Output 1:

Hello World

Output 2:

44 Binori Bunglows,
Bopal,
Ahmedabad

Date 5-8-25

Expt. No. _____ Page No. 1.

Experiment - 1

1)- Write a C program to print "Hello World".

⇒
```
#include <stdio.h>
int main() {
    printf("Hello World");
    return 0;
}
```

2)- Write a C program to print adress in multiple lines (newline).

⇒
```
#include <stdio.h>
int main() {
    printf("44 Binori Bunglows,\n");
    printf(" Bopal,\n");
    printf(" Ahmedabad);
    return 0;
}
```

7 Rohan
13/8/25

Teacher's Signature _____

Output 3:

```
Enter your name: Sahaj
Enter your age: 18
Hello, Sahaj! You are 18 years old.
```

Output 4:

```
Enter first number: 6
Enter second number: 6
Sum is 12
```

3)- Write a program that prompts the user to enter their name and age.

⇒
```
#include <stdio.h>
int main() {
    char name;
    int age;
    printf("Enter your name:");
    scanf("%s", name);
    printf("Enter your age:");
    scanf("%d", &age);
    printf("\nHello, %s! You are %d years old.", name, age);
    return 0;
}
```

4)- Write a C program to add two numbers, take numbers from user.

⇒
```
#include<stdio.h>
int main() {
    int a, b;
    printf("Enter first number:");
    scanf("%d", &a);
    printf("Enter second number:");
    scanf("%d", &b);
    printf("Sum is %d", a+b);
    return 0;
}
```

Output 1:

```
Enter length of rectangle : 10
Enter breadth of rectangle: 15
Perimeter of rectangle = 50
Area of rectangle = 150
```

Output 2:

```
Enter temperature in Celsius : 15
Temperature in farenheit = 59.000000
```

Date 5-8-25

Expt. No. _____ Page No. 3

Experiment-2

1)- Write a program in C to calculate the area and perimeter of a rectangle based on its length and width.

⇒
```
#include <stdio.h>
int main() {
    int length, breadth;
    printf("Enter length of rectangle:");
    scanf("%d", length);
    printf("Enter breadth of rectangle:");
    scanf("%d", breadth);
    printf("Perimeter of rectangle = %d \n", 2*(length+breadth));
    printf("Area of rectangle = %d", (length* breadth));
    return 0;
}
```

2)- Write a C program to convert temperature from Celsius to Farenheit.

⇒
```
#include <stdio.h>
int main(){
    float celsius;
    printf("Enter temperature in Celsius:");
    scanf("%f", celsius);
    float fareheit;
    farenheit = (celsius * 9/5 ) + 32;
    printf("Temperature in farenheit = %f", farenheit);
    return 0;
}
```

Teacher's Signature _____

Output 1:

```
Enter length 1 of triangle: 4
Enter length 2 of triangle: 4
Enter length 3 of triangle: 4
It is a valid triangle
It is an equilateral triangle
```

#What concepts we used in this part?

→ Maths:

For a valid triangle:

↳ the sum of any 2 sides should be greater than the third side.

[we used like $(l1 + l2) > l3$ ....]

→ C Language:

We used if - else statements to check whether the triangle is valid or not.

If it is valid we printed that "it is a valid triangle", and if it wasn't we printed that "it is not a valid triangle".

→ Maths: [Again]

For equilateral triangle all 3 sides should be equal.
For isoceles triangle any 2 sides of the triangle should be equal
For right angled triangle the sum of square of 2 sides should be equal to square of third side.

Date 13-8-25

Expt. No. _____ Page No. 4.

Experiment 3.1

1)- Write a program to check if the triangle is valid or not. If the validity is established, do check if the triangle is isosceles, equilateral, right angle or scalene. Take sides of the triangle as input from user.

⇒
```
#include <stdio.h>
int main () {
    int l1, l2, l3;
    printf("Enter length 1 of triangle:");
    scanf(" %d", &l1);
    printf("Enter length 2 of triangle:");
    scanf(" %d", &l2);
    printf("Enter length 3 of triangle:");
    scanf(" %d", &l3);

    if ((l1+l2)>l3 && (l2+l3)>l1 && (l1+l3)>l2) {
        printf("It is a valid triangle\n"); }
    else {
        printf("It is not a valid triangle\n");
    }

    if ((l1==l2) && (l2==l3)) {
        printf("It is an equilateral triangle"); }
    else if (l1*l1+l2*l2 = l3*l3) || (l2*l2+l3*l3=l1) ||
                (l1*l1+l3*l3 = l2*l2)) {
        printf("It is a right triangle");
    }
```

P. T. O →

Teacher's Signature _____

for scalene triangle all sides are different.

→ C Language [Again]:

We then used if-else if-else statement to check if the triangle is equilateral, right angled, isoceles or a scalene triangle and printed the result.

Output 2:

Enter your weight in Kgs : 70
Enter your height in mts : 1.79

BMI index is : 21.645324020961 81
Category : Ideal

What concepts we used in this part?

→ Maths: We used the simple formula of BMI = $\frac{\text{weight}}{\text{Height} * \text{Height}}$.

→ C Languge: After calculating the BMI, we used the table to print the category of the resultant BMI using if-elseif-else statements.

```
else if ((l1==l2)||(l2==l3)||(l3==l1)) {
        printf("It is a isoceles triangle");
}
else {
        printf("It is a scalene triangle");
}

return 0;
}
```

2)- Write a program to compute the BMI index of the person and print the BMI values as per the following ranges.

BMI = weight (Kgs) / Height (Mts) * Height (Mts)

⇒

```
#include <stdio.h>
int main() {
        float height, weight;
        printf("Enter your weight in Kgs: ");
        Scanf(" %d", &weight);
        printf("Enter your height in mts: ");
        scanf(" %d", &height);

        float bmi;
        bmi = weight/height * height;
        printf("BMI index is: %f", bmi);
```

Category	BMI
Starvation	<15
Anorexic	15.1 to 17.5
Underweight	17.6 to 18.5
Ideal	18.6 to 24.9
Overweight	25 to 29.9
Obese	30 to 39.9
Morbidity Obese	40.0 above.

Date 13-8-25

Expt. No. _____ Page No. 6

```
if bmi <15 {
        printf(" Category: Starvation "); }
else if bmi >= 15.1 && bmi =< 17.5 {
        printf (" Category: Anorexic ");
}
else if bmi >= 17.6 && bmi =< 18.5 {
        printf (" Category: Underweight ");
}
else if bmi >= 18.6 && bmi =< 24.9 {
        printf (" Category: Ideal ");
}
else if bmi >= 25 && bmi =< 29.9 {
        printf ("Category: Overweight ");
}
else if bmi >= 30 && bmi =< 39.9 {
        printf (" Category: Obese ");
}
else if bmi >= 40 {
        printf(" Category: Morbidity Obese ");
}
else {
        printf(" Invalid BMI, check your input ");
}

return 0;
}
```

Teacher's Signature _____

Output 3:

```
Enter coordinates of point 1 (x1, y1): 1 1
Enter coordinates of point 2 (x2, y2): 1 4
Enter coordinates of point 3 (x3, y3): 1 5
The points are collinear
```

What concepts we used in this part?

→ Maths:

To check the collinearity of 3 points we used the formula; $x_1*(y_2-y_3) + x_2*(y_3-y_1) + x_3*(y_1-y_2)$; which is the formula for area of triangle

If we get the value from above formula to be 0, then the points are collinear or else they are not collinear.

→ C language;

We took two values from the user in same line:

```
scanf("%d %d", &x1, &y1);
```

Here C language allows us to input 2 values from the user using only one line, if the values are separated by a space.

Then we used if-else statement to check if the area of triangle is 0 or not. If it is zero, we printed that the points are collinear.

Date 13-8-25

Expt. No. _____ Page No. 7

3)- Write a program to check if three points (x_1, y_1), (x_2, y_2) and (x_3, y_3) are collinear or not.

⇒
```
#include <stdio.h>
int main() {
    int x1, y1, x2, y2, x3, y3;
    printf("Enter coordinates of point 1 (x1, y1):");
    scanf("%d %d", &x1, &y1);

    printf("Enter coordinates of point 2 (x2, y2):");
    scanf("%d %d", &x2, &y2);

    printf("Enter coordinates of point 3 (x3, y3):");
    scanf("%d %d", &x3, &y3);

    int a;
    a = x1*(y2-y3) + x2*(y3-y1) + x3*(y1-y2);

    if (a==0){
        printf("The points are collinear"); }
    else {
        printf("The points are not collinear");
    }

    return 0;

}
```

Teacher's Signature _____

Output 4:

Enter the year: 2035
Monday

Date 13-8-25

Expt. No. _____ Page No. 8

4)- According to the gregorian calendar, it was Monday on the date 01/01/01. If any year is input through the keyboard write a program to find out what is the day on 1st January of that year.

⇒
```
#include <stdio.h>
int main() {
    int year, total_days, day, i;

    printf("Enter the year: ");
    scanf("%d", &year);
    total_days = 0;
    printf("Enter Year: ");
    scanf("%d", &year);

    for (i=1; i<year; i++){
        if (i%4==0 && i%100!=0 || i%400==0) {
            total_days = total_days +366;
        }
        else {
            total_days = total_days + 365;
        }
}

    day = total_days %7;
```

P.T.O →

Teacher's Signature _____

```
if (day==0) {
        printf ("Monday"); }
elseif (day==1) {
        printf("Tuesday");
}
else if(day == 2) {
        printf("Wednesday");
}
else if (day==3) {
        printf("Thursday");
}
else if (day==4) {
        printf("Friday");
}
else if (day==5) {
        printf("Saturday");
else if(day==6) {
        printf("Sunday");
}
return 0;
}
```

Teacher's Signature _____

Output 5:

Enter length of 1st rectangle: 3
Enter breadth of 1st rectangle: 3
Enter length of 2nd rectangle: 4
Enter breadth of 2nd rectangle: 4
Enter length of 3rd rectangle: 2
Enter breadth of 3rd rectangle: 2

largest perimeter = 16

What concepts we used in this part?

→ Maths:

After inputing the lengths and breadths of all 3 triangles we used the formula 2*(l+b) to calculate the perimeter of a triangle.

→ C Language:

(→ also known as conditional operator)

We used ternary operators to find the largest perimeter among the three and then printed it.

- Ternary Operators: They are shorthand way to write simple if-else statements.
  ↳ It takes 3 operands.

Date 13-8-25

Expt. No. _____ Page No. 10

5)- Write a program using ternary operator, the user should input the length and breadth of a rectangle, one has to find out which rectangle has the highest perimeter. The minimum number of rectangles should be three.

⇒
```
#include <stdio.h>
int main () {
    int l1, l2, l3, b1, b2, b3;
    printf(" Enter length of 1st rectangle: ");
    scanf(" %d ", &l1);
    printf(" Enter breadth of 1st rectangle: ");
    scanf (" %d ", &b1);

    printf(" Enter length of 2nd rectangle: ");
    scanf(" %d ", &l2);
    printf(" Enter breadth of 2nd rectangle: ");
    scanf(" %d", &b2);

    printf(" Enter length of 3rd rectangle: ");
    scanf(" %d", &l3);
    printf(" Enter breadth of 3rd rectangle: ");
    scanf(" %d", &b3);

    int per1, per2, per3, largest;
    per1 = 2* (l1 + b1);
    per2 = 2* (l2 + b2);
    per3 = 2* (l3 + b3);
```

Teacher's Signature _____

Date 13-8-25

Expt. No. _____ Page No. 11

```
largest = per1>per2?((per1>per3)?per1:per3):
                    ((per2>per3)?per2:per3);
printf("largest perimeter = %d", largest);

return 0;
}
```

Teacher's Signature _____

Output - 1:

```
bitwise a OR b is 7
bitwise a AND b is 5
bitwise not of a is -6
```

Output - 2:

```
Left shift operator 2
Right shift operator 40
```

What concepts we used?

- bitwise operators: operators that perform operations directly on the individual bits
- bitwise OR (|): Compares corresponding bits of two operands. If atleast one bit is 1 then output is 1; else it is 0.
- bitwise NOT (~): It is a unary operators that inverts each bit of its operand.
- bitwise AND (&): Compares bits of two operands. If both are 1 then output is 1; else it is 0.

Date 13-8-25

Expt. No. _____ Page No. 12

Experiment - 11

1)- Write a program to apply bitwise OR, AND and NOT operators on bit level.

⇒
```
#include <stdio.h>
int main() {
    int a=5;
    int b=7;
    printf("bitwise a OR b is %d \n", a|b);
    printf("bitwise a AND b is %d \n", a&b);
    printf("bitwise a not of a is %d \n", ~a);
    return 0;
}
```

2)- Write a program to apply left shif and right shift operator.

⇒
```
#include <stdio.h>
int main() {
    int a=10;
    printf("left shift operator, %d \n", a>>2);
    printf("Right shift operator, %d \n", a<<2);
    return 0;
}
```

Teacher's Signature _____

Output 1:

```
Enter a number : 4
Do you want to continue ? y
Enter a number : 0
Do you want to continue ? y
Enter a number : -1
Do you want to continue ? n
Positive numbers = 1
Negative numbers = 1
Zero = 1
```

Date _____

Expt. No. _____ Page No. 13

Experiment 3.2

1)- Write a program to enter numbers till the user wants. At the end, it should display the count of positive, negative and zeroes entered.

⇒
```
#include <stdio.h>
int main (){
    int a;
    int positive=0, negative=0, zero=0;
    char choice;
    do {
        printf(" Enter a number :\n");
        scanf(" %d", &a);
        if (a>0){
            positive ++ ;
        }
        else if (a <0){
            negative ++;
        }
        else {
            zero ++;
        }
        printf(" Do you want to continue ? (y/n) : ");
        scanf(" %c", & choice);
        }
        while(choice=='y' || choice == 'Y');
    printf(" Positive numbers = %d \n", positive);
    printf(" Negative numbers = %d \n", negative);
    printf(" zero = %d \n", zero);
    return 0;
}
```

Teacher's Signature _____

Output 2:

Enter a number : 5

5 * 1 = 5
5 * 2 = 10
5 * 3 = 15
5 * 4 = 20
5 * 5 = 25
5 * 6 = 30
5 * 7 = 35
5 * 8 = 40
5 * 9 = 45
5 * 10 = 50

Output 3:

a)
```
1
2 3
4 5 6
```

2)- Write a program to print the multiplication table of the number entered by the user. It should be in the correct format ⇒ Num * 1 = Num.

⇒
```
#include <stdio.h>
int main() {
    int a;
    printf("Enter a number: ");
    scanf("%d", &a);
    for (int i=1; i<=10; i++){
        printf("%d * %d = %d\n", a, i, a*i);
    }
    return 0;
}
```

3)- Write a program to generate the following set of output:

```
1
2 3
4 5 6
```

a)-
```
#include <stdio.h>
int main () {
    int num = 1;
    for (int i=1; i<=4; i++) {
        for (int k=1; k<=4-i; k++) {
            printf(" ");
        }
        for (int j=1; j<=i; j++) {
            printf("%d", num);
            num += 1;
        }
        printf("\n");
    }
    return 0;
}
```

b)-

```
        1
      1   1
    1   2   1
  1   3   3   1
1   4   6   4   1
```

Output 4:

```
End of year 1 population will be = 110000
End of year 2 population will be = 121000
End of year 3 population will be = 133100
End of year 4 population will be = 146410
End of year 5 population will be = 161051
End of year 6 population will be = 177156
End of year 7 population will be = 194,872
End of year 8 population will be = 214359
End of year 9 population will be = 235795
End of year 10 population will be = 259374
```

b) - #include <stdio.h>
int main() {
    int n = 5;
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n-i-1; j++) {
            printf(" ");
        }
        int val = 1;
        for (int k = 0; k <= i; k++) {
            printf("%d", val);
            val = val * (i-k)/(k+1);
        }
        printf("\n");
    }
    return 0; }

1
1 1
1 2 1
1 3 3 1
1 4 6 4 1

4) - The population of a town is 100000. The population has increased steadily at a rate of 10% for 10 years. Write a program to determine the population at the each year in the last decade.

⇒
#include <stdio.h>
int main() {
    long population = 100000;
    float rate = 0.10;
    int years = 10;
    for (int i = 1; i <= years; i++) {
        population = population * (1 + rate);
        printf("End of Year %d, population will be = %d", i, population);
    }
    return 0;
}

Teacher's Signature

Output 5:

Enter the limit (L): 20

Ramanujan numbes upto the cube of 20:
1729 = 1^3 + 12^3 = 9^3 + 10^3
4104 = 2^3 + 16^3 = 9^3 + 15^3

Date _____

Expt. No. _____ Page No. 16

5) Ramanujan number is the smallest number that can be expressed as the sum of two cubes in two different ways. Write a program to print all such numbers upto a reasonable limit.
Example: Ramanujan number: 1729
12^3+1^3 and 10^3+9^3. For a number L = 20 (that is limit).

⇒
```
#include <stdio.h>
int main() {
    int a, b, c, d;
    int L, n1, n2;
    printf("Enter the limit (L):");
    scanf("%d", &L);
    printf("\n Ramanujan numbers upto cube of %d:\n", L);
    for (a=1; a<=L; a++){
        for (b= a+1; b<=L; b++) {
            n1 = a*a*a + b*b*b;
            for (c=a+1; c<=L; c++){
                for (d= c+1; d<=L; d++ ){
                    n2= c*c*c + d*d*d;
                    if (n1==n2){
                        printf("%d = %d^3 + %d^3 = %d^3 +
                            %d^3\n", n1, a, b, c, d);
                    }
                }
            }
        }
    }
    return 0;
}
```

Teacher's Signature _____

Output 1:

5
5

Output 2:

Local variable = 15

Experiment 4

1)- Declare a global variable outside all functions and use it inside various functions to understand its accessibility.

⇒
```
#include<stdio.h>
int x=5;
void sahaj(){
    printf("%d",x);
}
int main(){
    sahaj();
    printf("%d",x);
    return 0;
}
```

2)- Declare a local variable inside a function and try to access it outside the function.

⇒
```
#include <stdio.h>
void abc(){
    int lv=15;
    printf("Local variable = %d", lv);
}
int main(){
    abc();
    //printf("Local variable outside function = %d", lv); // will give compilation error.
    return 0;
}
```

Teacher's Signature _____

Output 3:

Inside block: x= 10, y= 20

Output 4:

7 2 3

Date

Expt. No. Page No. 18

3)- Declare variables within different code blocks (enclosed by curly braces) and test their accessibility within and outside those blocks.

⇒
```
#include <stdio.h>
int main() {
    int x = 10;
    {
        int y = 20;
        printf("Inside block: x = %d, y = %d\n", x, y);
    }
    // printf("Outside block: x = %d, y = %d", x, y);
                                    will give compilation error!
    return 0;
}
```

4)- Declare a static variable inside a function. Observe how its value persists across function calls.

⇒
```
#include <stdio.h>
int abc() {
    static int z = 0;
    z++;
    return z;
}
int main() {
    printf("%d ", abc());
    printf("%d ", abc());
    printf("%d ", abc());
    return 0;
}
```

Teacher's Signature

Output 1:

```
Enter the range of the array: 5
Enter a element: 10
Enter a element: 5
Enter a element: 7
Enter a element: 43
Enter a element: 25
The second largest element = 25
```

Date

Expt. No. Page No. 19

Experiment -5

1)- Write a 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.

⇒
```
#include <stdio.h>
int main () {
    int n;
    printf(" Enter the range of the array: ");
    scanf(" %d", &n);
    int a[n];
    for(int i= 0; i<n; i++){
        printf(" Enter an element : ");
        scanf(" %d", & a[i]);
    }
    int temp;
    for (int i=0; i<n; i++){
       for(int j=1; j<n-1-i ; j++){
            if (a[j] >a [j+1]){
                temp= a [j];
                a[j]= a[j+1];
                a[j+1]= temp;
            }
       }
    }
    int second= a[n-2]
    printf("The second largest element = %d", second);
    return 0;
}
```

Teacher's Signature

Output 2:

```
Enter the range of the array: 6
Enter a element: 0
Enter a element: 1
Enter a element: -1
Enter a element: 2
Enter a element: 5
Enter a element: 6
Even = 2
Odd = 1
Zeroes = 1
Positive = 4
Negative = 1
```

4- Write a 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.

⇒
```
#include <stdio.h>
int main(){
    int n, odd=0, even=0, pos=0, neg=0, zero=0;
    printf("Enter the range of the array:");
    scanf("%d",&n);
    int a[n];
    for(int i=0; i<n; i++){
        printf("Enter a element:");
        scanf("%d", &a[i]);
    }
    for(int i=0; i<n; i++){
        if(a[i]%2==0){
            even+=1;
        }
        else if(a[i]%2!=0){
            odd+=1;
        }
        else if(a[i]>0){
            pos+=1;
        } else if(a[i]==0){
            zero+=1;
        } else {
            neg+=1;
        }
    }
```

P.T.O →

Teacher's Signature _____

Output 3:

```
Enter the range of the array: 5
Enter a element: 1
Enter a element: 1
Enter a element: 2
Enter a element: 3
Enter a element: 4
Enter a number to find its frequency: 1
Frequency = 2
```

```
    printf("Even = %d \n", even);
    printf("Odd = %d \n", odd);
    printf("Zeroes = %d \n", zero);
    printf("Positive = %d \n", pos);
    printf("Negative = %d \n", neg);
    return 0;
}
```

3)- Write a 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 the list.

⇒
```
#include <stdio.h>
int main() {
    int n, count=0, b;
    printf("Enter the range of the array:");
    scanf("%d", &n);
    int a[n];
    for (int i=0; i<n; i++){
        printf("Enter a element:");
        scanf("%d", &a[i]);
    }
    printf("Enter a number to find its frequency:");
    scanf("%d", &b);
    for (int i=0; i<n; i++){
        if (a[i]==b) {
            count++;
        }
    }
    printf("Frequency = %d", count);
    return 0;
}
```

Teacher's Signature _____

Output 4:

```
Enter the no. of rows for matrix A: 2
Enter the no. of columns for matrix A: 2
Enter the no. of rows for matrix B: 2
Enter the no. of columns for matrix B: 2
Input the elements of matrix A:
element - [0] [0] : 1
element - [0] [1] : 2
element - [1] [0] : 2
element - [1] [1] : 1
Input the elements of matrix B:
element - [0] [0] : 1
element - [0] [1] : 2
element - [1] [0] : 2
element - [1] [1] : 1
Matrix A (2x2):
1       2
2       1
Matrix B (2x2):
1       2
2       1
Product matrix C (2x2):
5       4
4       5
```

4)- Write a program that reads two matrices A(m x n) and B(p x q) and computes the product. Read matrix A & B ..... in case of incompatibility.

⇒

```
#include <stdio.h>
int main() {
    int m, n, p, q;
    printf("Enter the no. of rows for matrix A:");
    scanf("%d", &m);
    printf("Enter the no. of columns for matrix A:");
    scanf("%d", &n);
    printf("Enter the no. of rows for matrix B:");
    scanf("%d", &p);
    printf("Enter the no. of columns for matrix B:");
    scanf("%d", &q);
    if (n != p){
        printf("Not compatible for multiplication");
        return 0;
    }
    int a[m][n];
    int b[p][q];
    int c[m][q];
    printf("Insert the elements of matrix A:\n");
    for (int i=0; i<m; i++){
        for (int j=0; j<n; j++) {
            printf("Element-[%d][%d]:", i, j);
            scanf("%d", &a[i][j]);
        }
    }
```

Teacher's Signature _____

```
printf("Input elements of matrix B:\n");
for(int i=0; i<p; i++){
    for(int j=0; j<q; j++){
        printf("Element-[%d][%d]:", i, j);
        scanf("%d", &a[i][j]);
    }
}
printf("Matrix A (%d x %d):\n", m, n);
for(int i=0; i<m; i++){
    for(int j=0; j<n; j++){
        printf("%d ", a[i][j]);
    }
    printf("\n");
}
printf("Matrix B(%d x %d):\n", p, q);
for(int i=0; i<p; i++){
    for(int j=0; j<q; j++){
        printf("%d ", a[i][j]);
    }
    printf("\n");
}
for(int i=0; i<m; i++){
    for(int j=0; j<q; j++){
        c[i][j]=0;
    }
}
```

```
for (int i=0; i<m; i++){
    for (int j=0; j<q; j++){
        for (int k=0; k<p; k++){
            c[i][j] += a[i][k] * B[k][j];
        }
    }
}
printf("Product matrix c (%d x %d):\n", m, q);
for (int i=0; i<m; i++){
    for for (int j=0; j<q; j++){
        printf("%d  ", c[i][j]);
    }
    printf("\n");
}
return 0;
}
```

Output 1:

Factorial of 5 (recursive): 120
Factorial of 5 (non-recursive): 120

Experiment - 6

1) 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 result for different values of n and r with suitable messages.

⇒
```
#include <stdio.h>
long long int fact_recursive(int n){
        if(n==0){
                return 1;
        } else {
                return (long long int)n* fact_recursive(n-1);
        }
}

long long int non_recursive(int n){
        long long int result=1;
        for(int i=1;i<=n;i++){
                result*=i;
        }
        return result;
}
```

P.T.O →

Teacher's Signature _____

Output 2:

Enter 1st number: 10
Enter 2nd number: 20
GCD = 10

```
int main() {
    printf("Factorial of 5 (recursive): %lld\n", fact-recursive(5));
    printf("Factorial of 5 (non-recursive): %lld\n", non-recursive(5));
    printf("\n");
    return 0;
}
```

2) 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 numbers.

=>
```
#include <stdio.h>
int GCD(num1, num2){
    if (num2 == 0) {
        return num1;
    } else {
        return GCD(num2, num1 % num2);
    }
}
int main (){
    int a, b, gcd;
    printf("Enter 1st number: ");
    scanf("%d", &a);
    printf("Enter 2nd number: ");
    scanf("%d", &b);
    result = GCD(a, b);
    printf("GCD = %d\n", result);
    return 0;
}
```

Teacher's Signature _____

Output 3:

Enter number upto fibonacci series: 5
fibonacci sequence =
0 1 1 2 3 5

3) Develop a recursive function FIBO(num) that accepts an integer argument. Write a C program that invokes this function to generate the Fibonacci sequence upto num.

⇒
```
#include <stdio.h>
int FIBO(int n) {
    if (n==0) {
        return 0;
    } else if (n==1) {
        return 1;
    } else {
        return FIBO(n-1) + FIBO(n-2);
    }
}
int main() {
    int num;
    printf("Enter number upto fibonacci series:");
    scanf("%d", &num);
    printf("Fibonacci sequence = \n");
    for (int i = 0; i <= num; i++) {
        printf("%d", FIBO(i));
    }
    return 0;
}
```

Teacher's Signature _____

Output 4:

```
Enter lower limit : 5
Enter upper limit : 10
Prime numbers =
5  7
```

4)- Develop a C function ISPRIME(num) that accepts an integer argument and return 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.

⇒
```
#include <stdio.h>
#include <math.h>
int ISPRIME(int num){
    if (num <=1) {
        return 0;
    } if (num == 2) {
        return 1;
    } if (num %. 2 == 0){
        return 0;
    }
    for (int i= 3; i <= sqrt(num); i+=2) {
        if (num %.i ==0) {
            return 0;
        }
    }
    return 1;
}
```

P. T. O →

Teacher's Signature _____

Output 5:

```
Enter a string: sahoj
Original string: sahoj
Reversed string: jahas
```

```
int main() {
    int low, up;
    printf("Enter lower limit: ");
    scanf("%d", &low);
    printf("Enter upper limit: ");
    scanf("%d", &up);
    printf("Prime numbers = \n");
    for (int i = low; i <= up; i++) {
        if (ISPRIME(i)) {
            printf("%d ", i);
        }
    }
    printf("\n");
    return 0;
}
```

5) Develop a function REVERSE(str) that accepts a string argument. Write a C program that invokes this function to find the reverse of a given string.

⇒

```
#include <stdio.h>
#include <string.h>
void REVERSEC(char *str){
    int len = strlen(str);
    char temp;
    for (int i=0; int j = len-1; i<j; i++, j--) {
        temp = str[i];
        str[i] = str[j];
        str[j] = temp;
    }
}
```

Teacher's Signature _____

Expt. No. _____

Page No. 30

```
int main() {
    char str[100];
    printf("Enter a string:");
    scanf("%s", str);
    printf("Original string = %s\n", str);
    REVERSE(str);
    printf("\nReversed string = %s", str);
    return 0;
}
```

Teacher's Signature _____

Output 1:

Integer variable:
Value = 10
Address = 0x7ffd45a2e6c4
Integer pointer:
Value = 0x7ffd45a2e6c4
Value pointed = 10

Float variable:
Value = 3.14
Address = 0x7ffd45a2e6c8
Float pointer:
Value = 0x7ffd45a2e6c8
Value pointed = 3.14

Character variable:
Value = A
Address = 0x7ffd45a2e6cb
Character pointer:
Value = 0x7ffd45a2e6cb
Value pointed = A

Expt. No. _____ Date _____

Page No. 31

Experiment - 8

1)- Declare different types of pointers (int, float, char) and initialize them with the addresses of variables. Print the values of both the pointers and the variables they point to.

⇒
```
#include<stdio.h>
int main () {
    int a = 10;
    int *ptr1 = &a;
    float b = 3.14;
    float *ptr2 = &b;
    char c = 'A'
    char *ptr3 = &c;
    printf("Integer Variable:\n");
    printf("Value = %d\n", a);
    printf("Address = %p\n", &a);
    printf("Integer Pointer:\n");
    printf("Value = %p\n", ptr1);
    printf("Value pointed = %d\n", *ptr1);

    printf("Float Variable:\n");
    printf("Value = %f\n", b);
    printf("Address = %p\n", &b);
    printf("Float Pointer:\n");
    printf("Value = %p\n", ptr2);
    printf("Value pointed = %f\n", *ptr2);
```

P.T.O →

Teacher's Signature _____

Output 2:

p= 1099521652
p++ = 1099521656
p-- = 1099521652
q= 1099521648
q++= 1099521652
q--= 1099521648
r= 1099521647
r++ = 1099521648
r-- = 1099521647

```
printf("Character variable:\n");
printf("Value = %c\n", c);
printf("Address = %p\n", &c);
printf("Character pointer:\n");
printf("Value = %p\n", ptr3);
printf("Value pointed to = %c\n", *ptr3);
return 0;
}
```

2) Perform pointer arithmetic (increment and decrement) on pointers of different data types. Observe how the memory addresses change and the effects on data access.

⇒
```
#include <stdio.h>
int main() {
    int a = 22;
    int *p = &a;
    printf("p = %p\n", p);
    p++;
    printf("p++ = %u\n", p);
    printf("p-- = %u\n", p--);
    float b = 3.14;
    float *q = &b;
    printf("q = %u\n", q);
    printf("q++ = %u\n", q++);
    printf("q-- = %u\n", q--);
    char c = 'a';
    char *r = &c;
    printf("r = %u\n", r);
    printf("r++ = %u\n", r++);
    printf("r-- = %u\n", r--);
    return 0;
}
```

Teacher's Signature _____

Output 3:

Before modification: x=5, y=7
After modification: x=15, y=27

Date

Expt. No. Page No. 33

3) Write a function that accepts pointers as parameters. Pass variables by reference using pointers and modify their values within the function.

⇒
```
#include<stdio.h>
void modify(int *a, int *b){
    *a = *a + 10;
    *b = *b + 20;
}
int main(){
    int x=5, y=7;
    printf("Before modification: x=%d, y=%d\n", x, y);
    modify(&x, &y);
    printf("After modification: x=%d, y=%d\n", x, y);
    return 0;
}
```

Teacher's Signature

Output 1:

```
For 1st complex number
Enter the real and imaginary parts: 2 4
For 2nd complex number
Enter the real and imaginary parts: 4 2
Sum = 6 + 6
Difference = -2 + 2
```

Date _____

Expt. No. _____ Page No. 34

Experiment - 7

1)- Write a C program that uses functions to perform the following operations:
a)- Reading a complex number.
b)- Writing a complex number.
c)- Addition and substraction of two complex numbers.

⇒
```
#include<stdio.h>
typedef struct complex{
        float real;
        float imag;
}complex; complex add(complex n1, complex n2); complex sub(complex n1, complex n2);
int main(){
        complex n1, n2, result;
        printf("For 1st complex number \n");
        printf("Enter the real and imaginary parts:");
        scanf("%f %f", &n1.real, &n2.imag);
        printf("For 2nd complex number \n");
        printf("Enter the real and imaginary parts:");
        scanf("%f %f", &n2.real, &n2.imag);
        result= add(n1, n2);
        b = sub(n1, n2);
        printf("Sum=%.1f+%.1f", result.real, result.imag);
        printf("Difference=%.1f+%.1f", b.real, b.imag);
        return 0;
}
```

Teacher's Signature _____

Output-2:

For 1 employee:

Name: ABC
Basic pay = 22000

Employee name	Gross salary
ABC	44000

```
complex add (complex n1, complex n2) {
        complex temp;
        temp.real = n1.real + n2.real;
        temp.imag = n1.imag + n2.imag;
        return (temp);
}
complex sub (comple n1, complex n2) {
        complex c;
        c.real = n1.real + n2.real;
        c.imag = n1.imag + n2.imag;
        return (c);
}
```

2)- 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 employees name and gross salary.

⇒
```
#include <stdio.h>
Struct employee {
        char name[100];
        float bp;
        float da;
        float gs; };
int main () {
        struct employee e[100];
        printf(" Enter the Employee details : ");
        for (int i = 0; i < 100; i++) {
                printf(" Name : ");
                scanf("%s", e[i].name);
                printf(" Basic pay : ");
                scanf("%f", &e[i].bp);
```

Teacher's Signature

P.T.O →

Output 3:

Book ID : 001
Title: Harry Potter
Author: J. K. Rowling
Price: 430

```
e[i].da = e[i].bp * 0.52;
e[i].gs = e[i].bp + e[i].da; }

printf("\n %.s | %.s\n", "Employee Name", "Gross Salary");
for (int i=0; i<100; i++){
    printf("%.s|%.f\n", e[i].iname, e[i].gs);
}
return 0; }
```

3)- Create a book structure containing book-id, title, autor name and price. Write a C program to pass a structure as a fn. argument and print it

⇒

```
#include <stdio.h>
#Include <string.h>
Struct book {
    int id; char title[100];
    char author[100]; float price; };
void print(struct book b) {
    printf("Book ID: %d\n", b.id);
    printf("Title: %s\n", b.title);
    printf("Author: %s\n", b.author);
    printf("Price: %f\n", b.price); }

int main() {
    struct book b;
    b.id = 001;
    strcpy(b.title, "Harry Potter");
    strcpy(b.author, "J.K. Rowling");
    b.price = 430;
    print(b);
    return 0;
}
```

Teacher's Signature _____

Output 4

abc
zyz road
def road
ahmedabad
gujarat
380058

Date

Expt. No. Page No. 37

4)- Create a union containing 6 strings: name, home_address, hostel_address, city, state and zip- Write a C program to display your present details.

=>
```
#include <stdio.h>
#include <string.h>
union address {
    char name[20]; char home_address[30]; char hostel_address[30];
    char city[10]; char state[10]; int zip; } a;
void main() {
    strcpy(a.name, "abc");
    strcpy(a.home_address, "2y2 road");
    strcpy(a.hostel_address; "def road");
    strcpy(a.city, "ahmedabad");
    strcpy(a.state, "gujarat");
    a.zip = 380058;
    printf("%s\n", a.name); printf("%s\n", a.home_address);
    printf("%s\n", a.hostel_address); printf("%s\n", a.city);
    printf("%s\n", a.state); printf("%d\n", a.zip); }
```

Teacher's Signature

Output 2:

This is a test file.

Output 3:

Hello!
Sahaj this side.

Experiment - 9

1)- Write a program to create a new file and write text into it.

```
#include <stdio.h>
int main() {
    FILE *fp;
    fp=fopen("abc.txt","w");
    fprintf(fp,"Text");
    fclose(fp);
    return 0; }
```

2)- Open an existing file and read its content character by character.

```
#include<stdio.h>
int main() {
    FILE *fp;
    char ch;
    fp=fopen("abc.txt","r");
    if(fp==NULL){
        printf("Empty"); }
    while((ch=fgetc(fp))!=EOF){
        printf("%c",ch); }
    fclose(fp); return 0; }
```

3)- Open a file, read its content line by line and display each line.

```
#include <stdio.h>
void main() {
    FILE *fp=fopen("abc.txt","r"); char c[256];
    if(fp!=NULL){
        while(fgets(c,sizeof(c),fp){
            printf("%s",c);
    } fclose(fp); }
```

Teacher's Signature _____

P.T.O →

Output 1:

Linked List = 10 20 30

```
else {
    printf("Empty"); }
}
```

Experiment – 10

Q1) Write a program to create a single linked list in C using pointer & structure

```
#include <stdio.h>
typedef struct abc {
        int a;
        struct abc *b; } abc;
int main() {
        abc *c = (abc *) malloc (size of (abc));
        c -> a = 10;
        abc *d = (abc *) malloc (size of (abc));
        d -> a = 20;
        abc *e = (abc *) malloc (size of (abc));
        e -> a = 30;
        c -> b = d;
        d -> b = e;
        e -> b = NULL;
        printf("Linked list = ");
        abc* temp = c;
        while (temp) {
                printf("%d", temp -> a);
                temp = temp -> b;
        }
        return 0;
}
```

Teacher's Signature _____

Output 2:

1 2 3 4 5

2)- Write a program to insert item in middle of the linked list.

→
```
#include <stdio.h>
Struct Node {
    int data;
    Struct Node *nxt; };
struct Node * create (int x);
struct Node * insert ( struct Node *head, int x) {
    if (heed == NULL) {
        Return creat (x); }
    Struct Node * new = create(x);
    struct Node * current = head;
    int l = 0;
    while ( current != NULL ) {
        l++; current = current -> nxt;
    }
    int mid = (l % 2 = 0) ? l/2 : ( l+1) / 2;
    current = head;
    while ( mid > 1){
        current = current -> nxt;
        mid--;
    }
    new -> nxt = current -> nxt;
    current -> nxt = new;
    return head;
}
Void print (struct Node *head) {
    struct Node * current = head;
    while (current != NULL){
        printf(" %d", current -> data);
        current = current -> nxt;
    } printf("\n");
}
```

Teacher's Signature

Expt. No. _____ Date _____

Page No. 41

```
struct Node * create (int x) {
    struct Node * new = (struct Node*) malloc (sizeof( struct Node ));
    new -> data = x;
    new -> next = NULL;
    return new; }

int main () {
    struct Node * head = create (1);
    head -> nxt = create (2);
    head -> nxt -> nxt = create (4);
    head -> nxt -> nxt -> nxt = create (5);
    int x = 3;
    head = insert ( head, x );
    printf( head);
    return 0;
}
```

Teacher's Signature _____

Output 1:

Value of PI = 3.14

Output 2:

Square of n1 is: 25
Square of n2 is: 100

Expt. No. _____ Page No. 1 Date □□□□□□

Experiment - 12

1)- Write a program to define some constant variable in preprocessor.

⇒
```
#include <stdio.h>
#define PI 3.14
int main() {
    printf("Value of PI = %.f\n", PI);
    return 0;
}
```

~~2)- Write a function to define multiple macros to perform arithmetic.~~

2)- Write a program to define a function in directives.

⇒
```
#include <stdio.h>
#define square(x) ((x)*(x))
int main() {
    int n1 = 5;
    int n2 = 10;
    int sq = square(n1);
    int a = square(n2);
    printf("Square of n1 is: %d\n", n1);
    printf("Square of n2 is: %d\n", n2);
    return 0;
}
```

Teacher's Signature _____

Output:

Enter two numbers: 10 2

Addition = 12

Subtraction = 8

Product = 20

Division = 5

Expt. No. _____ Page No. 3 Date □□□□□□

Experiment - 13

1) Write a program to define multiple macro to perform arithmetic functions.

⇒
```
#include <stdio.h>
#define add(x,y) ((x)+(y))
#define sub(x,y) ((x)-(y))
#define mult(x,y) ((x)*(y))
#define divide(x,y) ((x)/(y))
int main(){
    int n1, n2;
    int sum, diff, pro;
    float q;
    printf("Enter two numbers: ");
    scanf("%d %d", &n1, &n2);
    sum = add(n1, n2);
    diff = sub(n1, n2);
    pro = mult(n1, n2);
    q = divide(n1, n2);
    printf("Addition = %d", sum);
    printf("Subtraction = %d", diff);
    printf("Product = %d", pro);
    printf("Division = %d", q);
    return 0;
}
```

Teacher's Signature _____

Output:

Sum=17
Difference=1
Product=72

Expt. No. _____ Page No. 4 Date □□□□□□

Experiment-14

1)- Write a program to create static library for performing arithmetic function

→ ① arith.c:

```
#include "arith.h";
# include <stdio.h>;
int sum( int a, int b) {
        return(a+b);
}
int sub(int a, int b) {
        return(a-b);
}
int mul( int a, int b) {
        return (a* b);
}
```

② arith.h:

```
# ifndef ARITH
# define ARITH
int sum (int a, int b);
int sub ( int a, int b);
int mul (int a, int b);
# endif
```

Teacher's Signature _____

Expt. No. _____ Page No. 5 Date □□□□□□

③
```
#include<stdio.h>
#include "arith.h"
int main() {
    int a=4, b=3;
    printf("Sum = %d\n", sum(a,b));
    printf("Difference = %d\n", sub(a,b));
    printf("Product = %d\n", mul(a,b));
    return 0; }
```

commands to run:

→ gcc -c arith.c

→ ar rcs libarith.a arith.o

→ gcc main.c -L. -larith -o main

→ ./main.

Teacher's Signature _____

Expt. No. _____ Page No. 6 Date □□□□□□

Experiment-15

→ Same code as last experiment.

Commands to run shared library:

→ gcc -fPIC -c arith.c

→ gcc -shared -o libarith.so arith.o

→ gcc main.c -L. -larith -o main

→ ./main.

Teacher's Signature _____