Programs:

1) **Write a C++ program to overload function called ‘square’ to calculate the square of an int variable and the square of a double variable.**

```
#include <iostream>

using namespace std;

int square(int x){

    cout << "Square is " << x*x << endl;

    return 1;

}

int square(double x){

    cout << "Square is " << x*x << endl;

    return 1;

}

int main(){

    int x;

    double y;

    cout << "Enter the Integer Value: ";

    cin>>x;

    cout << "\nEnter the Double Value: ";

    cin>>y;

    square(x);

    square(y);

    return 0;

}
```

Output:

```
┌──(root💀ROBEENKS)-[~/…/Class/PCC-SEM-3/OOPS/EXPT3]
└─# ./a.out
Enter the Integer Value: 30

Enter the Double Value: 30.121
Square is 900
Square is 907.275
```

2) Write a C++ program to overload function called ‘area’ to find the area of square rectangle and circle.

```
#include <iostream>
using namespace std;
int area(int x){
    cout << "Area is " << x*x << endl;
    return 1;
}
int area(int x, int y){
    cout << "Area is " << x*y << endl;
    return 1;
}
int area(double r){
    cout << "Area is " << 3.14*r*r << endl;
}
int main(){
    int square_x;
    int rect_x,rect_y;
    double circle_r;
    cout << "Enter the lenght of Sqare: ";
    cin>>square_x;
    cout << "\nEnter the lenght and bredth of rectangle: ";
    cin>>rect_x;
    cin>>rect_y;
    cout << "\nEnter the Radius of the cirlce: ";
    cin>>circle_r;
    area(square_x);
    area(rect_x,rect_y);
    area(circle_r);

    return 0;
}
```

Output:

```
┌──(root💀ROBEENKS)-[~/…/Class/PCC-SEM-3/OOPS/EXPT3]
└─# ./a.out
Enter the length of Sqare: 30

Enter the length and bredth of rectangle: 20 10

Enter the Radius of the cirlce: 50
Area is 900
Area is 200
Area is 7850
```

3) Write a C++ program to overload function called ‘swap’ to swap two variables of integer, float and char types.

```
#include <iostream>
using namespace std;
int swap(int x,int y){
    int temp;
    temp=x;
    x=y;
   y=temp;
    cout << "Swapped value is " << x <<" " << y << endl;
    return 1;
}
float swap(float x, float y){
    float temp;
    temp=x;
    x=y;
    y=temp;
    cout << "Swapped value is " << x <<" " << y << endl;
    return 1;
}
char swap(char x, char y){
    char temp;
    temp=x;
    x=y;
    y=temp;
    cout << "Swapped value is " << x <<" " << y << endl;
    return 1;
```

```
}
int main(){
    int integer_x,integer_y;
    float float_x,float_y;
    char char_x,char_y;
    cout << "Enter the X and Y value of Integer: ";
    cin>>integer_x;
    cin>>integer_y;
    cout << "Enter the X and Y value of Float: ";
    cin>>float_x;
    cin>>float_y;
    cout << "Enter the X and Y value of Char: ";
    cin>>char_x;
    cin>>char_y;
    swap(integer_x,integer_y);
    swap(float_x,float_y);
    swap(char_x,char_y);
    return 0;
}
```

Output:

```
┌──(root💀ROBEENKS)-[~/…/Class/PCC-SEM-3/OOPS/EXPT3]
└─# ./a.out
Enter the X and Y value of Integer: 10 40
Enter the X and Y value of Float: 13.2 50.2
Enter the X and Y value of Char: A Z
Swapped value is 40 10
Swapped value is 50.2 13.2
Swapped value is Z A
```

4) Write a C++ program to overload function called ‘sum’ that adds the elements of two multi-dimensional arrays for 2 integer arrays and 2 double arrays.

```
#include <iostream>
using namespace std;
int sum(int a[2][2], int b[2][2])
```

```
{
    int c[2][2];
    for (int i = 0; i < 2; i++)
    {
        for (int j = 0; j < 2; j++)
        {
            c[i][j] = a[i][j] + b[i][j];
        }
    }
    for (int i = 0; i < 2; i++)
    {
        for (int j = 0; j < 2; j++)
        {
            cout << c[i][j] << "\t";
        }
        cout << " " << endl;
    }
    return 1;
}
float sum(float a[2][2], float b[2][2])
{
    float c[2][2];
    for (int i = 0; i < 2; i++)
    {
        for (int j = 0; j < 2; j++)
        {
            c[i][j] = a[i][j] + b[i][j];
        }
    }
    cout << "\n";
    for (int i = 0; i < 2; i++){
        for (int j = 0; j < 2; j++) {
```

```
            cout << c[i][j] << "\t";
        }
        cout << " " << endl;
    }
}
int main()
{
    int int_a[2][2] = {{1, 2}, {3, 6}};
    int int_b[2][2] = {{2, 10}, {10, 50}};
    float float_a[2][2] = {{10.1, 2.2}, {3.4, 6.2}};
    float float_b[2][2] = {{2.0, 5.12}, {10.3, 5.4}};
    sum(int_a, int_b);
    sum(float_a, float_b);
    return 0;
}
```

Output:

```
┌──(root💀ROBEENKS)-[~/…/Class/PCC-SEM-3/OOPS/EXPT3]
└─# ./a.out
3       12
13      56

12.1    7.32
13.7    11.6
```

5) Write a recursive function power (base, exponent) that when invoked returns base^exponent. E.g. power (3,4) = 3*3*3*3. Exponent has to be greater or equal to 1.

```
#include <iostream>
using namespace std;
int power(int base, int exponent){
    int result;
    if (exponent == 0){
```

```
        return 1;
    }
    return base * power(base, exponent - 1);
}
int main(){
    int base, exponent;
    cout << "Enter the Base: ";
    cin >> base;
    cout << "Enter the Exponent: ";
    cin >> exponent;
    cout << base << " ^ " << exponent << " = " << power(base, exponent);
    return 0;
}
```

Output:

```
┌──(root💀ROBEENKS)-[~/…/Class/PCC-SEM-3/OOPS/EXPT3]
└─# ./a.out
Enter the Base: 20
Enter the Exponent: 5
20 ^ 5 = 3200000
```

6) Write a C++ Program to implement Linear Search using recursion.

```
#include <iostream>
using namespace std;
int LinearSearch(int a[20], int val, int count)
{
    if (val == a[count]) {
        cout << val << " Value found at postion " << count;
        return 1;
    }
    if (count == sizeof(a)){
        cout << val << " Not found";
        return 0;
    }
```

```
        count++;
        LinearSearch(a, val, count);
        return 1;
    }
    int main()
    {
        int num;
        int values[20];
        int val;
        int count = 0;
        cout << "Enter the Number of Elements: ";
        cin >> num;
        cout << "Enter the elements in the array: ";
        for (int i = 0; i < num; i++){
            cin >> values[i];
        }
        cout << "Enter the value to be searched: ";
        cin >> val;
        LinearSearch(values, val, count);
        return 0;
    }
```

Output:

```
┌──(root💀ROBEENKS)-[~/…/Class/PCC-SEM-3/OOPS/EXPT3]
└─# ./a.out
Enter the Number of Elements: 4
Enter the elements in the array: 3 5 7 2
Enter the value to be searched: 5
5 Value found at postion 1
```