**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.**

**Input:**

#include <iostream>

#include<conio.h>

using namespace std;

int square(int x) {

return x\*x;

}

double square(double x) {

return x\*x;

}

int main()

{

int x; double y;

cout << "Enter the Int Number: ";

cin >> x;

cout << "Square of integer number is " << square(x) << endl;

cout << "Enter the double Number: ";

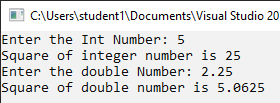
cin >> y;

cout << "Square of double number is " << square(y) << endl;

\_getch();

}

**Output:**

****

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

**Input:**

#include <iostream>

#include<conio.h>

using namespace std;

const double pi = 3.14;

int area(int side) {

return side \* side;

}

double area(double radius) {

return (pi\*radius\*radius);

}

int area(int length, int breadth) {

return length \* breadth;

}

int main()

{

int side, length, breadth;

double radius;

cout << "Enter the side of the Square: ";

cin >> side;

cout << "The area of the Square is " << area(side) << endl;

cout << "Enter the length & breadth of the rectangle: ";

cin >> length >> breadth;

cout << "The area of the rectangle is " << area(length, breadth) << endl;

cout << "Enter the radius of the circle: ";

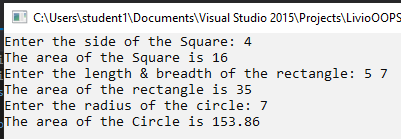
cin >> radius;

cout << "The area of the Circle is " << area(radius) << endl;

\_getch();

}

**Output:**

****

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

**Input:**

#include <iostream>

#include<conio.h>

using namespace std;

void swap(int a, int b) {

int temp = a;

a = b;

b = temp;

cout << "The value of int 'a' is " << a << " and the value of int 'b' is " << b << endl;

return;

}

void swap(float a, float b) {

float temp = a;

a = b;

b = temp;

cout << "The value of float 'c' is " << a << " and the value of float 'd' is " << b << endl;

return;

}

void swap(char a, char b) {

char temp = a;

a = b;

b = temp;

cout << "The value of char 'e' is " << a << " and the value of char 'f' is " << b << endl;

return;

}

int main()

{

int a, b;

float c, d;

char e, f;

cout << "Enter the int value of a : ";

cin >> a;

cout << "Enter the int value of b: ";

cin >> b;

swap(a, b);

cout << "Enter the float value of a: ";

cin >> c;

cout << "Enter the float value of b: ";

cin >> d;

swap(c, d);

cout << "Enter the char value of a: ";

cin >> e;

cout << "Enter the char value of b: ";

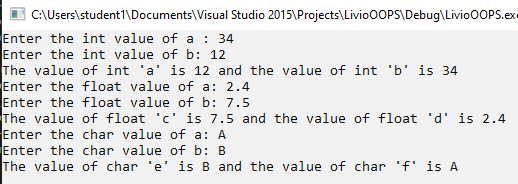
cin >> f;

swap(e, f);

\_getch();

}

**Output:**

****

1. **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.**

**Input:**

#include <iostream>

#include<conio.h>

using namespace std;

void sum(int x[2][2], int y[2][2]) {

int sumArr[2][2] = { 0 };

cout << "The sum of element of two integer multi-dimensional arrays are:" << endl;

for (int i = 0; i<2; i++) {

for (int j = 0; j<2; j++)

sumArr[i][j] = x[i][j] + y[i][j];

}

for (int i = 0; i<2; i++) {

for (int j = 0; j<2; j++)

cout << sumArr[i][j] << "\t";

cout << "\n";

}

return;

}

void sum(float x[2][2], float y[2][2]) {

float sumArr[2][2] = { 0 };

cout << "The sum of element of two float multi-dimensional arrays are:" << endl;

for (int i = 0; i<2; i++) {

for (int j = 0; j<2; j++)

sumArr[i][j] = x[i][j] + y[i][j];

}

for (int i = 0; i<2; i++) {

for (int j = 0; j<2; j++)

cout << sumArr[i][j] << "\t";

cout << "\n";

}

return;

}

int main()

{

int iMDArr1[2][2] = { 0 };

int iMDArr2[2][2] = { 0 };

float fMDArr1[2][2] = { 0 };

float fMDArr2[2][2] = { 0 };

int i = 0, j = 0;

cout << "Enter the elements for integer array1 at the index:" << endl;

for (i = 0; i<2; i++) {

for (j = 0; j<2; j++) {

cout << "[" << i << "]" << "[" << j << "]" << ":\t";

cin >> iMDArr1[i][j];

}

}

cout << "Enter the elements for integer array2 at the index:" << endl;

for (i = 0; i<2; i++) {

for (j = 0; j<2; j++) {

cout << "[" << i << "]" << "[" << j << "]" << ":\t";

cin >> iMDArr2[i][j];

}

}

sum(iMDArr1, iMDArr2);

cout << "Enter the elements for integer array1 at the index:" << endl;

for (i = 0; i<2; i++) {

for (j = 0; j<2; j++) {

cout << "[" << i << "]" << "[" << j << "]" << ":\t";

cin >> fMDArr1[i][j];

}

}

cout << "Enter the value of elements for integer array1 at the index:" << endl;

for (i = 0; i<2; i++) {

for (j = 0; j<2; j++) {

cout << "[" << i << "]" << "[" << j << "]" << ":\t";

cin >> fMDArr2[i][j];

}

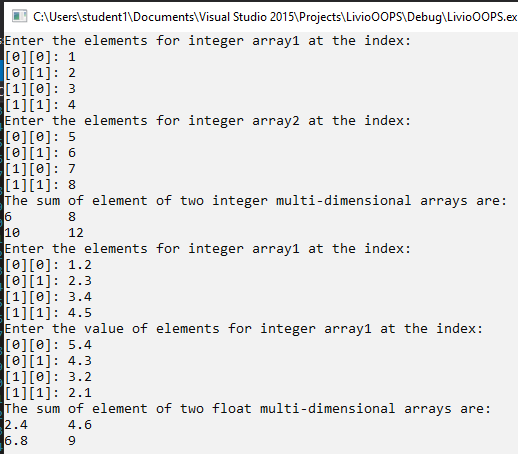
}

sum(fMDArr1, fMDArr1);

\_getch();

}

**Output:**

****

1. **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.**

**Input:**

#include<iostream>

#include<conio.h>

using namespace std;

int power(int base, int exponent) {

if (exponent == 1)

return base;

return base\*power(base, exponent - 1);

}

int main() {

int base = 0, exponent = 0;

cout << "Enter the base value:\t";

cin >> base;

cout << "Enter the exponent:\t";

cin >> exponent;

if (exponent < 1) {

cout << "Please enter exponent greater or equal to 1";

}

else {

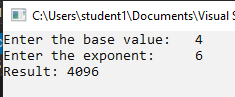
cout << "Result: " << power(base, exponent);

}

\_getch();

}

**Output:**

****

1. **Write a C++ Program to implement Linear Search using recursion.**

**Input:**

#include<iostream>

#include<conio.h>

using namespace std;

int linearSearch(int arr[], int searchElement, int index, int size) {

if (index == size) {

cout << "Element not found" << endl;

return 0;

}

if (searchElement == arr[index]) {

cout << "Element found at index: " << index + 1;

return index;

}

return linearSearch(arr, searchElement, index + 1, size);

}

int main() {

int size, arr[100], i = 0, searchElement ;

cout << "Enter size of Array: ";

cin >> size;

cout << "Enter elements of the array: "<<endl;

while (i < size)

{

cin >> arr[i];

i++;

}

cout << "Enter the element to be searched: ";

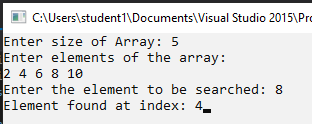
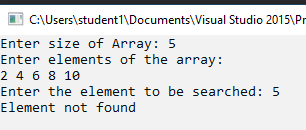
cin >> searchElement;

linearSearch(arr, searchElement, 0, size);

\_getch();

}

**Output:**

** **