# EXPERIMENT NUMBER –Practical 6.1

**TOPIC**

**WAP to calculate and display cube of an integer and float variable using function overloading.**

**AIM OF EXPERIMENT-**

**To learn the concept of polymorphism.**

**FLOWCHART/ ALGORITHM-**

Start.

Step 1→ Creating a header file for input output stream and define the context.

# Step 2 → After that used using namespace std;

Step 3 → Declare data members int and float to get output in the format of integer and real number

Step 4→ Create the int main() member function for passing or declaring the values of int a; and float b;

Step 5→ Then using pre-defined object cout printing the cube of integer and float which I have created in main function.

Step 6→ Declaring the member function int cube () and float cube (). Passing the 10 and 5.5 for int and float respectively

Step 7→ Within the int cube we passing the 1 argument to do multiplication of one number to get the cube of that number.

# Step 8→ End the program by returning the cube of the integer and float which I have created and declared. On the output screen its showing whole cube of integer and another one output showing cube till 3 rd decimal value.

# Step 9→for 10 cube showing 1000 and for float number 5.5 cube of that number showing 166.375

# Stop.

**PROGRAM CODE**

#include <iostream>

using namespace std;

int cube(int);

float cube(float);

int main()

{

int a = 10;

float b = 5.5;

cout<<"==| smita shinde uid - 20BCS4643 |=="<<endl;

cout<< "Cube of integer number " << a << " is " << cube(a) <<endl;

cout<< "Cube of float number " << b << " is " << cube(b) <<endl;

return 0;

}

int cube(int a)

{

return a\*a\*a;

}

float cube(float b)

{

return b\*b\*b;

}

**ERRORS ENCOUNTERED DURING PROGRAM’S EXECUTION**

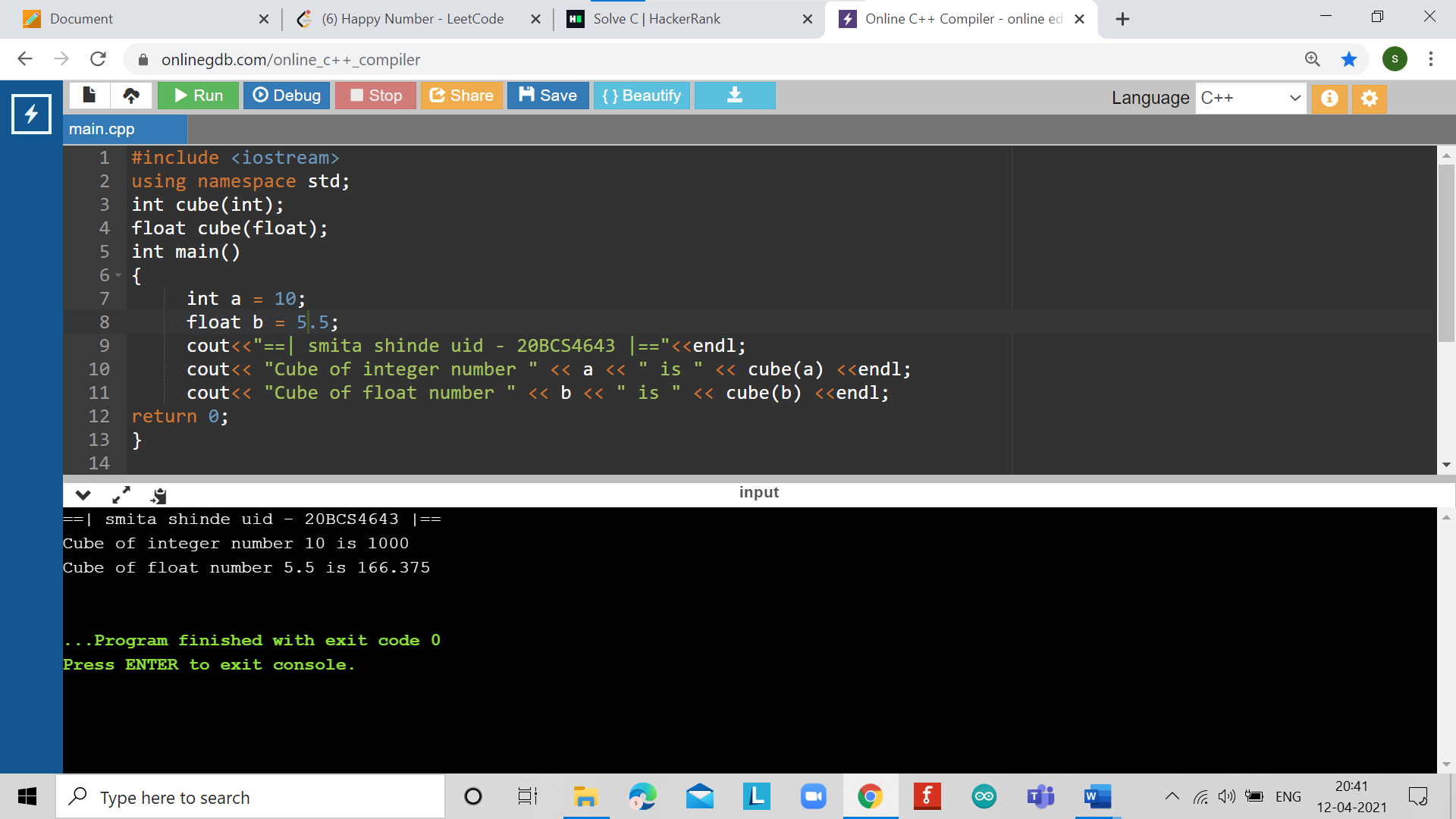
**(Kindly jot down the compile time errors encountered)**

No error is found when compiling the program

**PROGRAMS’ EXPLANATION (in brief)**

1. We start our program with preprocessor (#) and header file (< iostream >) we have many types of header files but in this program, we used iostream. #include< iostream >, Int main () is a function which work as a container of statements. All the statements are enclosed within the pair of braces { }. “using namespace std” means we use the namespace named std. “std” is an abbreviation for standard. So that means we use all the things with in “std” namespace.
2. In this program, we declare two variables as cube of int datatype and cube of float datatype. Then we assign values to them as 6 and 2.5 respectively. And then using overloading the cubes of both the numbers is displayed separately. The output is shown below.

**OUTPUT**

****

# EXPERIMENT NUMBER –Practical 6.2

**TOPIC**

**Program to demonstrate the unary operator overloading for operator ++. Make a class test. Create a default constructor to initialize the variable. 1) Overload operator ++ (Pre) with definition to pre-decrement the value of a variable 2) Overload operator ++ (post) with definition to post-decrement the value of variable.**

**AIM OF EXPERIMENT-**

**To learn the concept of polymorphism.**

**FLOWCHART/ ALGORITHM-**

Start.

Step 1→ Creating a header file for input output stream and define the context.

# Step 2 → After that I create a class test and declare data member int num;

# Within private context.

Step 3 → Create the default constructor to initlize the variable. As name followed by the name of class test.

Step 4→ Create parameterized constructor to return object after incrementing for passing the arguments.

Step 5→ Declare and define the function void display() to display .

Step 6→ Then the overloading the function operator ++ for increment this object. ( overloaded prefix ++ operator)

Step 7→ Declare and define the function return Test(num); to return object with increment value.

Step 8→ Then the overloading the function operator ++ for save the original value. (overloaded post ++ operator)

Step 9→ Declare and define the function int main() to call all the functions.

Step 10→ Create the derived class object obj1 , obj2, obj3 For calling member functions.

Step 11→ Call the function display() using dot(.) operator .

# Step 12→ End the program by returning increment display obj1 ,increment obj2 display obj3 as passed arguments when defined objects as 11 12 0 13 and 11 .

# Stop.

**PROGRAM CODE**

#include <iostream>

using namespace std;

class Test {

private:

int num;

public:

// required constructors

// default constructor to initlize the variable

Test() {

num = 0;

cout<<"==|smita shinde uid 20BCS4643 |=="<<endl;

}

// parameterized constructor to return object after incrementing

Test(int n) {

num = n;

}

// method to display time

void display() {

cout<< "Number: " <<num<<endl;

}

// overloaded prefix ++ operator

Test operator++ () {

// increment this object

++num;

// return object with increment value

return Test(num);

}

// overloaded postfix ++ operator

Test operator++( int ) {

// save the orignal value

Test t(num);

// increment current object

++num;

// return old original value

return t;

}

};

int main() {

Test obj1(10), obj2(11), obj3;

++obj1; //increment obj1

obj1.display(); // display obj1

obj2++; // increment obj2

obj2.display(); // display obj2

obj3.display(); // display obj3

obj3 = obj2++; // increment obj2 again and assign pre-incremented value to obj3

obj2.display(); // display obj2

obj3.display(); // display obj3

return 0;

}

**PROGRAMS’ EXPLANATION (in brief)**

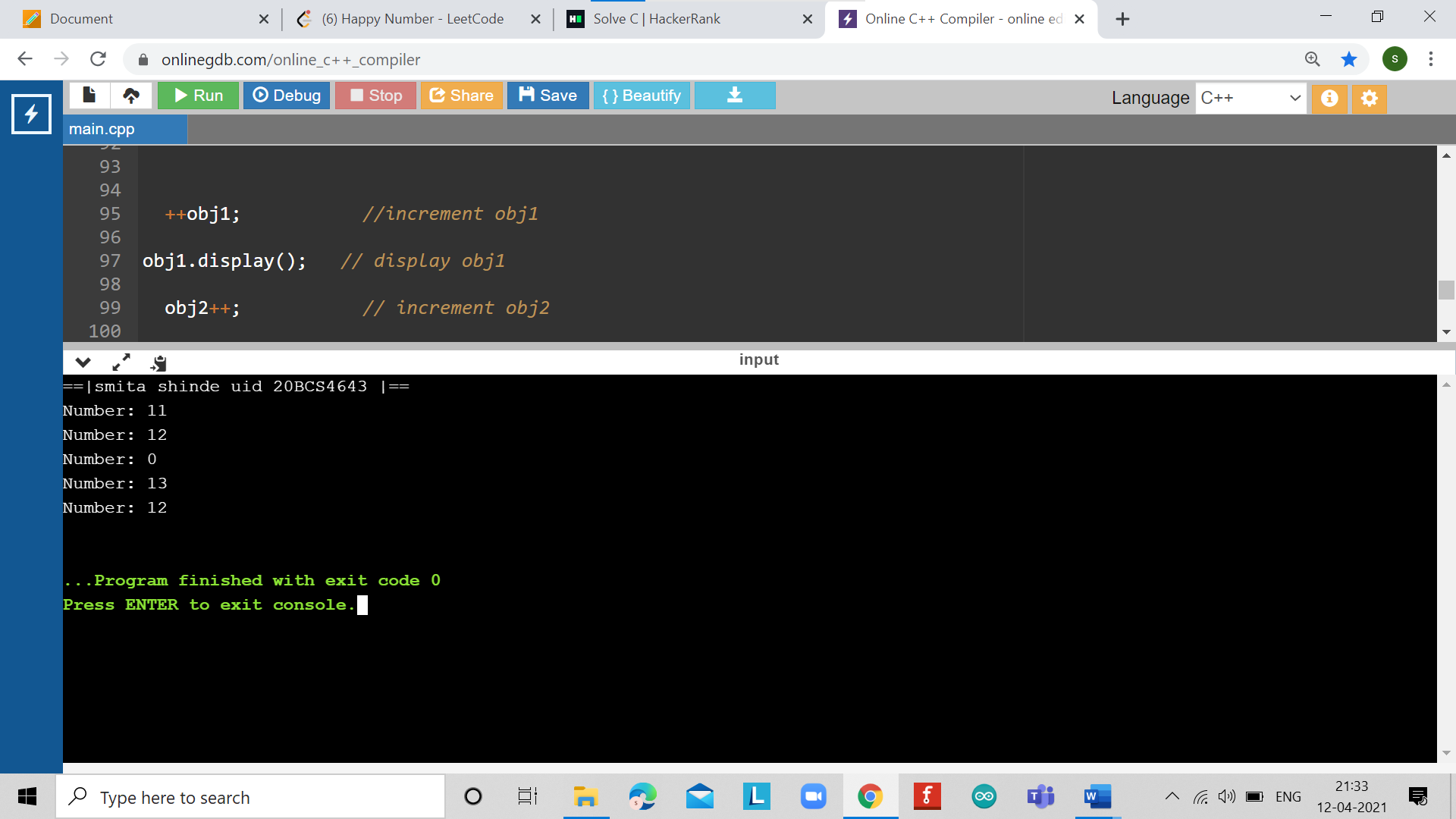
1. We start our program with preprocessor (#) and header file (< iostream >) we have many types of header files but in this program, we used iostream. #include< iostream >, Int main () is a function which work as a container of statements. All the statements are enclosed within the pair of braces { }. “using namespace std” means we use the namespace named std. “std” is an abbreviation for standard. So that means we use all the things with in “std” namespace.
2. After incrementing obj1 it is printed by calling display function.
3. After incrementing obj2 it is printed by calling display function.

Initial value of obj3.

After incrementing obj2 it is printed by calling display function.

assignment of Pre-increment object obj2 it is printed by calling display function.

**OUTPUT**

****

# EXPERIMENT NUMBER –Practical 6.3

**TOPIC**

WAP for creating a matrix class which can handle integer matrices of different dimensions. Overload the operator (+) for addition and (==) comparison of matrices.

**AIM OF EXPERIMENT-**

**To learn the concept of polymorphism.**

**FLOWCHART/ ALGORITHM-**

Start.

Step 1→ Creating a header file for input output stream and define the context.

Step 2 → first define the preprocessor as maxrows to 50 and maxcols to 50.

# After that used using namespace std;

Step 3 → Declare and define the class name followed by class matrix within public access specifier.

Step 4→ Defining data members rows and columns.

Step 5→ Declare and define the function void display() and defining the predefined objects cout for getting output.

Step 6→ Then declare an array to input the values to three matrices

Step 7→ Then using function overloading of addition operator(+) and compiler performs the addition of matrix 1 and matrix 3.

Step 8→ Then using function overloading of comparison operator (==)

Step 9→ Each element of matrix 1 and matrix 2 are matched then the displays as they are equal.

Step 10→ it compares matrix 1 and matrix 3 display as they are not equal.

# Stop.

* **PROGRAM CODE**

#include <iostream>

#define MAXROWS 50

#define MAXCOLS 50

using namespace std;

class Matrix {

public:

int arr[MAXROWS][MAXCOLS];

int rows, cols;

Matrix() {

rows = cols = 2;

}

Matrix(int r, int c, int mat[MAXROWS][MAXCOLS]) {

rows = r;

cols = c;

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

for (int j = 0; j < cols; j++) {

arr[i][j] = mat[i][j];

}

}

}

void display() {

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

cout<< " [ ";

for (int j = 0; j < cols; j++) {

cout<<arr[i][j] << ", ";

}

cout<< "]" <<endl;

}

cout<<endl;

}

Matrix operator+(Matrix x) {

if (x.rows != rows || x.cols != cols || (rows == 0 && cols == 0)) {

return Matrix();

}

int mat[MAXROWS][MAXCOLS];

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

for (int j = 0; j < cols; j++) {

mat[i][j] = arr[i][j] + x.arr[i][j];

}

}

return Matrix(rows, cols, mat);

}

int operator==(Matrix x) {

if (x.rows != rows || x.cols != cols) {

return 0;

}

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

for (int j = 0; j < cols; j++) {

if (arr[i][j] != x.arr[i][j]) {

return 0;

}

}

}

return 1;

}

};

int main()

{

int arr1[MAXROWS][MAXCOLS], arr2[MAXROWS][MAXCOLS];

arr1[0][0] = 1;

arr1[0][1] = 2;

arr1[1][0] = 3;

arr1[1][1] = 4;

arr2[0][0] = 4;

arr2[0][1] = 3;

arr2[1][0] = 2;

arr2[1][1] = 1;

Matrix mat1(2, 2, arr1);

Matrix mat2(2, 2, arr1);

Matrix mat3(2, 2, arr2);

Matrix mat4;

cout<< "Elements of Matrix 1:" <<endl;

mat1.display();

cout<< "Elements of Matrix 2:" <<endl;

mat2.display();

cout<< "Elements of Matrix 3:" <<endl;

mat3.display();

mat4 = mat1 + mat3;

cout<< "Elements of Matrix after addition of Matrix 1 and Matrix 3:" <<endl;

mat4.display();

if (mat1 == mat2) {

cout<< "Matrix 1 is equals to Matrix 2" <<endl;

}

else {

cout<< "Matrix 1 is not equals to Matrix 2" <<endl;

}

if (mat1 == mat3) {

cout<< "Matrix 1 is equals to Matrix 3" <<endl;

}

else {

cout<< "Matrix 1 is not equals to Matrix 3" <<endl;

}

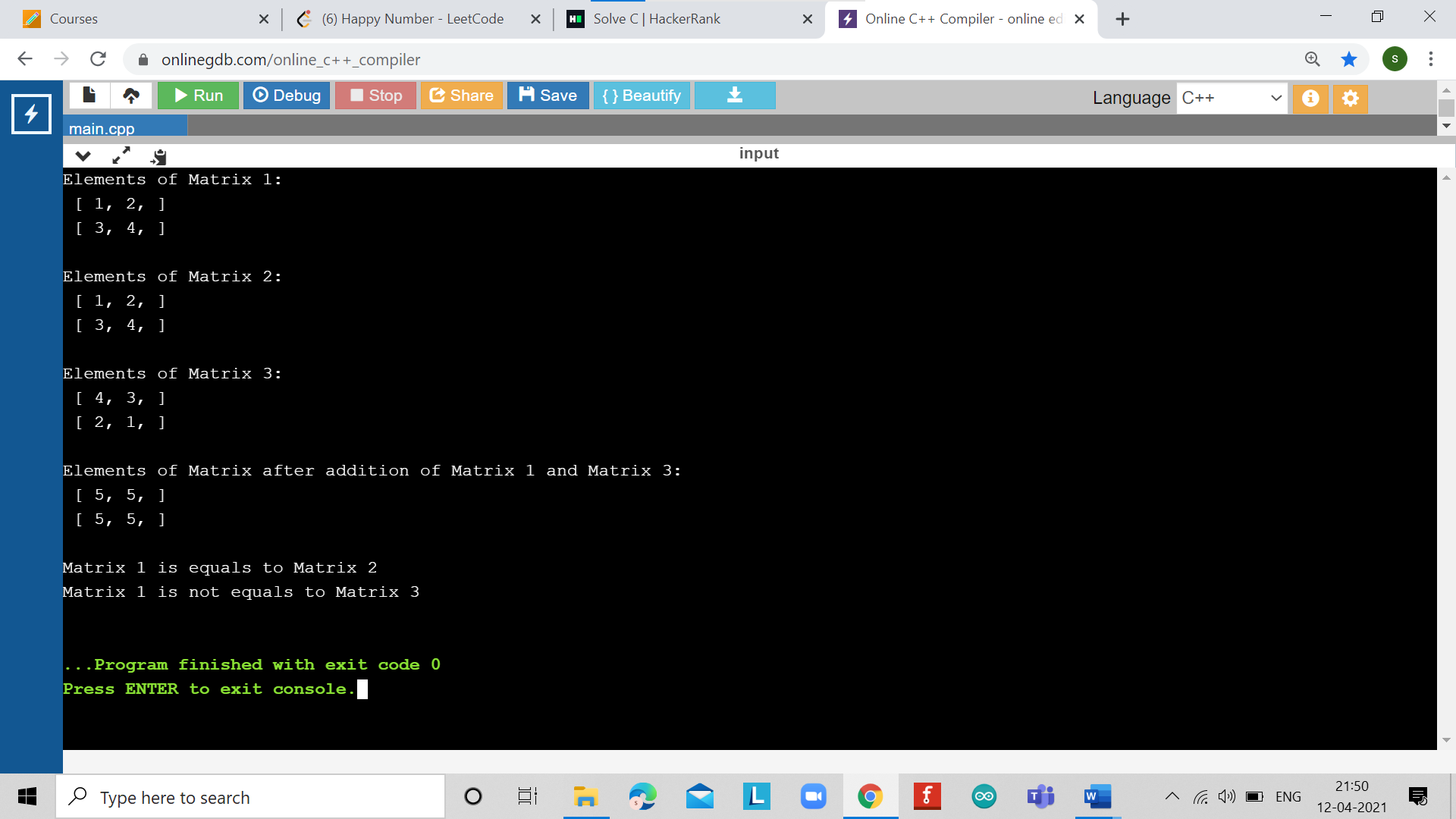
return 0;

}

**PROGRAMS’ EXPLANATION (in brief)**

1. We start our program with preprocessor (#) and header file (< iostream >) we have many types of header files but in this program, we used iostream. #include< iostream >, Int main () is a function which work as a container of statements. All the statements are enclosed within the pair of braces { }. “using namespace std” means we use the namespace named std. “std” is an abbreviation for standard. So that means we use all the things with in “std” namespace.
2. In this program, we first define the preprocessor as maxrows to 50 and maxcolums to 50. Then we declare an array to input the values to three matrices. Then using function overloading of addition operator(+) and compiler performs the addition of matrix 1 and matrix 3. Then using function overloading of comparison operator (==) , each element of matrix 1 and matrix 2 are matched then the displays as they are equal and then it compares matrix 1 and matrix 3 then it displays as they are not equal. These are displayed on the screen . The output is shown below.

**OUTPUT-**

****

# EXPERIMENT NUMBER –Practical 6.4

**TOPIC**

WAP to create a class Pairs. Objects of type Pairs can be used in any situation where ordered pairs are needed. Our Task is to overload operator >> and << so that objects of class Pairs are to be input and output in the form (5,3) (5,-6) (-5,6) or (-5,-3).There is no need to implement any constructor/method .

**AIM OF EXPERIMENT-**

**To learn the concept of polymorphism.**

**FLOWCHART/ ALGORITHM-**

Start.

Step 1→ Creating a header file for input output stream and define the context.

Step 2 → Declare and define the class name followed by class Pairs within public access specifier.

Step 4→ Defining data members char within private context .

Step 5→ Declare an array of characters the predefined objects cout for getting output.

Step 6→ The user is asked to enter the pair object. For the input

Step 7→ If the pair object entered is in tShe form of (a,b) then the entered pair is displayed on screen.

Step 8→ otherwise If it is in form of a,b then the out is Invalid pair value found**.**

# Stop.

**PROGRAM CODE**

#include <iostream>

#include <cstring>

using namespace std;

class Pairs {

private:

char numpair[20];

public:

friend ostream&operator<<(ostream&output, const Pairs &p) {

output<<p.numpair;

return output;

}

friend istream&operator>>(istream&input, Pairs &p) {

char pair[20];

input>> pair;

int len = strlen(pair);

if (len< 5 || pair[0] != '(' || pair[len - 1] != ')' || !strstr(pair, ",")) {

cout<< "Invalid pair value found!" <<endl;

strcpy(p.numpair, "");

}

else {

strcpy(p.numpair, pair);

}

return input;

}

};

int main() {

Pairs p;

cout<< "Enter the value of pair object: ";

cin>> p;

cout<< "Entered pair value is: " << p <<endl;

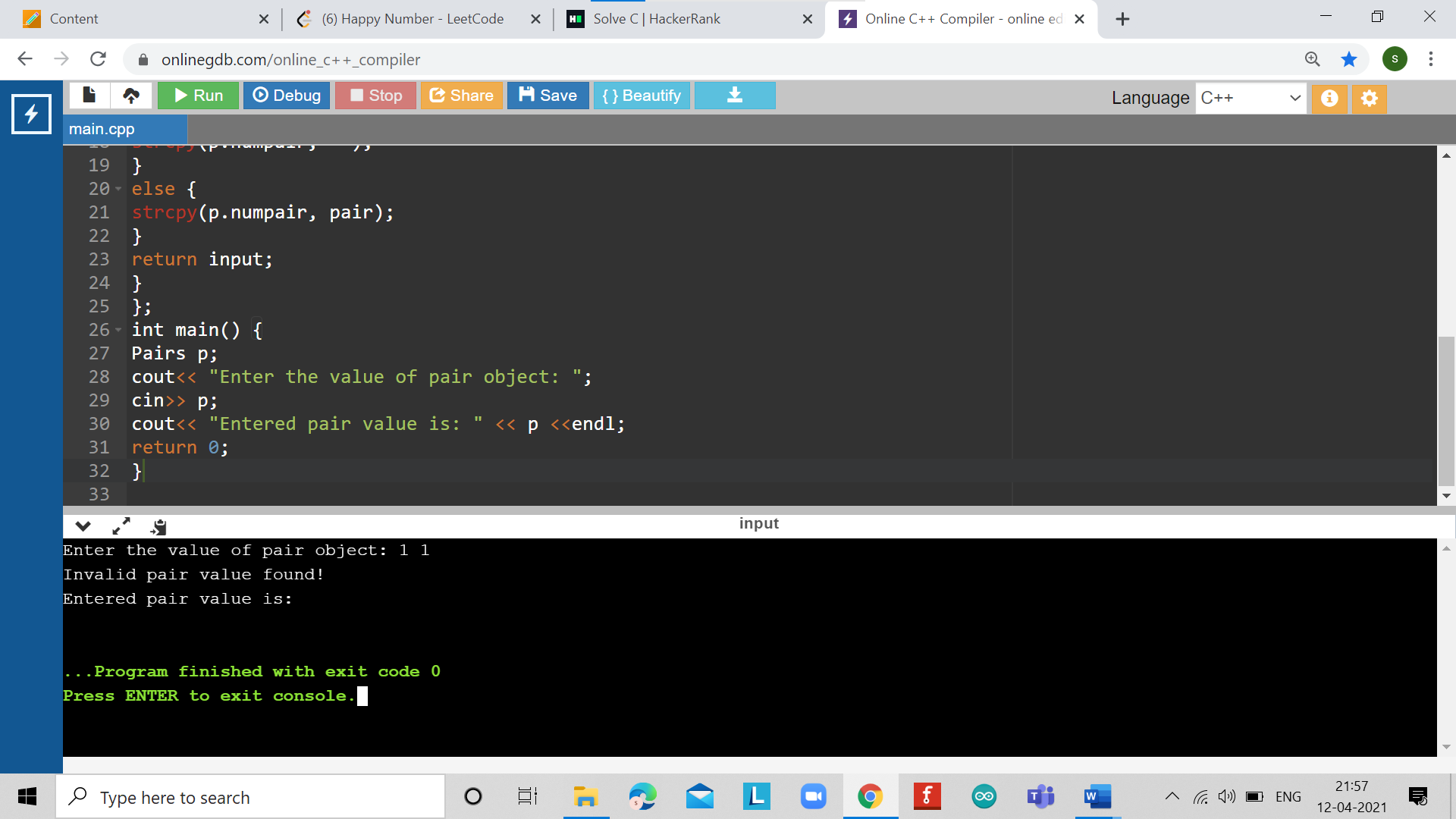
return 0;

}

**PROGRAMS’ EXPLANATION (in brief)**

1. We start our program with preprocessor (#) and header file (< iostream >) we have many types of header files but in this program, we used iostream. #include< iostream >, Int main () is a function which work as a container of statements. All the statements are enclosed within the pair of braces { }. “using namespace std” means we use the namespace named std. “std” is an abbreviation for standard. So that means we use all the things with in “std” namespace.
2. In this program first create a class named Pairs in which declare an array of characters. In this the user is asked to enter the pair object. If the pair object entered is in the form of (a,b) then the entered pair is displayed on screen otherwise if it is in form of a,b then the out is Invalid pair value found. The output is shown below.

**OUTPUT**

****

**LEARNING OUTCOMES**

|  |
| --- |
| * Identify situations where computational methods would be useful. |
| * Approach the programming tasks using techniques learnt and write pseudo-code. |
| * Choose the right data representation formats based on the requirements of the problem. |
| * Use the comparisons and limitations of the various programming constructs and choose the right one for the task. |

**EVALUATION COLUMN (To be filled by concerned faculty only)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No.** | **Parameters** | **Maximum**  **Marks** | **Marks**  **Obtained** |
| 1. | Worksheet Completion including writing learning objective/ Outcome | 10 |  |
| 2. | Post Lab Quiz Result | 5 |  |
| 3. | Student engagement in Simulation/ Performance/ Pre Lab Questions | 5 |  |
| 4. | Total Marks | 20 |  |