a_0 to the final sum
9. display sum
10. stop

Source Code

```
/* C Program to Evaluate Polynomial using Horner's method */

#include <stdio.h>
int main()
{
    float a[100],sum=0,x;
    int n,i;

    printf("\nEnter degree of the polynomial X :: ");
    scanf("%d",&n);
    printf("\nEnter coefficient's of the polynomial X :: \n");
    for(i=n;i>=0;i--)
    {
        printf("\nEnter Coefficient of [ X^%d ] :: ",i);
        scanf("%f",&a[i]);
    }
    printf("\nEnter the value of X :: ");
    scanf("%f",&x);
    for(i=n;i>0;i--)
    {
        sum=(sum+a[i])*x;
    }
    sum=sum+a[0];
    printf("\nValue of the polynomial is = [ %f ]\n",sum);
    return 0;
}
```

Output

```
swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week3$ gcc horner.c
swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week3$ ./a.out

Enter degree of the polynomial X :: 2

Enter coefficient's of the polynomial X ::

Enter Coefficient of [ X^2 ] :: 2

Enter Coefficient of [ X^1 ] :: 1

Enter Coefficient of [ X^0 ] :: 1

Enter the value of X :: 2

Value of the polynomial is = [ 11.000000 ]
```

Viva Questions

1. How can we use for loops when the number of iterations are not known?
2. Compare in terms of their functions, between while and do...while loops?
3. In an exit-controlled loop, if the body is executed n times, then the test condition is evaluated how many times?
4. Do you think in a for loop expression, the starting value of the control variable must be less than its ending value?
5. Do you think the use of continue statement is considered as unstructured programming?
6. Write a for statement to print each of the sequences of integers 1, 2, 4, 8, 16, 32?
7. Change the following for loop to while loop
for(m = 1; m <= 10; m = m + 1)
printf(m)

4 Arrays

- 4.1 a. Develop, implement and execute a C program to read a list of integers and store it in a single dimensional array. Write a C program to print the second largest integer in a list of integers.

Algorithm

1. Let inputArray is an array of length N, and maxElement and secondMaxElement are two integer variables to store maximum and second maximum element of array.
2. We need atleast two elements in array to find second largest element in array.
3. Initialize maxElement and secondMaxElement with INT_MIN. INT_MIN is the minimum value that can be represented by a signed int. INT_MIN macro is defined in limits.h header file.
4. Traverse inputArray from first element to last element.
5. if(current_element > maxElement) then secondMaxElement = maxElement; and maxElement = current_element; because if we found an element which is greater than current maximum element then current maximum element will become second maximum element.
6. Else If(current_element > secondMaxElement) then secondMaxElement = current_element; this mean current_element is greater than secondMaxElement but smaller than maxElement.

Source Code

```
#include <stdio.h>
#include <limits.h> // For INT_MIN

#define MAX_SIZE 1000 // Maximum array size

int main()
{
    int arr[MAX_SIZE], size, i;
    int max1, max2;
    printf("Enter size of the array (1-1000): ");
    scanf("%d", &size);
    printf("Enter elements in the array: ");
    for(i=0; i<size; i++)
    {
        scanf("%d", &arr[i]);
    }
}
```

```

max1 = max2 = INT_MIN;
for(i=0; i<size; i++)
{
    if(arr[i] > max1)
    {
        max2 = max1;
        max1 = arr[i];
    }
    else if(arr[i] > max2 && arr[i] < max1)
    {
        max2 = arr[i];
    }
}
printf("Second largest = %d", max2);
return 0;
}

```

Output

```

swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week4$ gcc secondlarge.c
swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week4$ ./a.out
Enter size of the array (1-1000): 4
Enter elements in the array: 56
45
67
34
Second largest = 56swathy@swathy-Lenovo-ideapad-320-15ISK:~/Documents/Lab/Week4$

```

4.2 Develop, implement and execute a C program to read a list of integers and store it in a single dimensional array. Write a C program to count and display positive, negative, odd and even numbers in an array

Algorithm

1. Declare and read variables n, i
2. Read elements into an array
3. If(a[i] > 0) then display number is positive
4. If(a[i]%2==0) then display number is even
5. If(a[i] < 0) then display number is negative
6. If(a[i]%2!=0) then display number is odd

Source Code

```
#include <stdio.h>

#define N 10

int main()
{
    int a[N], i, pos = 0, neg = 0, even = 0, odd = 0, zero = 0;

    printf("Enter %d integer numbers\n", N);
    for(i = 0; i < N; i++)
    {
        scanf("%d", &a[i]);
        if(a[i] == 0)
        {
            zero++;
        }
        else if(a[i] > 0)
            pos++;
        else
            neg++;

        if(a[i] == 0)
        {
        }
        else if(a[i] % 2 == 0)
            even++;
        else
            odd++;
    }

    printf("\nPositive: %d\n", pos);
    printf("Negative: %d\n", neg);
    printf("Even: %d\n", even);
    printf("Odd: %d\n", odd);
    printf("Zero: %d\n", zero);

    return 0;
}
```

Output

```
swathy@swathy-Lenovo-ideapa  
swathy@swathy-Lenovo-ideapa  
Enter 10 integer numbers  
3  
4  
0  
-1  
-4  
7  
6  
0  
-5  
0  
  
Positive: 4  
Negative: 3  
Even: 3  
Odd: 4  
Zero: 3
```

- 4.3 Develop, implement and execute a C program to read a list of integers and store it in a single dimensional array. Write a C program to find the frequency of a particular number in a list of integers

Algorithm

1. START
2. INITIALIZE arr[] =1, 2, 8, 3, 2, 2, 5, 1 .
3. length = sizeof(arr)/sizeof(arr[0])
4. DEFINE fr[length].
5. SET visited = -1.
6. SET i= 0. REPEAT STEP 7 to STEP 12 UNTIL i=length.
7. SET count = 1

8. SET j =0. REPEAT STEP 9 and STEP 10 UNTIL j=length.
9. if(arr[i]==arr[j]) then count++ fr[j] =visited
10. j= j+1.
11. if(fr[i]!=visited) then fr[i]=count
12. i=i+1
13. PRINT "_____"
14. PRINT "Element | Frequency"
15. PRINT "_____"
16. SET i=0. REPEAT STEP 17 to STEP 18 UNTIL i<length.
17. if(fr[i]!=visited) then PRINT arr[i] PRINT | PRINT fr[i]
18. i=i+1.
19. PRINT "_____"
20. RETURN 0.
21. END

Source code

```
#include <stdio.h>

int main()
{
    //Initialize array
    int arr[10],n,fr[10],visited = -1,i,j,length;
    printf("Enter the size of array");
    scanf("%d",&n);
    printf("Enter the array elements");
    for(i=0;i<n;i++)
    {
        scanf("%d",&arr[i]);
    }
    //Calculate length of array arr
    //length = sizeof(arr)/sizeof(arr[0]);
    fr[n] =0;
    for(int i = 0; i < n; i++){
        int count = 1;
        for(int j = i+1; j < n; j++){
            if(arr[i] == arr[j]){
                count++;
            }
        }
        fr[i] = count;
    }
    //Print the frequency of each element
    for(i=0;i<n;i++){
        printf("%d | %d\n",arr[i],fr[i]);
    }
}
```

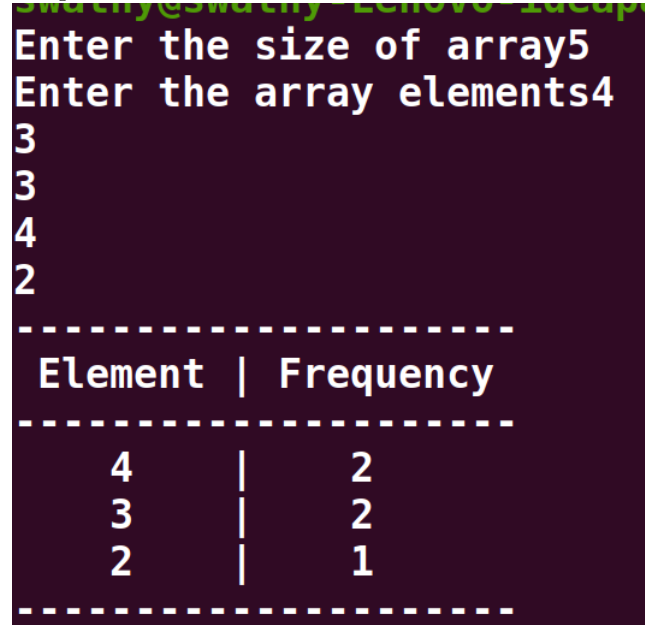
```

        //To avoid counting same element again
        fr[j] = visited;
    }
}
if(fr[i] != visited)
    fr[i] = count;
}

//Displays the frequency of each element present in array
printf("-----\n");
printf(" Element | Frequency\n");
printf("-----\n");
for(int i = 0; i < n; i++){
    if(fr[i] != visited){
        printf("    %d", arr[i]);
        printf(" | ");
        printf(" %d\n", fr[i]);
    }
}
printf("-----\n");
return 0;
}

```

Output



```

swathy@swathy-Lenovo-IdeaPad:
Enter the size of array5
Enter the array elements4
3
3
4
2
-----
Element | Frequency
-----
    4   |    2
    3   |    2
    2   |    1
-----

```

- 4.4 d. Develop, implement and execute a C program that reads two matrices A (m x n) and B (p x q) and Compute the product A and B. Read matrix A and matrix B in row major order respectively. Print both the input matrices and resultant matrix with suitable headings and output should be in matrix format only. Program must check the compatibility of orders of the matrices for multiplication. Report appropriate message in case of incompatibility.

Source Code

```
#include <stdio.h>

int main()
{
    int m, n, p, q, c, d, k, sum = 0;
    int first[10][10], second[10][10], multiply[10][10];

    printf("Enter number of rows and columns of first matrix\n");
    scanf("%d%d", &m, &n);
    printf("Enter elements of first matrix\n");

    for (c = 0; c < m; c++)
        for (d = 0; d < n; d++)
            scanf("%d", &first[c][d]);

    printf("Enter number of rows and columns of second matrix\n");
    scanf("%d%d", &p, &q);

    if (n != p)
        printf("The multiplication isn't possible.\n");
    else
    {
        printf("Enter elements of second matrix\n");

        for (c = 0; c < p; c++)
            for (d = 0; d < q; d++)
                scanf("%d", &second[c][d]);

        for (c = 0; c < m; c++) {
            for (d = 0; d < q; d++) {
                for (k = 0; k < p; k++) {
                    sum = sum + first[c][k]*second[k][d];
                }
            }
        }
    }
}
```

```

        multiply[c][d] = sum;
        sum = 0;
    }
}

printf("Product of the matrices:\n");

for (c = 0; c < m; c++) {
    for (d = 0; d < q; d++)
        printf("%d\t", multiply[c][d]);

    printf("\n");
}

return 0;
}

```

Output

```

Enter number of rows and columns of first matrix
2
2
Enter elements of first matrix
2 2 2 2
Enter number of rows and columns of second matrix
2 2
Enter elements of second matrix
2 2 2 2
Product of the matrices:
8      8
8      8

```

Viva Questions

What will be the output of the following code ?

```

void main()
{
    int a[5]={3,5,6,2,7};
    int i=3;
    if(a[i]==a[i+3])
        printf("true");
    else
        printf("false");
}

```

What will be the output of the following code ?

```
void main()
{
    int a[5] = {5,7,3,1,2};
    a[2]=a[1];
    a[1]=a[3];
    a[2]=a[2]+a[3];
    printf("%d\t%d", a[1],a[2]);
}
```

Distinguish Lvalue and Rvalue of an array element? Write the output of the following code?

```
void main()
{
    int a[3][2] = {10, 20, 30, 40, 50, 60};
    printf("%d", a[0][4]);
}
```

Which of the following multi-dimensional array declaration is correct for realizing a 2x3 matrix?

- a. int m[2][3]
- b. int m[3][2]
- c. int m[3],m[2]

5 STRINGS

- 5.1 Develop a user-defined function STRCOPY (str1, str2) to simulate the built-in library function strcpy (str1, str2) that copies a string str2 to another string str1. Write a C program that invokes this function to perform string copying. Also perform the same operation using built-in function

Algorithm

1. start
 2. Declare two character arrays
 3. Read the source string
 4. Iterate the loop from 0 index to "
 5. Copy character by character until ' ' is reached
 6. Assign the null string to the second character array to which the string was copied
 7. Display the second character array
 8. stop
-
1. start
 2. declare two character arrays i.e;s1,s2
 3. Initialize the first array i.e;s1
 4. call the library function strcpy(s2,s1);
 5. Display s2
 6. stop

Source Code

```
/*copy one string to another without using string function*/
#include <stdio.h>

/* Function prototype */
void STRCOPY(char str2[30], char str1[30]);

/* Main function */
int main()
{
```

```

char str1[30], str2[30];
int i;

printf("Enter string:\n");
gets(str1);

STRCOPY(str2, str1);

printf("Copied string is: %s", str2);

return 0;
}

/* Function Definition*/
void STRCOPY(char str2[30], char str1[30])
{
    int i;
    for(i=0;str1[i]!='\0';i++)
    {
        str2[i] = str1[i];
    }
    str2[i] = '\0';
}

/* Copy one string to another using Library function*/
#include <stdio.h>
#include <string.h>
int main()
{
    char s1[1000],s2[1000];
    int i;

    printf("Enter any string: ");
    gets(s1);
    strcpy(s2,s1);
    printf("original string s1='%s'\n",s1);
    printf("copied string   s2='%s'",s2);
    return 0;
}

```

Output

```

Enter string:
Welcome to C ↵
Copied string is: Welcome to C

```

```
1 Enter any string: hello c beginners
2 original string s1='hello c beginners'
3 copied string s2='hello c beginners'
```

- 5.2 Develop a user-defined function STRCONCT (str1, str2) to simulate the built-in library function strcat (str1, str2) that takes two arguments str1 and str2, concatenates str2 and str1 and stores the result in str1. Write a C program that invokes this function to perform string concatenation. Also perform the same operation using built-in function

Algorithm

1. start
2. Declare two strings
3. Read the two strings using gets()
4. compute the length of the first string
5. copy the second string from 0th index to " " from the length index position of string1
6. add the null character " " to the end of string1
7. display the string1
8. stop

Source Code

```
/* Concatenate two strings without using strcat() */
#include <stdio.h>
#include <string.h>

void STRCONCT(char *s1, char *s2)
{
    int i;
    int j=strlen(s2);

    for(i=0;s2[i];i++)
    {
        s1[i+j]=s2[i];
    }
}
```



```

        s1[i+j]='\0';

    }
    int main()
    {

        char s1[1000],s2[1000];

        printf("Enter string1: ");
        gets(s1);
        printf("Enter string2: ");
        gets(s2);
        STRCONCT(s1,s2);
        printf("combined two strings = '%s'\n",s1);

        return 0;

    }
    /* Concatenate two string using strcat()*/
    #include <stdio.h>
    #include <string.h>
    i

    int main()
    {
        char s1[1000],s2[1000];

        printf("Enter string1: ");
        gets(s1);
        printf("Enter string2: ");
        gets(s2);
        strcat(s1,s2);

        printf("combined two strings = '%s'\n",s1);

        return 0;

    }

```

Output

```

1 Enter string1: hello
2 Enter string2: world
3 combined two strings = 'helloworld'

```

```
1 Enter string1: welcome to
2 Enter string2: c beginners
3 combined two strings = 'welcome to c beginners'
```

- 5.3 Develop a C program that returns a pointer to the first occurrence of the string in a given string using built-in library function `strstr()`. Example: `strstr()` function is used to locate first occurrence of the string “test” in the string “This is a test string for testing”. Pointer is returned at first occurrence of the string “test”.

Algorithm

1. start
2. declare a string and substring
3. read the string and substring
4. invoke `strstr()` with the arguments of string and substring
5. if substring exists in string it returns the string else displays substring is not found
6. stop

Source Code

```
#include <stdio.h>
#include <string.h>
int main ()
{
    char string[55] ="This is a test string for testing";
    char *p;
    p = strstr (string,"test");
    if(p)
    {
        printf("string found\n" );
        printf ("First occurrence of string \"test\" in \"%s\" is\"
            \"%s\"",string, p);
    }
    else printf("string not found\n" );
    return 0;
}
```

Output

```
string found
First occurrence of string "test" in "This is a test string for testing" is "test string for testing"
```

- 5.4 Develop a C program using the library function `strcmp` (`str1`, `str2`) that compares the string pointed to by `str1` to the string pointed to by `str2` and returns an integer. Display appropriate messages based on the return values of this function as follows
- if return value < 0 then it indicates `str1` is less than `str2`.
 - if return value > 0 then it indicates `str2` is less than `str1`.
 - if return value $= 0$ then it indicates `str1` is equal to `str2`.

Algorithm 1) The function `strcmp(String1, String2)` is the string library function to compare two strings. If both the strings are equal then it returns 1 otherwise it returns 0.

2) The function `strcmp(String1, String2)` is available in the `string.h` header file.

Source Code

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

int main() {
    char str1[10], str2[10], str3[10];
    int result;
    printf("enter string1");
    gets(str1);
    printf("enter string2");
    gets(str2);
    printf("enter string3");
    gets(str3);

    // comparing strings str1 and str2
    result = strcmp(str1, str2);
    printf("strcmp(str1, str2) = %d\n", result);

    // comparing strings str1 and str3
    result = strcmp(str1, str3);
    printf("strcmp(str1, str3) = %d\n", result);
}
```

```
    return 0;  
}
```

Output

```
enter string1Hello  
enter string2Hello  
enter string3hello  
strcmp(str1, str2) = 0  
strcmp(str1, str3) = -32
```

Viva Questions

1. Which command is used to combine the two strings?
2. Which command is used to copy the strings?
3. What is the difference between string copy strcpy() and memory copy memcpy() ?
4. How can you copy just a portion of a string?
5. Using which function you can find a substring from a string?
6. Which function is used to compare two strings?
7. How can you remove trailing spaces from a string?
8. How can you remove leading spaces from a string?

6 Functions

- 6.1 Design and develop a recursive and non-recursive function FACT (num) to find the factorial of a number, n!, defined by $\text{FACT}(n) = 1$, if $n = 0$. Otherwise $\text{FACT}(n) = n * \text{FACT}(n-1)$. Using this function, write a C program to compute the binomial coefficient. Tabulate the results for different values of n and r with suitable messages

Algorithm

Factorial of a given number by using Recursive function

1. Start
2. Read a number N
3. Call a function factorial(N) by passing the values of N
4. If $N=1$ then it returns 1 as the factorial
5. Otherwise it calculates the factorial $f = N * \text{factorial}(N-1)$ by calling the same function again and again
6. Display the factorial of number N
7. Stop

Factorial of a given number by using Non-Recursive function

1. Start
2. Read a number N
3. Initialize fact =1
4. Calculate $\text{fact} = \text{fact} * N$ using a loop and decrement N value till $N = 1$
5. Display the factorial of number N
6. Stop

Source Code

```
/*Factorial of a given number by using Recursive function*/
#include<stdio.h>
int fact(int n)
{
    if(n==0)
        return 1;
    else
```

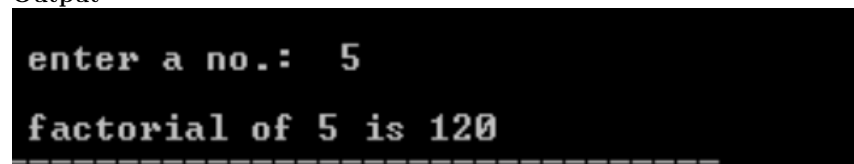
```

return n*fact(n-1);
}
void main()
{
int n,f;
printf("\n enter a no.: ");
scanf("%d",&n);
f=fact(n);
printf("\n factorial of %d is %d",n,f);
}

/*Factorial of a given number by using Non-Recursive function*/
#include<stdio.h>
int fact(int n)
{
int f=1,i;
if((n==0)|| (n==1))
return(1);
else
{
for(i=1;i<=n;i++)
f=f*i;
}
return(f);
}
void main()
{
int n;
printf("enter the number :");
scanf("%d",&n);
printf("factorial of number%d",fact(n));
}

```

Output



```

enter a no.: 5
factorial of 5 is 120
=====

```

6.2 Design and develop a recursive function GCD (num1, num2) that accepts two integer arguments. Write a C program that invokes this function to find the greatest common divisor of two given integers.

Algorithm

GCD of a given two integer by using Recursive Function

1. Start
2. Read two numbers a and b
3. Call a function gcd (a, b) by passing the value of a and b
4. Check a != b, then
5. Check a > b then calculate gcd of two numbers by calling the function gcd(a-b, b)
6. Otherwise calculate gcd of two numbers by calling the function gcd(a, b-a)
7. Display the gcd of two numbers
8. Stop

GCD of a given two integer by using Non-Recursive Function

1. Start
2. Read two numbers a and b
3. Check if a != b then
4. Check if a > b - 1 then set a = a - b otherwise set b = b - a
5. Display the gcd as the value of a.
6. Stop

Source Code

```
/*GCD of a given two integer by using Recursive Function*/
#include <stdio.h>
int gcd(int m, int n)
{
    if(n==0)
        return m;
    else
        gcd(n, m%n);
}
void main()
{
```

```

int a,b,g,l;
printf("\n enter a: ");
scanf("%d",&a);
printf("\n enter b: ");
scanf("%d",&b);
g=gcd(a,b);
l=(a*b)/g;
printf("\n GCD of %d and %d : %d",a,b,g);
printf("\n LCM of %d and %d : %d",a,b,l);
}
/*GCD of a given two integer by using Non-Recursive Function*/
#include<stdio.h>
int gcd(int,int);
void main()
{
    int a,b,x;
    printf("\n enter a: ");
    scanf("%d",&a);
    printf("\n enter b: ");
    scanf("%d",&b);
    x=gcd(a,b);
    printf("G.C.D of %d and %d is %d",a,b,x);
}
int gcd(int a,int b)
{
    int r;
    while(b!=0)
    {
        r=a%b;
        a=b;
        b=r;
    }
    return a;
}

```

Output

```

enter a: 34
enter b: 12
G.C.D of 34 and 12 is 2

```


6.3 C.Design and develop a recursive function FIBO (num) that accepts an integer argument. Write a C program that invokes this function to generate the Fibonacci sequence up to num.

Algorithm To print Fibonacci Series using Non-Recursive function

1. Start
2. initialize the a=0, b=1
3. read n
4. if n== 1 print a go to step 7. else goto step 5
5. if n== 2 print a, b go to step 7 else print a,b
6. initialize i=3
7. if i!= n do as follows. If not goto step 7
8. c=a+b
9. print c
10. a=b
11. b=c
12. increment i value
13. goto step 6(i)
14. Stop

To print Fibonacci Series using Recursive function

1. Start
2. Declare variables n, i, c
3. Read variable n
4. for (c = 1 ; c != n ; c++)
5. Display Fibonacci(i)
6. Increment i
7. if (n == 0)
8. return 0
9. else if (n == 1)

10. return 1
11. return (Fibonacci(n-1) + Fibonacci(n-2))
12. Stop

Source Code

```
/*Fibonacci Series using Non-Recursive function*/
#include<stdio.h>
int fibo(int,int);
void main()
{
    int n,x,i;
    printf("\n enter the no. of terms: ");
    scanf("%d",&n);
    printf("\n the fibonacci sequence is: ");
    for(i=0;i<=n;i++)
    {
        x=fib(i);
        printf("%5d",x);
    }
}

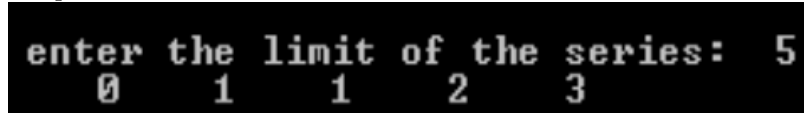
int fib(int n)
{
    if(n==0 || n==1)
        return n;
    else
        return fib(n-1)+fib(n- 2);
}

/*To print Fibonacci Series using Recursive function*/
#include<stdio.h>
void fib(int);
void main()
{
    int n;
    printf("\n enter the limit of the series: ");
    scanf("%d",&n);
    fib(n);
}

void fib(int n)
{
    int x=0,y=1,z,i;
    printf("%5d%5d",x,y);
    for(i=2;i<n;i++)
    {
        z=x+y;
```

```
printf("%5d",z);
x=y;
y=z;
}
}
```

Output



```
enter the limit of the series: 5
 0    1    1    2    3
```

6.4 d.Design and develop a C function ISPRIME (num) that accepts an integer argument and returns 1 if the argument is prime, a 0 otherwise. Write a C program that invokes this function to generate prime numbers between the given ranges.

Algorithm

1. start
2. read the lower limit and upperlimit
3. Iterate until lowerlimit \geq upperlimit the steps 3 to 7
4. invoke the function isprime(num)
5. if num%2 =0 return 0
6. else return 1
7. display num
8. stop

Source Code

```
#include <stdio.h>
int isPrime(int num);
void printPrimes(int lowerLimit, int upperLimit);
int main()
{
    int lowerLimit, upperLimit;

    printf("Enter the lower and upper limit to list primes: ");
    scanf("%d%d", &lowerLimit, &upperLimit);
```

```

// Call function to print all primes between the given range.
printPrimes(lowerLimit, upperLimit);

    return 0;
}
void printPrimes(int lowerLimit, int upperLimit)
{
printf("All prime number between %d to %d are: ",
lowerLimit, upperLimit);

    while(lowerLimit <= upperLimit)
    {
        // Print if current number is prime.
        if(isPrime(lowerLimit))
        {
            printf("%d, ", lowerLimit);
        }

        lowerLimit++;
    }
}
int isPrime(int num)
{
    int i;

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

        if(num % i == 0)
        {
            return 0;
        }
    }

    return 1;
}

```

Output

```

Enter the lower and upper limit to list primes: 10 50
All prime number between 10 to 50 are: 11, 13, 17, 19, 23, 29, 31, 37, 41,
43, 47,

```

- 6.5 e.Design and develop a function REVERSE (str) that accepts a string arguments. Write a C program that invokes this function to find the reverse of a given string.

Algorithm

1. Start
2. Read a sentence using scanf
3. Use `strrev(char *)` function to reverse a sentence
4. calculate length of a string
5. iterate till length/2
6. interchange the first character and last character,second character last second character and so on
7. Display the reversed sentence
8. Stop

Source code

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

void revstr(char *str1)
{
    int i, len, temp;
    len = strlen(str1);

    // use for loop to iterate the string
    for (i = 0; i < len/2; i++)
    {
        temp = str1[i];
        str1[i] = str1[len - i - 1];
        str1[len - i - 1] = temp;
    }
}

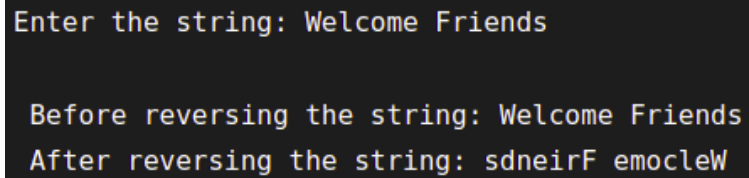
int main()
{
    char str[50];
    printf (" Enter the string: ");
```

```
    gets(str);

    printf (" \n Before reversing the string: %s \n", str);

    revstr(str);
    printf (" After reversing the string: %s", str);
}
```

Output

A screenshot of a terminal window with a dark background and light-colored text. The output shows the user entering 'Welcome Friends', followed by the program printing 'Before reversing the string: Welcome Friends' and 'After reversing the string: sdneirF emocleW'.

```
Enter the string: Welcome Friends

Before reversing the string: Welcome Friends
After reversing the string: sdneirF emocleW
```

Viva Questions

1. What is recursion?
2. What is the necessity of creating a user defined function in C?
3. What are the types of functions according to return values and number of arguments?
4. Can you pass a parameter to a function? If yes, then what are the parameter passing techniques?
5. Which data structure is used in recursion?

7 Pointers

- 7.1 a. Develop a C program using pointers to compute the sum, mean and standard deviation of all elements stored in an array of n real numbers.

Algorithm

1. Start
2. Input the elements Read n, a[i]
3. Initialize sum=0, sumstd=0
4. Compute sum, mean, standard deviation
Ptr=a
For i = 0 through n in steps of 1
Sum=sum+*ptr
Ptr++
Mean=sum/n
Ptr=a
For i 0 through n in steps of 1
Sumstd=sumstd+pow((*ptr-mean),2)
Ptr++
std=sqrt(sumstd/n)
5. Output
Print sum, mean, standard deviation
6. Stop

Source Code

```
#include <stdio.h>
#include <math.h>
void main ()
{
    float a[20], sum1 = 0, sum2 = 0, mean, var, dev;
    int i, n;
    printf ("Enter no of elements:");
    scanf ("%d", &n);
    printf ("Enter array elements:");
    for (i = 0; i < n; i++)
    {
        scanf ("%f", a + i);
        sum1 = sum1 + * (a + i);
    }
    mean = sum1 / n;
    for (i = 0; i < n; i++)
```

```

{
    sum2 = sum2 + pow ((*(a + i) - mean), 2);
}
var = sum2 / n;
dev = sqrt (var);
printf ("Sum      :%f\n", sum1);
printf ("Mean      :%f\n", mean);
printf ("Variance  :%f\n", var);
printf ("Deviation :%f\n", dev);
}

```

Output

```

Enter no of elements:
5
Enter array elements:
1
2
3
4
5
Sum      :15.000000
Mean     :3.000000
Variance :2.000000
Deviation :1.414214

```

7.2 b. Develop a C program to read a list of integers and store it in an array. Then read the array elements using a pointer and print the value along with the memory addresses

Algorithm

1. start
2. declare integer array arr[10],*p
3. for i in 0 to 10
4. read the elements using pointer variable (p+i)
5. Display the array elements and address using the pointer
for i in 0 to 10
address:(p+i) element:*(p+i)
6. stop

source Code

```

#include <stdio.h>

int main()

```



```

{
    int arr[10];           //declare integer array
    int *pa;              //declare an integer pointer
    int i;

    pa=&arr[0];           //assign base address of array

    printf("Enter array elements:\n");
    for(i=0;i < 10; i++){
        printf("Enter element %02d: ",i+1);
        scanf("%d",pa+i); //reading through pointer
    }

    printf("\nEntered array elements are:");
    printf("\nAddress\t\tValue\n");
    for(i=0;i<10;i++){
        printf("%08X\t\t%03d\n", (pa+i), *(pa+i));
    }
    return 0;
}

```

Output

```

Enter array elements:
Enter element 01: 11
Enter element 02: 23
Enter element 03: 444
Enter element 04: 4
Enter element 05: 5
Enter element 06: 6
Enter element 07: 77
Enter element 08: 89
Enter element 09: 67
Enter element 10: 12

Entered array elements are:
Address      Value
E73BF180     011
E73BF184     023
E73BF188     444
E73BF18C     004
E73BF190     005
E73BF194     006
E73BF198     077
E73BF19C     089
E73BF1A0     067
E73BF1A4     012

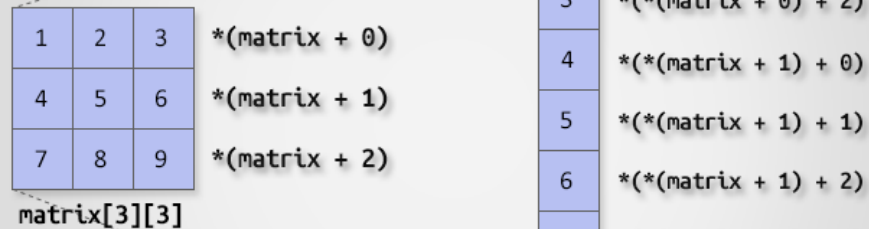
```

7.3 Design and develop non-recursive functions `input_matrix` (`matrix`, `rows`, `cols`) and `print_matrix` (`matrix`, `rows`, `cols`) that stores integers into a twodimensional array and displays the integers in matrix form. Write a C program to input and print elements of a two dimensional array using pointers and functions.

```
matrix          => Points to base address of two-dimensional array.
                  Since array decays to pointer.

*(matrix)       => Points to first row of two-dimensional array.
*(matrix + 0)   => Points to first row of two-dimensional array.
*(matrix + 1)   => Points to second row of two-dimensional array.

**matrix        => Points to matrix[0][0]
*(*(matrix + 0)) => Points to matrix[0][0]
*(*(matrix + 0) + 0) => Points to matrix[0][0]
*(matrix + 1)   => Points to matrix[0][1]
*(*(matrix + 0) + 1) => Points to matrix[0][1]
*(*(matrix + 2) + 2) => Points to matrix[2][2]
```



Source Code

```
/* C program to access two dimensional array using pointer.*/
#include <stdio.h>
#define ROWS 3
#define COLS 3

void inputMatrix(int matrix[][COLS], int rows, int cols);
void printMatrix(int matrix[][COLS], int rows, int cols);
int main()
{
    int matrix[ROWS][COLS];
    int i, j;

    /* Input elements in matrix */
    printf("Enter elements in %dx%d matrix.\n", ROWS, COLS);
    inputMatrix(matrix, ROWS, COLS);

    /* Print elements in matrix */
    printf("Elements of %dx%d matrix.\n", ROWS, COLS);
    printMatrix(matrix, ROWS, COLS);
    return 0;
}

void inputMatrix(int matrix[][COLS], int rows, int cols)
{
    int i, j;
    for(i = 0; i < rows; i++)
    {
        for(j = 0; j < cols; j++)
        {
            scanf("%d", (*(matrix + i) + j));
        }
    }
}

void printMatrix(int (*matrix)[COLS], int rows, int cols)
{
    int i, j;
    for (i = 0; i < rows; i++)
    {
        for (j = 0; j < cols; j++)
        {
            // (*(matrix + i) + j) is equivalent to matrix[i][j]
            printf("%d ", (*(matrix + i) + j));
        }

        printf("\n");
    }
}
```

```
    }  
}
```

Output

```
Enter elements in 3x3 matrix.  
1 2 3  
4 5 6  
7 8 9  
Elements of 3x3 matrix.  
1 2 3  
4 5 6  
7 8 9
```

- 7.4 Develop a C program to store a list of integers in a single dimensional array using dynamic memory allocation (limit will be at run time) using malloc () function. Write a C program to read the elements and print the sum of all elements along with the entered elements. Also use free () function to release the memory.

Algorithm

1. start
2. declare *ptr,limit,i,sum
3. read the memory dynamically using malloc()
ptr= (int*)malloc(limit * sizeof(int))
4. Read the elements in to the array
for i in 0 to limit
(ptr+i)
5. Display the array elements and sum
for i in 0 to limit *(ptr+i) sum+=*(ptr+i)
6. stop

Source Code

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

```

int main()
{
    int* ptr; //declaration of integer pointer
    int limit; //to store array limit
    int i; //loop counter
    int sum; //to store sum of all elements

    printf("Enter limit of the array: ");
    scanf("%d", &limit);

    //declare memory dynamically
    ptr = (int*)malloc(limit * sizeof(int));

    //read array elements
    for (i = 0; i < limit; i++) {
        printf("Enter element %02d: ", i + 1);
        scanf("%d", (ptr + i));
    }

    //print array elements
    printf("\nEntered array elements are:\n");
    for (i = 0; i < limit; i++) {
        printf("%d\n", *(ptr + i));
    }

    //calculate sum of all elements
    sum = 0; //assign 0 to replace garbage value
    for (i = 0; i < limit; i++) {
        sum += *(ptr + i);
    }
    printf("Sum of array elements is: %d\n", sum);

    //free memory
    free(ptr); //hey, don't forget to free dynamically allocated memory.

    return 0;
}

```

Output

```
Enter limit of the array: 5
Enter element 01: 100
Enter element 02: 200
Enter element 03: 300
Enter element 04: 400
Enter element 05: 500

Entered array elements are:
100
200
300
400
500
Sum of array elements is: 1500
```

Viva Voice Questions

1. Define Pointer
2. Explain pointer arithmetic with appropriate examples
3. Explain how to access the array elements using pointer
4. Difference between `ptr = (a+0) (ptr+i)` and `*(ptr+i)` where `ptr` is an integer pointer and `(a+0)` is the base address

8 Structures and Unions

8.1 a. Write a C program that uses functions to perform the following operations:

- i. Reading a complex number
- ii. Writing a complex number
- iii. Addition and subtraction of two complex numbers

Note: represent complex number using a structure

Algorithm

1. Start
2. Declare structure for complex numbers
3. Read the complex number
4. Read choice
5. If choice = 1 then addition operation will perform and it contains following steps
 $w.\text{realpart} = w1.\text{realpart} + w2.\text{realpart};$
 $w.\text{imgpart} = w1.\text{imgpart} + w2.\text{imgpart};$ goto step 4
6. If choice = 2 then multiplication operation will perform and it contains following steps
 $w.\text{realpart} = (w1.\text{realpart} * w2.\text{realpart}) - (w1.\text{imgpart} * w2.\text{imgpart});$
 $w.\text{imgpart} = (w1.\text{realpart} * w2.\text{imgpart}) + (w1.\text{imgpart} * w2.\text{realpart});$ goto step 4
7. If choice = 0 then exit operation will perform
8. If $w.\text{imgpart} \neq 0$ then print $\text{realpart} + i\text{imgpart}$ else Print realpart .
9. Stop

Source Code

```
#include <stdio.h>
#include <string.h>
struct complex1
{
    int r;
    int i;
}c1,c2,c3;
void add(struct complex1 c1,struct complex1 c2);
void sub(struct complex1 c1,struct complex1 c2);
void multi(struct complex1 c1,struct complex1 c2);
```

```

void main()
{
printf("\n enter first complex no: ");
scanf("%d%d",&c1.r,&c1.i);
printf("\n enter second complex no: ");
scanf("%d%d",&c2.r,&c2.i);
printf("\n entered first complex no:%d+i%d",c1.r,c1.i);
printf("\n entered second complex no:%d+i%d",c2.r,c2.i);
add(c1,c2);
sub(c1,c2);
multi(c1,c2);
}
void add(struct complex1 c1,struct complex1 c2)
{
c3.r=c1.r+c2.r;
c3.i=c1.i+c2.i;
printf("\n addition of two complex numbers is %d+i%d",
c3.r,c3.i);
}
void sub(struct complex1 c1,struct complex1 c2)
{
c3.r=c1.r-c2.r;
c3.i=c1.i-c2.i;
printf("\n subtraction of two complex numbers is
%d+i%d",c3.r,c3.i);
}
void multi(struct complex1 c1,struct complex1 c2)
{
c3.r=(c1.r*c2.r)-(c1.i*c2.i);
c3.i=(c1.r*c2.i)+(c1.i*c2.r);
printf("\n multiplication of two complex numbers is %d+i%d",
c3.r,c3.i);
}

```

Output

```

enter first complex no:  4 5
enter second complex no:  6 7

enter first complex no:4+i5
enter second complex no:6+i7
addition of two complex numbers is 10+i12
subtraction of two complex numbers is          -2+i-2
multiplication of two complex numbers is -11+i58_

```


- 8.2 b. Write a C program to compute the monthly pay of 100 employees using each employee's name, basic pay. The DA is computed as 52% of the basic pay. Gross salary (basic pay + DA). Print the employee's name and gross salary

Algorithm

1. Start
2. Declare structure for employee details
3. Read 100 employee details
4. Repeat loop 100 employees and calculate gross salary of each employee
 $DA = 0.52 * \text{basic}$ Gross Salary = Basic + DA
5. Print each employee name and gross salary
6. Stop

Source Code

```
/*Program to calculate gross salary of the employees*/
#include <stdio.h>
struct employee
{
    char name[20];
    float basic;
    float da;
    float gross;
}e[5];
void main()
{
    int i;
    for(i=0; i<5; i++)
        scanf("%s%f", e[i].name, &e[i].basic);
    for(i=0; i<5; i++)
    {
        e[i].da = 52.0/100 * e[i].basic;
        e[i].gross = e[i].da + e[i].basic;
        printf("\n name=%s gross=%f", e[i].name, e[i].gross);
    }
}
```

Output

```
enter employee name and basicsalarya 2300
b 32000
c 15000
e 12000
f 22000

name=a gross=3496.000000
name=b gross=48640.000000
name=c gross=22800.000000
name=e gross=18240.000000
name=f gross=33440.000000_
```

8.3 c.Create a Book structure containing book_id, title, author name and price. Write a C program to pass a structure as a function argument and print the book details.

Algorithm

1. Start
2. Declare structure for book details
3. Assign a book details to structure
4. Pass a structure as argument in function
5. Print the book details from the function
6. Stop

Source code

```
/* Program to print the book details using structure*/
#include <stdio.h>
#include <string.h>
struct book
{
int bookid;
char title[30];
char author[20];
```

```

float price;
};
void print_book(struct book b1);
void main()
{
    struct book b1;
    printf("\n enter book details: ");
    scanf("%d%s%s%f",&b1.bookid,b1.title,b1.author,&b1.price);
    print_book(b1);
}
void print_book(struct book b1)
{
    printf("\n book details:%d\t %s\t %s\t %f",b1.bookid,
    b1.title,b1.author,b1.price);
}

```

Output

```

enter book details: 201 cprogramming balaguriswany 230
book details:201      cprogramming      balaguriswany      230.000000_

```

8.4 d.Create a union containing 6 strings: name, home_address, hostel_address, city, state and zip. Write a C program to display your present address

Algorithm

1. Start
2. Declare union for address details like name, home_address, hostel_address, city, state and zip
3. Assign address to union
4. Print details name, home_address, hostel_address, city, state and zip
5. Stop

Source Code

```

/* Program to Print the address details using union*/
#include <stdio.h>
#include <string.h>
union details
{
    char name[20];

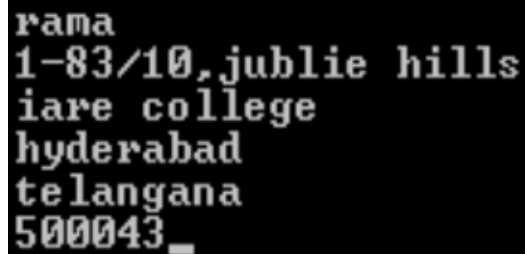
```

```

char home_add[30];
char hostel_add[30];
char city[10];
char state[10];
int pincode;
}a;
void main()
{
strcpy(a.name,"rama");
printf("\n %s",a.name);
strcpy(a.home_add,"1-83/10,jublie hills");
printf("\n %s",a.home_add);
strcpy(a.hostel_add,"iare college");
printf("\n %s",a.hostel_add);
strcpy(a.city,"hyderabad");
printf("\n %s",a.city);
strcpy(a.state,"telangana");
printf("\n %s",a.state);
a.pincode=500043;
printf("\n %d",a.pincode);
getch();
}

```

Output



```

rama
1-83/10,jublie hills
iare college
hyderabad
telangana
500043_

```

Viva Voice Questions

1. What is a structure ?
2. Differentiate between array and structure ?
3. How do you access the member of a structure ?
4. What is the difference between structure and union?
5. Differentiate between address operator and dereferencing operator

9 Additional Programs

- 9.1 Write a C program to read in two numbers, x and n , and then compute the sum of this geometric progression: $1+x+x^2+x^3+\dots+x^n$. 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 $n \leq 0$, then go back and read in the next pair of numbers of without computing the sum. Are any values of x also illegal? If so, test for them too.

Algorithm

1. Start
2. read values of x and n , $\text{sum} = 1$, $i = 1$
3. check for $n \leq 0$
4. if $n \leq 0$ — $x \leq 0$
5. print values are not valid
6. read values of x and n
7. perform the loop operation
8. for($i = 1$; $i \leq n$; $i++$) then follows
9. $\text{sum} = \text{sum} + \text{pow}(x, i)$
10. print sum
11. Stop

Source Code

To read in two numbers, x and n , and then compute the sum of this geometric progression:
 $1+x+x^2+x^3+\dots+x^n$

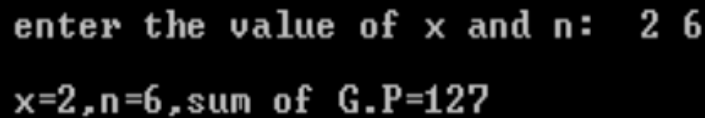
```
#include <stdio.h>
#include <math.h>
void main()
{
    int x,n,sum=0,i;
    printf("\n enter the value of x and n: ");
    scanf("%d%d",&x,&n);
```

```

if(x<0 || n<0)
{
printf("\n enter only positive values for x and n");
goto L;
}
for(i=0;i<=n;i++)
{
sum=sum+pow(x,i);
}
printf("\n x=%d,n=%d,sum of G.P=%d",x,n,sum);
getch();
}

```

Output



```

enter the value of x and n: 2 6
x=2,n=6,sum of G.P=127

```

9.2 b. Develop a C program to find the 2's complement of a given binary number. 2's complement 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.

Algorithm

1. start
2. declare the subprogram "complement(char *a)"
3. initialize the variable i
4. read the binary number
5. perform the loop operation. if it is true then follows. if not goto step 7
6. for(i = 0; a[i] != '\0'; i++)
7. if(a[i] != '0' & a[i] != '1') then displayed the number is not valid. enter the correct number.
8. Exit the loop
9. call sub program 'complemt(a)'

10. Stop

Source Code

```
/*To find the 2's complement of a binary number*/
#include<stdio.h>
void twoscomplement(int a[10],int);
void main()
{
    int n,i,a[10];
    printf("\n enter number of bits in the binary number: ");
    scanf("%d",&n);
    printf("\n enter the binary number into the array: ");
    for(i=0;i<n;i++)
        scanf("%d",&a[i]);
    twoscomplement(a,n);
    getch();
}

void twoscomplement(int a[10],int n)
{
    int i,c=1;
    for(i=0;i<n;i++)
    {
        if(a[i]==0)
            a[i]=1;
        else
            a[i]=0;
    }
    printf("\n one's complement of the binary number is: ");
    for(i=0;i<n;i++)
        printf("%5d",a[i]);
    for(i=n-1;i>=0;i--)
    {
        a[i]=a[i]+c;
        if(a[i]==1)
        {
            c=0;
            break;
        }
        else{
            a[i]=a[i]-2;
            c=1;
        }
    }
    printf("\n 2's complement of the binary number is: ");
    if(c==1)
        printf("%5d",c);
}
```

```

for(i=0;i<n;i++)
printf("%5d",a[i]);
}

```

Output

```

enter number of bits in the binary number: 5
enter the binary number into the array:  1 1 1 0 1 0
one's complement of the binary number is:  0  0  0  0  1  0
2's complement of the binary number is:  0  0  0  0  1  1

```

- 9.3 c.Develop a C program to convert a Roman numeral to its decimal equivalent. E.g. check for the inputs
 - Roman number IX is equivalent to 9 and Roman number XI is equivalent to 11

Algorithm

1. Start
2. Read roman numbers
3. Repeat loop for all given roman letters and convert each letter into digit and add to variable
4. Print the result
5. Stop

Source Code

To convert a Roman numeral to its decimal equivalent

```

#include<stdio.h>
#include<string.h>
void main()
{
char a[10];
int total[10],sum=0,i,l;
printf("\n enter a roman number:
");
gets(a);
l=strlen(a);
for(i=0;i<l;i++)
{
switch(a[i])
{

```

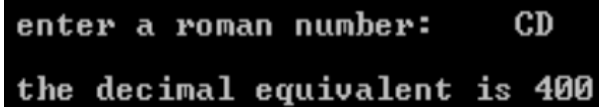


```

case 'M':total[i]=1000;
break;
case 'D':total[i]=500;
break;
case 'C':total[i]=100;
break;
case 'L':total[i]=50;
break;
case 'X':total[i]=10;
break;
case 'I':total[i]=1;
break;
}
sum=sum+total[i];
}
for(i=0;i<l-1;i++)
{
if(total[i]<total[i+1])
{
sum=sum-2*total[i];
}
}
printf("\n the decimal equivalent is %d",sum);
}
}

```

Output



```

enter a roman number:    CD
the decimal equivalent is 400

```

Viva Questions

1. What is a pointer ?
2. How do you declare a pointer variable
3. How do you use a pointer to a function ?
4. Difference between array and pointers ?
5. What is void pointer?
6. What is the difference between malloc and realloc functions?

10 Preprocessor Directives

- 10.1 a. Define a macro with one parameter to compute the volume of a sphere. Write a C program using this macro to compute the volume for spheres of radius 5, 10 and 15 meters

Algorithm

1. Start
2. Define PI constant value 3.14
3. Define macro function to pass a radius value as a argument
4. Call the macro function
5. Print the result
6. Stop

Source Code

```
/*To compute the volume for spheres of radius 5, 10 and 15 meters*/
#include <stdio.h>
#include <math.h>
#define PI 3.142
#define volume(r) ((4/3.0)*PI*pow(r,3))
void main()
{
    int r;
    float v;
    scanf("%d",&r);
    v=volume(r);
    printf("\n volume of the sphere v=%.3f",v);
}
```

Output

```
spheres of radius 5 Volume is 314.150000
spheres of radius 10 Volume is 1256.600000
spheres of radius 15 Volume is 2827.350000_
```

10.2 b. Define a macro that receives an array and the number of elements in the array as arguments. Write a C program for using this macro to print the elements of the array.

Algorithm

1. Start
2. Define macro function that receive array and number of elements
3. Repeat until all elements in the array complete and print element
4. Call the macro function
5. Stop

Source Code

To print the elements of the array

```
#include <stdio.h>
#define PRINTARRAY(a,l)
int i=0;
while(i<l)
{
printf("%5d",a[i]);
i++;

}
void main()
{
int a[10]={10,25,47,56,89,34,68,45,28,03};
PRINTARRAY(a,10);
}
```

Output

```
10  25  47  56  89  34  68  45  28  3
-----
```

10.3 Write symbolic constants for the binary arithmetic operators +, -, *, and /. Write a C program to illustrate the use of these symbolic constants

Algorithm

1. Start
2. Define + as ADD
3. Define - as SUB
4. Define * as MUL
5. Define / as DIV
6. Perform addition operation using ADD and print the result
7. Perform subtraction operation using SUB and print the result
8. Perform Multiplication operation using MUL and print the result
9. Perform Division operation using DIV and print the result
10. Stop

Source Code

```
/*To illustrate the use of these symbolic constants*/
#include <stdio.h>
#define PLUS +
#define MINUS -
#define MUL *
#define DIV /
#define MOD %
void main()
{
    int a=20,b=30;
    printf("\n addition=%d",a PLUS b);
    printf("\n subtraction=%d",a MINUS b);
    printf("\n multiplication=%d",a MUL b);
    printf("\n division=%d",a DIV b);
    printf("\n modulus=%d",a MOD b);
}
```

Output

```
addition=50  
subtraction=-10  
multiplication=600  
division=0  
modulous=20
```

Viva Questions

1. What is the preprocessor in C ?
2. What is a symbolic constant??
3. What is a macro in C programming??
4. How can you avoid including a header more than once?
5. Can a file other than a .h file be included with include?

11 Files

- 11.1 a. Create an employee file employee.txt and write 5 records having employee name, designation, salary, branch and city. Develop a C program to display the contents of employee.txt file.

Algorithm0

1. Start
2. Read file name
3. Open file in read mode
4. Repeat until end of the file and print the contents if the file
5. Close the file
6. Stop

Source code

To display the contents of a file on screen

```
#include <stdio.h>
void main()
{
    FILE *fp;
    char ch;
    fp=fopen("employee.txt","r");
    if(fp==NULL)
    {
        printf("file opening problem");
        return;
    }
    while(!feof(fp))
    {
        ch=fgetc(fp);
        printf("%c",ch);
    }
    fclose(fp);
}
```

11.2 b. Create a studentolddata.txt file containing student name, roll no, branch, section, address. Develop a C program to copy the contents of studentolddata.txt file to another file studentnewdata.txt.

Algorithm

1. Start
2. read command line arguments
3. check if no of arguments =3 or not. If not print invalid no of arguments
4. open source file in read mode
5. if NULL pointer, then print source file cannot be open
6. open destination file in write mode
7. if NULL pointer, then print destination file cannot be open
8. read a character from source file and write to destination file until EOF
9. Close source file and destination file
10. Stop

Source Code

```
/*Copy the contents from one file to another*/
#include <stdio.h>
void main()
{
    FILE *fp1, *fp2;
    char ch;
    fp1=fopen("old.txt","r");
    fp2=fopen("new.txt","w");
    if(fp1==NULL||fp2==NULL)
    {
        printf("file opening problem");
        return;
    }
    while(!feof(fp1))
    {
        ch=fgetc(fp1);
        fputc(ch,fp2);
    }
    fclose(fp1);
    fclose(fp2);
    getch();
}
```

11.3 c. Develop a C program to create a text file info.txt to store the information given below. Implement using a C program to count the number of words and characters in the file info.txt.

Algorithm

1. start
2. open the file using file pointer
3. read the contents of file character by character
4. iterate the contents until the END of FILE is reached
5. if white space or newline is encountered increment the word count and line count
6. stop

Source Code

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

void main()
{
    FILE *fptr;
    char ch;
    int wrd=1,charctr=1;
    char fname[20];
    printf("\n\n Count the number of words and characters in a file :\n");
    printf("-----\n");
    printf(" Input the filename to be opened : ");
    scanf("%s",fname);

    fptr=fopen(fname,"r");
    if(fptr==NULL)
    {
        printf(" File does not exist or can not be opened.");
    }
    else
    {
        ch=fgetc(fptr);
        printf(" The content of the file %s are : ",fname);
        while(ch!=EOF)
        {
            printf("%c",ch);
            if(ch==' ' || ch=='\n')

```



```

        {
            wrd++;
        }
        else
        {
            charctr++;
        }
        ch=fgetc(fptr);
    }
    printf("\n The number of words in the file %s are : %d\n",fname,wrd-2);
    printf(" The number of characters in the file %s are : %d\n\n",fname,charctr-1);
}
fclose(fptr);
}

```

- 11.4 d. Given two university information files “student-name.txt” and “roll_number.txt” that contains students Name and Roll numbers respectively. Write a C program to create a new file called “output.txt” and copy the content of files “studentname.txt” and “roll_number.txt” into output file. Display the contents of output file “output.txt” on to the screen.

Algorithm

1. Start
2. read command line arguments
3. check if no of arguments =3 or not. If not print invalid no of arguments
4. open source file in read mode
5. if NULL pointer, then print source file cannot be open
6. open destination file in write mode
7. if NULL pointer, then print destination file cannot be open
8. read a character from source file and write to destination file until EOF
9. Close source file and destination file
10. Stop

Source code

Copy the contents from one file to another

```
#include <stdio.h>
void main()
{
FILE *fp1, *fp2;
char ch;
fp1=fopen("student.txt", "r");
fp2=fopen("output.txt", "w");
if(fp1==NULL || fp2==NULL)
{
printf("file opening problem");
return;
}
while(!feof(fp1))
{
ch=fgetc(fp1);
fputc(ch, fp2);
}
fclose(fp1);
fclose(fp2);
getch();
}
```

Viva Questions

1. What is use of `math.h` header file ?
2. Describe the file opening mode “w+”
3. Is FILE a built-in data type?
4. How can we determine whether a file is successfully opened or not using `fopen()` function?
5. How can we know read/write permission any given file?

12 Command Line Arguments

12.1 a. Develop a C program to read a set of arguments and display all arguments given through command line.

1. Start
2. Pass a to arguments argc and argv in main function
3. Check the condition argc equal to 2 then printf the argv[1] value
4. Else if argc is greater than 2 then print Too many arguments supplied
5. Else One argument expected
6. Stop

Source Code

```
/*To read arguments at the command line and display it*/
#include <stdio.h>
#include <string.h>
void main(int argc, char *argv[])
{
    int i;
    printf("\n total no of arguments=%d", argc);
    for(i=0; i<argc; i++)
        printf("\n args[%d]=%s", i, argv[i]);
}
```

12.2 b. Develop a C program to read a file at command line argument and display the contents of the file.

Algorithm

1. Start
2. Read command line arguments
3. Check if no of arguments =2 or not. If not print invalid no of arguments
4. Open file in read mode
5. Repeat until end of the file and print the contents if the file
6. Close the file
7. Stop

Source Code

```
#include <stdio.h>
void main(int argc, char * argv[])
{
    FILE *fp;
    char ch;
    fp=fopen(argv[1], "r");
    if(fp==NULL)
    {
        printf("\n opening problem");
        return;
    }
    while(!feof(fp))
    {
        ch=getc(fp);
        printf("%c", ch);
    }
    fclose(fp);
}
```

12.3 Develop a C program to read N integers and find the sum of N integer numbers using command line arguments.

Algorithm

1. start
2. if argc<2 not sufficient arguments
3. else iterate until the i<argc calculate sum+=atoi(argv[i])
4. display sum
5. stop

Source Code

```
#include <stdio.h>

int main(int argc, char *argv[])
{
    int a,b,sum;
    int i; //for loop counter

    if(argc<2)
    {
```

```

        printf("please use \"prg_name value1 value2 ... \"\n");
        return -1;
    }

    sum=0;
    for(i=1; i<argc; i++)
    {
        printf("arg[%2d]: %d\n",i,atoi(argv[i]));
        sum += atoi(argv[i]);
    }

    printf("SUM of all values: %d\n",sum);

    return 0;
}

```

12.4 d. Develop a C program to read three integers and find the largest integer among three using command line argument.

Algorithm

1. start
2. if $\text{argc} < 4$ — $\text{argc} < 5$) enter 4 arguments only
3. $a = \text{atoi}(\text{argv}[1]);$
 $b = \text{atoi}(\text{argv}[2]);$
 $c = \text{atoi}(\text{argv}[3]);$
4. if $(a < 0 \text{ — } b < 0 \text{ — } c < 0)$ "enter positive values only"
5. if $(!(a != b \text{ } b != c \text{ } a != c))$ "enter three different values"
6. else if $(a \geq b \text{ } a \geq c)$ "a is largest"
7. else if $(b \geq c \text{ } b \geq a)$ "b is largest"
8. else "C is largest"
9. stop

Source Code

```

// C program for finding the largest integer
// among three numbers using command line arguments
#include <stdio.h>

// Taking argument as command line

```

```

int main(int argc, char *argv[])
{
    int a, b, c;
    // Checking if number of argument is
    // equal to 4 or not.
    if (argc < 4 || argc > 5)
    {
        printf("enter 4 arguments only eg.\\"filename arg1 arg2 arg3!!\\"");
        return 0;
    }
    // Converting string type to integer type
    // using function "atoi( argument)"
    a = atoi(argv[1]);
    b = atoi(argv[2]);
    c = atoi(argv[3]);

    // Checking if all the numbers are positive or not
    if (a < 0 || b < 0 || c < 0)
    {
        printf("enter only positive values in arguments !!");
        return 1;
    }

    // Checking if all the numbers are different or not
    if (!(a != b && b != c && a != c))
    {
        printf("please enter three different value ");
        return 1;
    }
    else
    {
        // Checking condition for "a" to be largest
        if (a > b && a > c)
            printf("%d is largest", a);

        // Checking condition for "b" to be largest
        else if (b > c && b > a)
            printf ("%d is largest", b);

        // Checking condition for "c" to be largest..
        else if (c > a && c > b)
            printf("%d is largest ",c);
    }
    return 0;
}

```