Experiment 1 - Installation, Enviornment Setup & Starting With C Language

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

Program:
```
#include <stdio.h>

int main () {
    printf ("Hello World \n")
    return 0;
}
```

2. Write a C program to print the address in multiple line (new

Prog:
```
#include <stdio.h>
  int main () {
      printf ("Sector 37 \n");
      printf ("Krishna Puram Ward 97 \n");
      printf ("Dehradun 248001 \n");
      printf ("India \n");
      return 0;
  }
```

3. Write a program that prompts the user to enter their name & age.

```
# include <studio.h>
  int main (){

   int age;
   printf ("Enter your name :");
   scanf ("%s" & name);
   printf ("Enter your age :");
   scanf ("%d" & age);
   printf ("Your name is %s and You are %d years old /n" name, age);
   return 0;
  }
```

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

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

Experiment 2 - Operator

1. Write a C program to calculate the area & perimeter of rectang based on its length & width.

```
# include < stdio.h >
  float main() {
  float length, width, area, peri ;
      printf ("Enter the length : \n");
      scanf ("% f",& width);
      printf ("Enter the width" \n");
      scanf ("% f", & width);
      area = length * width ;
      peri = 2* (length + width);
      printf ("The area of rectangle is = % f \n" area);
      printf ("The perimeter of rectangle is = % f \n", peri);
      return 0 ;
}
```

2. Write a C program to convert temperature from Celcius to Farenhe using the formula F = (C * 9/5) + 32

```
# include < stdio.h >
  float main () {
      printf ("Enter temperature in Celcius : \n");
      scanf ("% f" , Celcius) ;
      f = (Celcius * 9/5) + 32 ;
      printf ("Given timperature in farenheit is = % f \n ", f);
      return 0 ; }
```

Experiment 11 : BITWISE OPERATOR

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

```
# include <stdio.h>
  int main () {
  // bit level operator
  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 not of a is %d \n", ~a );
  return 0;
}
```

2. Write a program to apply left shift & right shift operator.

```
# include <stdio.h>
  int main () {
  int a = 5
  int b = 7
  printf ("The left shift operator of a %d \n", a <<1);
  printf ("The left shift operator of b %d \n", b<< 1);
  printf ("The right shift operator of a %d \n", a>>1);
  printf ("The right shift operator of b %d \n", b >>1);
  return 0;
}
```

Date _____ Page No. _____

```
printf("Enter a string : ");
fgets(myString, sizeof(myString), stdin);

myString[strespn(myString, "\n")] = 0;

printf("Original String : %s\n", myString);

REVERSE(myString);

printf("Reversed string : %s\n", myString);
    return 0;
}
```

Mohan

Teacher's Signature _____

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 isoceles, 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) ||
```

Teacher's Signature :

```
        (l1*l1 + l3*l3 = l2*l2)) {
    printf ("It is a right triangle");
}
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 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;
print ("BMI Index is: %.f", bmi);

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

Teacher's Signature :

```
    return 0;
}
```

3- Write a program to check if three points (x_1, y_1) (x_2, y_2) (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");
    }
```

Teacher's Signature :

```
    return 0;
}
```

4- 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", &l2);
    printf("Enter breadth of 3rd rectangle: ");
```

Teacher's Signature :

```
scanf("%d", &b3);

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

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

return 0;
}
```

5. According to the gregorian calender, 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 in that year.

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

    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;

if(day==0){
  printf("monday");
}
else if(day==1){
  printf("Tuesday");
}
else if(day==2){
  printf("wednesday");
}
else if(day==3){
  printf("Thursday");
}
```

Teacher's Signature : _____

```
else if (day == 4) {
    printf ("friday");
}
else if (day == 5) {
    printf ("saturday");
}
else if (day == 6) {
    printf ("sunday");
}
return 0;
}
```

Experiment - 3.2

1) Write a program to enter numbers till the user wants. At the end, it should display the count of +ve, -ve & 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;
}
```

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
```

Teacher's Signature :

```
#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;
}
```

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 each year in the last decade.

```
#include <stdio.h>
int main() {
    float population = 100000;
    float rate = 0.10;
    int years = 10;
    for (int i = 1; i <= years; i++) {
```

Teacher's Signature :

```
        population = population * (1 + rate);
        printf ("End of year %d population will be
                = %0.d", i, population)
    }
    return 0;
}
```

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 up to a reasonable limit.

Example of Ramanujan number : 1729
$12^3 + 1^3$ & $10^3 + 9^3$. for a number L = 20 (that is limit)

```
#include <stdio.h>
#include <math.h>
# define LIMIT 100000

int main () {
    int count_sums [LIMIT] = {0};
    int sum_pairs [LIMIT] [2] = {0};
    int i, j, sum;

for (i=1; i<47; i++) {
    sum = i * i * i + j * j * j;
    if (sum < LIMIT) {
```

Teacher's Signature : _____

```
        if (count_sums[sum] == 1) {
            printf("%d = %d^3 + %d^3 and %d^3 + %d^3\n",
                   sum, sum_pairs[sum][0], sum_pairs[sum][1], i, j);
        }
        if (count_sums[sum] == 0) {
            sum_pairs[sum][0] = i;
            sum_pairs[sum][1] = j;
        }
        count_sums[sum]++;
      }
    }
  }
  return 0;
}
```

Expt. No. _____

Page No. 11

Experiment - 4

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

```
#include <stdio.h>
int global_variable = 100;
void modify_global () {
    global_variable = 200;
    printf ("Inside modify_global(): global_variable = %d\n",
            global_variable);
}
void accessGlobal () {
    printf ("Inside accessGlobal(): global_variable = %d\n", global_variable);
}
int main() {
    printf ("Inside main(): Initial global_variable = %d\n", global_variable);
    modifyGlobal();
    printf ("Inside main(): global_variable after modifyGlobal() call =
            %d\n", global_variable);
    accessGlobal();
    global_variable = 30;
    printf ("Inside main(): global_variable after direct modification
        = %d\n", global_variable);
    return 0;
}
```

Teacher's Signature : _____

Expt. No. _____

Page No. 20

2- Declare a local variable inside a function and try to access it outside the function. Compare this with accessing the global variable from within the function.

```
#include <stdio.h>

int global_var = 10;
void my_function() {
    int local_var = 20;
    printf("Inside my_function:\n");
    printf("Accessing global_var: %d\n", global_var);
    printf("Accessing local_var: %d\n", local_var);
}

int main() {
    printf("Inside main:\n");
    printf("Accessing global_var: %d\n", global_var);

    my_function();

    return 0;
}
```

Teacher's Signature : _____

Expt. No. _____

Page No. 21

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

```
#include <stdio.h>

int main() {
  int global_to_main = 10;
  printf("Inside main block: global_to_main = %d\n", global_to_main);
  {
    int inner_block_var1 = 20;
    printf("Inside inner block 1: global_to_main = %d, inner_block_var1
        = %d\n", global_to_main, inner_block_var1);

    int inner_block_var2 = 30;
    printf("Inside inner block 2: global_to_main = %d,
        inner_block_var1 = %d, inner_block_var2 = %d\n",
        global_to_main, inner_block_var1, inner_block_var2);
  }
  return 0;
}
```

Teacher's Signature : _____

Expt. No. _____ Date _____

Page No. 22

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

```
#include <stdio.h>

void counterfunction() {
    static int callcount = 0;
    callcount++;
    printf("function called %d times.\n", callcount);
}

int main() {
    counterfunction();
    counterfunction();
    counterfunction();
    return 0;
}
```

Teacher's Signature : _____

Expt. No. _____ Date _____

Page No. 23

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 of integers.

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

int main() {
    int n;
    printf("Enter the number of elements in the array:");
    Scanf("%d", &n);

    if (n<2) {
    printf("Invalid input: The array must contain at least
            two elements to find the second largest.\n");
    return 1;
    }
int arr[n];

printf("Enter %d integers:\n", n);
```

Teacher's Signature : _____

```
for (int i = 0; i<n ; i++) {
    printf ("Element %d : ", i+1);
    Scanf ("%d", &arr[i]);
}

int firstLargest = INT_MIN;
int SecondLargest = INT_MIN;

for (int i=0; i<n; i++) {
    if (arr[i] > firstLargest) {
        secondLargest = firstLargest;
        firstLargest = arr[i];
    } else if (arr[i] > SecondLargest && arr[i] != firstLargest) {
        secondLargest = arr[i];
    }
}

if (secondLargest == INT_MIN) {
    printf("There is no distinct second largest element");
} else {
printf("The second largest element in the array is :
        %d\n", secondLargest);
    }
    return 0;
}
```

Teacher's Signature :

2- Write a program to read a list of integers & store it in a single dimensional array. Write a C program to count and display +ve, -ve, odd & even numbers in an array.

```
#include <stdio.h>
int main () {
    int size, i;
    int positive_count = 0;
    int negative_count = 0;
    int odd_count = 0;
    int even_count = 0;

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

int arr[size];

printf ("Enter %d integers: \n", size);
 for (i = 0; i < size; i++) {
   scanf ("%d", &arr[i]);
 }
for (i = 0; i < size; i++) {
   if (arr[i] > 0) {
      positive_count++;
   } else if (arr[i] < 0) {
```

```
            negative_count++;
        }
        if (arr[i] % 2 == 0) {
            even_count++;
        } else {
            odd_count++;
        }
    }
    printf("\n--- Analysis of numbers ---\n");
    printf("Positive numbers: %d\n", positive_count);
    printf("Negative numbers: %d\n", negative_count);
    printf("Odd numbers: %d\n", odd_count);
    printf("even numbers: %d\n", even_count);

    return 0;
}
```

3- Write a program to read a list of integers and store it in a single dimensional array. Write a C programm to find the frequency of a particular number in a list of integers.

```
#include <stdio.h>
int main() {
    int n, i;
```

Expt. No. _____ Date _____

Page No. 27

```
printf("Enter the number of elements in the array: ");
scanf("%d", &n);

int arr[n];

printf("Enter %d integers:\n", n);
for (i = 0; i < n; i++) {
    printf("Enter element %d: ", i+1);
    scanf("%d", &arr[i]);
}

int search_num;
printf("Enter the no. to find its frequency: ")
scanf("%d", &search_num);

int frequency = 0;
for (i=0; i<n; i++) {
    if (arr[i] == search_num) {
        frequency++;
    }
}
printf("The no. %d appears %d times in the array.\n",
        search_num, frequency);

return 0;
}
```

4- Write a program to read two matrices A (m x n) and B (p x q) and computes the product A & B. Read matrix A and matrix B in row major order respectively. Print both the input matrices & resultant matrix with suitable headings & output should be in matrix formal only. Program must check the compatibilty of order of the matrices for multiplication. Report appropriate message in case of incompatibility.

```
#include <stdio.h>
int main () {
    int m, n, p, q;

printf ("Enter the number of rows & columns for matrix A: ");
scanf ("%d%d", &m, &n);

printf ("Enter the number of rows & columns for matrix B: ");
scanf ("%d%d", &p, &q);

if (n != p) {
printf ("Matrix multiplication is not possible. The number
        of columns in the first matrix must be equal to
        the number of rows in the second matrix.\n");
return 1;
}
```

Teacher's Signature :

```
int matrixA[m][n];
int matrixB[p][q];
int resultMatrix[m][q];

printf("\n Enter elements of matrix A (%d x %d): \n", m, n);
for (int i = 0; i < m; i++) {
  for (int j = 0; j < n; j++) {
    scanf("%d", &matrixA[i][j]);
  }
}

printf("\n Enter elements of matrix B (%d x %d): \n", p, q);
for (int i = 0; i < p; i++) {
  for (int j = 0; j < q; j++) {
    scanf("%d", &matrixB[i][j]);
  }
}

for (int i = 0; i < m; i++) {
  for (int j = 0; j < q; j++) {
    resultmatrix[i][j] = 0;
  }
}
for (int i = 0; i < m; i++) {
  for (int j = 0; j < q; j++) {
    for (int k = 0; k < n; k++) {
```

Teacher's Signature :

Expt. No. _____ Date _____

Page No. 30

```
            resultMatrix[i][j] += matrixA[i][k] * matrixB[k][j];
        }
    }
}
printf("\nMatrix A:\n");
for(int i=0; i<m; i++){
    for(int j=0; j<n; j++){
        printf("%d\t", matrixA[i][j]);
    }
    printf("\n");
}

printf("\nMatrix B:\n");
for(int i=0; i<p; i++){
    for(j=0; j<q; j++){
        printf("%d\t", matrixB[i][j]);
    }
    printf("\n");
}
printf("\n resultant matrix (A x B):\n");
for(int i=0; i<m; i++){
    for(int j=0; j<q; j++){
        printf("%d\t", resultMatrix[i][j]);
    }
    printf("\n");
}
return 0;
}
```

Teacher's Signature : _____

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 results for diff. values of n & 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 fact_non_recursive(int n) {
    long long int result = 1;
    int i;
    for (i = 1; i<=n; i++) {
        result *= i;
    }
```

```
    return result;
}

long long int binomial_coefficient(int n, int r) {
    if (r < 0 || r > n) {
        return 0;
    }
    return fact_non_recursive(n) / (fact_non_recursive(r) *
        fact_non_recursive(n - r));
}

int main() {
    int n_values[] = {5, 6, 7, 8, 9};
    int r_values[] = {2, 3, 4};
    int i, j;

printf("--- factorial Calculation trampples ----\n");
printf("Number | Recursive fact | Non-Recursive fact\n");
printf("---------------\n");
    for (i = 0; i < 5; i++) {
        int num = n_values[i];
    printf("%6d | %14lld | %18lld\n", num, fact_recursive(num),
            fact_non_recursive(num));
    }
printf("\n");
printf("--- Binomial Coefficient C(n,r) Tabulation---\n");
printf("n | r | C(n,r)\n");
printf("-----------\n");
```

```
for (i = 0; i < size of (n_values) / size of (n_values[0]); i++) {
  for (j = 0; j < size of (r_values) / size of (r_values[0]); j++) {
  int n = n_values [i];
  int r = r_values [j];
  if (r <= n) {
     printf ("%2d | %1d | %7lld \n", n, r, binomial_coefficient
           (n, r));
     }
  }
}
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 given inputs.

```
#include <stdio.h>

int GCD (int num1, int num2) {
   if (num2 == 0) {
      return num1;
```

```
    else {
        return GCD(num2, num1 % num2);
    }
}

int main() {
    int number1, number2;

    printf("Enter the first integer: ");
    scanf("%d", &number1);

    printf("Enter the second integer: ");
    scanf("%d", &number2);

    printf("The Greatest common division of %d
            and %d is: %d\n", number1, number2,
            GCD(number1, number2));

    return 0;
}
```

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 up to num.

equence :-

sequence:)

```
#include <stdio.h>

int fIBO (int num) {
    if (num == 0) {
        return 0;
    } else if (num == 1) {
        return 1;
    } else {
        return fIBO(num-1) + fIBO(num-2);
    }
}

int main() {
    int n, i;

    printf("Enter the number of terms for the fibonacci
            sequence: ");
    scanf("%d", &n);

    if (n<0) {
        printf("Please enter a non-negative number of terms.\n");
    } else {
        printf("fibonacci sequence up to %d terms:\n", n);
        for (i=0; i<n; i++) {
            printf("%d ", fIBO(i));
        }
```

```
        printf ("\n");
    }
    return 0;
}
```

4. 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 b/w 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;
}
int main() {
    int start_range, end_range;
printf("Enter the starting number of the range: ");
scanf("%d", &start_range);
printf("Enter the ending number of the range: ");
scanf("%d", &end_range);

if (start_range > end_range) {
    int temp = start_range;
    start_range = end_range;
    end_range = temp;
}
printf("Prime numbers b/w %d & %d are:\n", start_range,
    end_range);

for (int i = start_range; i <= end_range; i++) {
    if (ISPRIME(i)) {
        printf("%d\n", i);
    }
}
return 0;
}
```

Teacher's Signature : _____

Expt. No. _____ Date _____

Page No. 38

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 REVERSE (char str[]) {
    int length = strlen (str);
    int start = 0;
    int end = length - 1;
    char temp;

    while (start < end) {
        temp = str [start];
        str [start] = str [end];
        str[end] = temp;
        start ++;
        end --;
    }
}

int main() {
    char myString[100];
```

```
printf ("Enter a string : ");
fgets (myString, sizeof (myString), stdin);

myString [strcspn(myString, "\n")] = 0;

printf ("Original string : %s\n", myString);

REVERSE (myString);

printf ("Reversed string : %s\n", myString);

    return 0;
}
```

Expt. 7

Page No.

Experiment-7

Structures & Union

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 Subtraction of two complex numbers.

& NOTE: represent complex number using a structure.

```
#include <stdio.h>

struct Complex readComplex() {
    struct Complex c;
    printf("Enter real part: ");
    scanf("%f", &c.imag);
    return c;
}

void writeComplex(struct Complex c) {
    printf("%.2f + %.2fi\n", c.real, c.imag);
}

struct Complex add(struct Complex a, struct Complex b) {
```

```
    struct Complex result;
    result.real = a.real + b.real;
    result.imag = a.imag + b.imag;
    return result;
}

struct Complex subtract(struct Complex a, struct Complex b) {
    struct Complex result;
    result.real = a.real - b.real;
    result.imag = a.imag - b.imag;
    return result;
}

int main() {
    struct Complex c1, c2, sum, diff;

    printf("Enter first complex number:\n");
    c1 = readComplex();

    printf("Enter Second complex number:\n");
    c2 = readComplex();

    sum = add(c1, c2);
    diff = subtract(c1, c2);
```

```
printf ("\n first number : ");
write complex (c1);

printf (" Second number: ");
write Complex (c2);

printf ("\n Sum : ");
write Complex (sum);

printf ("Difference: ");
write Complex (diff);

return 0;
}
```

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 & gross salary.

```
#include <stdio.h>

struct Employee {
    char name[50];
    float basic;
```

```
int main() {
    struct Employee emp[100];
    int i;

    for (i=0; i<100; i++) {
        printf("Enter name of employee %d: ", i+1);
        scanf("%s", emp[i].name);

        printf("Enter basic pay of %s: ", emp[i].name);
        scanf("%f", &emp[i].basic);

        emp[i].grow = emp[i].basic + (0.52 * emp[i].basic);
    }

    printf("\n Employee Name \t Gross Salary \n");
    for (i=0; i<100; i++) {
        printf("%s \t \t %.2f \n", emp[i].name, emp[i].grow);
    }

    return 0;
}
```

Expt. No. _____ Date _____

Page No. _____

3- 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.

```
#include <stdio.h>

struct Book {
    int book_id;
    char title[50];
    char author[50];
    float price;
};

void displayBook(struct Book b) {
    printf("\n---Book details---\n");
    printf("Book ID: %d\n", b.book_id);
    printf("Title: %s\n", b.title);
    printf("Author: %s\n", b.author);
    printf("Price: %.2f\n", b.price);
}

int main() {
    struct Book b1;

    printf("Enter Book ID: ");
    scanf("%d", &b1.book_id);
```

Expt. No. _____ Date _____

Page No. _____

```
    printf("Enter book Title : ");
    scanf("%[^\n]", b1.title);

    printf("Enter Author name : ");
    scanf("%[^\n]", b1.author);

    printf("Enter Price : ");
    scanf("%f", &b1.price);

    displayBook(b1);

    return 0;
}
```

4- Create a union containing 6 strings : name, home-address, hostel-address, city, state and zip. Write a C program to display your present address.

```
#include <stdio.h>

union Address {
    char name[50];
    char home-address[100];
    char hostel-address[100];
    char city[50];
```

```
    char state[50];
    char zip[10];
};

int main() {
    union Address addr;

    printf("Enter your Present Address : ");
    scanf("%[^\n]", addr.home_address);

    printf("\n Your Present Address is :\n");
    printf("%s\n", addr.home_address);

    return 0;
}
```

Experiment - 8

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

```
#include <stdio.h>
int main () {
    int intVar = 10;
    float floatVar = 20.5f;
    char charVar = 'A';

    int *intPtr;
    float *floatPtr;
    char *charPtr;

    intPtr = &intVar;
    float Ptr = &floatVar;
    charPtr = &charVar;

printf ("Value of intVar : %d \n", intVar);
printf ("Value of floatVar : %.2f \n", floatVar);
printf ("Value of charVar : %c \n\n", charVar);
```

Teacher's Signature :

Date

Expt. No. _____ Page No. 41

```
printf ("Value of intPtr (address of intVar) : %p\n", (void*) intPtr);
printf ("Value of floatPtr (address of floatVar) : %p\n", (void*) floatPtr);
printf ("Value of charPtr (address of charPtr) : %p\n", (void*) charPtr);

printf ("Value pointed to by intPtr : %d\n", *intPtr);
printf ("Value pointed to by floatPtr : %.2f\n", *floatPtr);
printf ("Value pointed to by charPtr : %c\n", *charPtr);

return 0;
}
```

Expt. No. Date

Page No.

Experiment - 9

File Handling in C.

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

```
#include <stdio.h>

int main () {
    FILE * fp;
    char text [100];

    fp = fopen ("myfile.txt", "w");

    if (fp == Null) {
        printf ("Error opening file!");
        return 1;
    }

    printf ("Enter text to write into the file:");
    scanf ("%[^\n]", text);

    fprintf (fp, "%s", text);

    fclose (fp);
```

Expt. No. _____ Date _____

Page No. _____

```
    printf ("\n file created and text written Sucessfully!\n");
    return 0;
}
```

2. Open an existing file & read its content character by character, and then close the file.

```
#include <stdio.h>

int main () {
    FILE *fp;
    char ch;

    fp = fopen ("myfile.txt", "r");

    if (fp == NULL) {
        printf ("Error: file not found!");
        return 1;
    }
    printf ("file content:\n");

    while ((ch = fgetc (fp)) != EOF) {
        putchar (ch);
    }
    fclose (fp);
    return 0;
}
```

Teacher's Signature : _____

Expt. No.

Page No.

Experiment - 10

Dynamic Memory Allocation

1- Write a program to create a simple linked list in C using pointer and structure.

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

struct Node {
    int data;
    struct Node* next;
};

int main () {
    struct Node * head, * second, * third;

    head = (struct Node*) malloc (Size of (struct Node));
    second = (struct Node *) malloc (Size of (struct Node));
    third = (struct Node *) malloc (Size of (struct Node));

    head -> data = 10;
    head -> next = second;
```

Teacher's Signature :

Expt. No. _____ Date _____

Page No. _____

```
second -> data = 20;
second -> next = third;

third -> data = 30;
third -> next = NULL;

struct Node *temp = head;
printf ("Linked list : ");
while (temp != NULL) {
    printf ("%d -> ", temp -> data);
    temp = temp -> next;
}
printf ("NULL \n");
return 0;
}
```

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

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

struct Node {
    int data;
    struct Node *next;
};
```

Teacher's Signature : _____

Expt. No. Page No. Date

```
int main () {
    struct Node *head = NULL,
    * newNode, * temp;
    int n, value, pos;

printf ("How many nodes ?");
scanf ("%d", &n);

for (int i = 0; i < n; i++) {
    newNode = (struct Node*) malloc (sizeof (struct Node));
    printf ("Enter data for node %d: ", i+1);
    scanf ("%d", &newNode -> data);
    newNode -> next = NULL;

    if (head == NULL)
        head = newNode;
    else {
        temp = head;
        while (temp -> next != NULL)
            temp = temp -> next;
        temp -> next = newNode;
    }
}

printf ("\n Enter value to insert: ");
scanf ("%d", &value);
```

Teacher's Signature :

Expt. No. Date

Page No.

```
printf ("Insert after which position ?");
scanf ("%d", &pos);

newNode = (struct Node*) malloc (sizeof (struct Node));
newNode → data = value;

temp = head;
for (int i = 1; i < pos; i++) {
    temp = temp → next;
}

newNode → next = temp → next;
temp → next = newNode;

printf ("\n Updated linked list :");
temp = head;
while (temp != NULL) {
printf ("%d → ", temp → data);
    temp = temp → next;
}
printf ("NULL \n");

return 0;
}
```

Teacher's Signature :

Expt. No. _____ Date _____

Page No. _____

Experiment-12

Preprocessor and Derivatives in C

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

```
#include <stdio.h>

#define PI 3.14159
#define Max_Value 100
#define MESSAGE "Hello, preprocessor!\n"

int main() {
    printf("Value of PI: %.5f\n", PI);
    printf("Maxim Value: %d\n", Max_value);
    printf("%s\n", MESSAGE);

    return 0;
}
```

2. Write a program to define a function in directive.

```
#Include <stdio.h>
#define SQUARE(n) ((n) * (n))
```

Teacher's Signature: _____

Expt. No. _____ Date _____

Page No. _____

```
#define MAX(a,b) ((a)>(b)?(a):(b))

int main() {
  int num=5;
  int x = 10, y = 20;

  printf("Square of %d is %d\n", num, SQUAR(num));
  printf("Maximum of %d & %d is %d\n", x, y, MAX(x,y));

  return 0;
}
```

Expt. No. _____ Date _____

Page No. _____

Experiment - 13

Macros in C.

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

```
#include <stdio.h>

#define ADD (a,b) ((a) + (b))
#define SUB (a,b) ((a) - (b))
#define MUL (a,b) ((a) * (b))
#define DIV (a,b) ((b) != 0 ? ((a) / (b)) : 0)
#define MOD (a,b) ((a) % (b))

int main() {
    int x = 20, y = 6;

    printf("Addition : %d + %d = %d\n", x, y, ADD(x,y));
printf("Substration : %d - %d = %d\n", x, y, SUB(x,y));
printf("Multiplication : %d * %d = %d\n", x, y, MUL(x,y));
printf("Division : %d / %d = %d\n", x, y, DIV(x,y));
printf("Modulo : %d %% %d = %d\n", x, y, MOD(x,y));

    return 0;
}
```

Teacher's Signature : _____

Expt. No. _____ Date _____

Page No. _____

Experiment - 14

Static library in C

1. Write a program to create a static library for performing arithmetic functions.

```
#ifndef ARITHMETIC_H
#define ARITHMETIC_H

int add(int a, int b);
int sub(int a, int b);
int mul(int a, int b);
int divide(int a, int b);

#endif
```

```
#include "arithmetic.h"

int add(int a, int b) {
    return a+b;
}
int sub(int a, int b) {
    return a-b;
}
int mul(int a, int b) {
```

Teacher's Signature : _____

Expt. No. _____ Date _____

Page No. _____

```
    return a * b;
}

int divide (int a, int b) {
    if (b == 0)
        return 0;
    return a / b;
}

#include <stdio.h>
#include "arithmetic.h"

int main() {
    int a = 12, b = 4;

    printf("Addition: %d\n", add(a, b));
    printf("Subtraction: %d\n", sub(a, b));
    printf("Multiplication: %d\n", mul(a, b));
    printf("Division: %d\n", divide(a, b));

    return 0;
}
```

Teacher's Signature : _____

Expt. No. _____ Date _____

Page No. _____

(ii)- Write a program to use static library in other program.

```
#ifndef MATHLIB_H
#define MATHLIB_H

int add (int a, int b);
int sub (int a, int b);

#endif

#include "mathlib.h"

int add (int a, int b) {
    return a + b;
}
int sub (int a, int b) {
    return a - b;
}

#include <stdio.h>
#include "mathlib.h"

int main () {
    int a = 10, b = 5;
```

Teacher's Signature : _____

Expt. No. _____ Date _____

Page No. _____

```
printf("Add = %d\n", add(a,b));
printf("Sub = %d\n", sub(a,b));

return 0;
}
```

Page No.

Experiment-15

Shared library in C

1. Write a program to create a shared library for performing arithmetic functions.

```
#ifndef ARITH-H
#define ARITH-H

int add (int a, int b);
int sub (int a, int b);
int mul (int a, int b);
float divide (int a, int b);

#endif
```

```
#include "arith.h"

int add (int a, int b) {
    return a + b;
}
int sub (int a, int b) {
    return a - b;
}
int mul (int a, int b) {
    a * b;
```

Expt. No. _____ Date _____

Page No. _____

```
}
float divide (int a, int b){
    if (b == 0) return 0;
    return (float) a/b;
}

#include <stdio.h>
#include "arith.h"

int main() {
    int a = 20, b = 4;

    printf("Add = %d\n", add(a,b));
    printf("Sub = %d\n", sub(a,b));
    printf("Mul = %d\n", mul(a,b));
    printf("Div = %.2f\n", divide(a,b));

    return 0;
}
```

2. Write a program to use shared library in other program.

```
#ifndef ARITH_H
#define ARITH_H
```

Teacher's Signature : _____

Date

Page No.

```
int add (int a, int b);
int sub (int a, int b);
int mul (int a, int b);
float divide (int a, int b);

#endif

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

int main () {
    int a = 15, b = 3;

printf("Addition : %d\n", add (a,b));
printf ("Subtraction : %d\n", sub(a,b));
printf ("Multiplication : %d\n", mul(a,b));
printf ("Division : %d\n", divide (a,b));

return 0;
}
```

Teacher's Signature :