**Object-Oriented Programming Language**

**04/27/2023**

**Due:05/01/2023 11:59p.m.**

**Lab Assignment 9**

1. We can compile this code without error.

#include <iostream>

using namespace std;

class Student

{

int ID;

string sName;

// create constructor

//public:

// Student(const string& \_sName):sName(\_sName){}

};

int main(){

Student s1;

Student s2(s1);

}

But after we add a constructor, the code can’t compile. Please use “= default “ make this code working again. No output is needed.

1. Define a class “NonCopyable”, this class has only public part, no private part. Please block the copy constructor and copy assignment operator to make this class NonCopyable. This work no need input and output.
2. Define a class “Distance” with two private variables “feet” and ” inches”. Following is the main function you can’t change.

int main() {

Distance D1(11, 10), D2(5, 11), D3;

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

cin >> D3;

cout << "First Distance : " << D1 << endl;

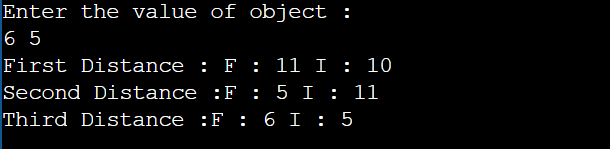
cout << "Second Distance :" << D2 << endl;

cout << "Third Distance :" << D3 << endl;

return 0;

}

Your output should look like this.



4. Implement a class “VecFour” as a vector of four

doubles. Implement suitable constructors and operators so the class can support the following code:

int main (){

VecFour a = VecFour(1.0,1.0,2.0,2.0) ;

cout << "The vector \'a\' is: " << a << endl ;

VecFour b ;

cout << "Please input a vector: " ;

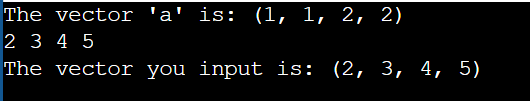
cin >> b ; // 0.0,1.0,2.0,5.0

cout << "The vector you input is: " << b << endl ;

return 0 ;

}

Your output should look like this.



1. Follow the previous question. Implement suitable constructors and operators so the class can support the following code:

Int main(){

VecFour a = VecFour(1.0,1.0,2.0,2.0) ;

VecFour c = 2.5\*a ;

cout << "The vector \'c\' is : " << c << endl ;

c \*= a;

cout << "The vector \'c\' changes to : " << c << endl ;

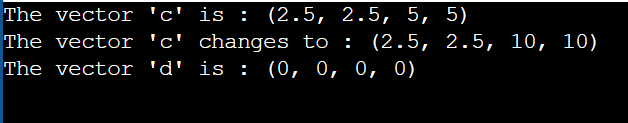
VecFour d ; // 0.0,0.0,0.0,0.0

cout << "The vector \'d\' is : " << d << endl ;

return 0 ;

}

Your output should look like this.



1. defines a class called “Complex”, which represents a complex number with two private member variables real and imag representing the real and imaginary parts. an overloaded + operator for adding two complex numbers together. Additionally, two friend functions operator>> and operator<< are defined to input and output complex objects.

This code you should not change.

int main() {

Complex c1, c2;

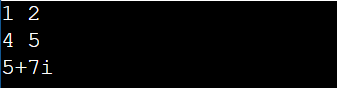
cin >> c1 >> c2;

cout << c1 + c2 << endl;

return 0;

}

Your input and output should look like this.



1. follow the previous question, add a operator>= to test which complex is greater.

This code you should not change.

int main() {

Complex c1, c2;

cin >> c1 >> c2;

if(c1 >= c2)

cout << c1 << " >= " << c2 << endl;

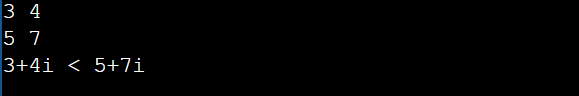
else

cout << c1 << " < " << c2 << endl;

return 0;

}

Your input and output should look like this.



1. Write a class “Safearay” with an integer array of size 10. Implement the constructor to initialize the array elements with integers from 0 to 9. Overload the subscript operator to provide index bounds checking. If an index is out of bounds, display an "out of range" error message and return the first element of the array.

This code you should not change.

int main(){

safearay A;

cout << "A[2] = " << A[2] <<endl;

cout << "A[5] = " << A[5]<<endl;

cout << "A[12] = " << A[12]<<endl;

return 0;

}

The output should look like this

