Object Oriented Programming

Lab Report

Lab04

Group Members Name & Reg #:	**Muhammad Haris Irfan** **(FA18-BCE-090)**
Class	Object Oriented Programming CSC241 (**BCE-4B**)
Instructor's Name	Maam Amber Madeeha Zeb

In Lab Tasks

5.1 Task 1:

Code the example of complex class given above and include the functions for addition, subtraction and multiplication of objects of complex class and return the object containing result. Test all the functions in main.

Solution:

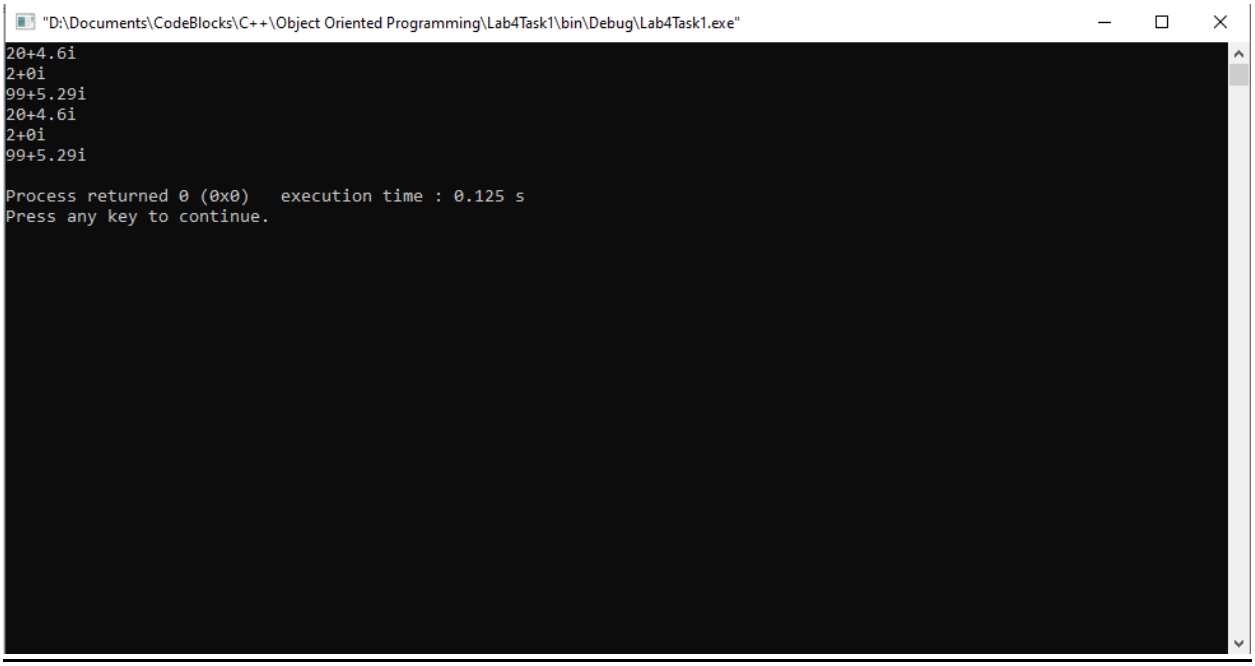
The code is given below,

```
#include<iostream>
using namespace std;
class Complex
{
private:
double real, imag;
public:
Complex() // Default Constructor
{
real = 0.0;
imag = 0.0;
}
// Two argument Constructor
Complex (double r, double im)
{
real = r;
imag = im;
}
void show()
{
cout<<real<<"+"<<imag<<"i"<<endl;
}

Complex Add (Complex &b)
{
return Complex(real + b.real, imag+ b.imag);
}
Complex Addwithoutref (Complex b)
{
return Complex(real + b.real, imag+ b.imag);
}
```

```
Complex subtr (Complex &b)
{
return Complex(real - b.real, imag - b.imag);
}
Complex subtrwithr (Complex b)
{
return Complex(real - b.real, imag - b.imag);
}
Complex multiply (Complex &b)
{
return Complex(real * b.real, imag * b.imag);
}
Complex multiplywregf (Complex b)
{
return Complex(real * b.real, imag * b.imag);
}
};
int main()
{
Complex C1(11, 2.3);
Complex C2(9, 2.3);
Complex C3;
C3 = C1.Add(C2);
C3.show();
C3= C1.subtr(C2);
C3.show();
C3=C1.multiply(C2);
C3.show();
C3=C1.Addwithoutref(C2);
C3.show();
C3=C1.subtrwithr(C2);
C3.show();
C3=C1.multiplywregf(C2);
}
```

Console Output is shown below.



5.2 Task:2

Modify the above task such that; define the member function show() outside the class.

Solution:

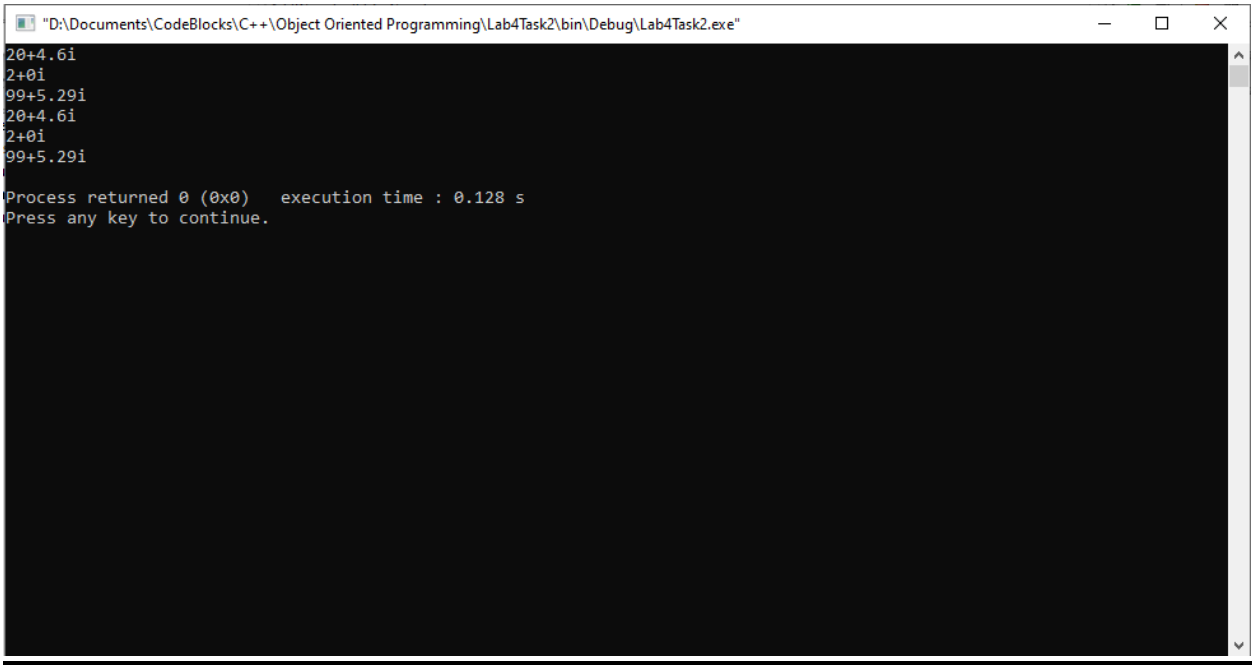
I am attaching my code below,

```
#include<iostream>
using namespace std;
class Complex
{
private:
double real, imag;
public:
Complex() // Default Constructor
{
real = 0.0;
imag = 0.0;
}
// Two argument Constructor
Complex (double r, double im)
{
real = r;
imag = im;
}
void show();
Complex Add (Complex &b)
{
return Complex(real + b.real, imag+ b.imag);
}
Complex Addwithoutref (Complex b)
{
return Complex(real + b.real, imag+ b.imag);
}
```

```
Complex subtr (Complex &b)
{
return Complex(real - b.real, imag - b.imag);
}
Complex subtrwithr (Complex b)
{
return Complex(real - b.real, imag - b.imag);
}
Complex multiply (Complex &b)
{
return Complex(real * b.real, imag * b.imag);
}
Complex multiplywregf (Complex b)
{
return Complex(real * b.real, imag * b.imag);
}

};
void Complex:: show()
{
cout<<real<<"+"<<imag<<"i"<<endl;
}
int main()
{
Complex C1(11, 2.3);
Complex C2(9, 2.3);
Complex C3;
C3 = C1.Add(C2);
C3.show();
C3= C1.subtr(C2);
C3.show();
C3=C1.multiply(C2);
C3.show();
C3=C1.Addwithoutref(C2);
C3.show();
C3=C1.subtrwithr(C2);
C3.show();
C3=C1.multiplywregf(C2);
C3.show();
}
```

The result for this program is shown below,


"D:\Documents\CodeBlocks\C++\Object Oriented Programming\Lab4Task2\bin\Debug\Lab4Task2.exe"
20+4.6i
2+0i
99+5.29i
20+4.6i
2+0i
99+5.29i
Process returned 0 (0x0)   execution time : 0.128 s
Press any key to continue.

5.3 Task:3

Modify the task (5.1) by defining all the member functions outside the class definition.

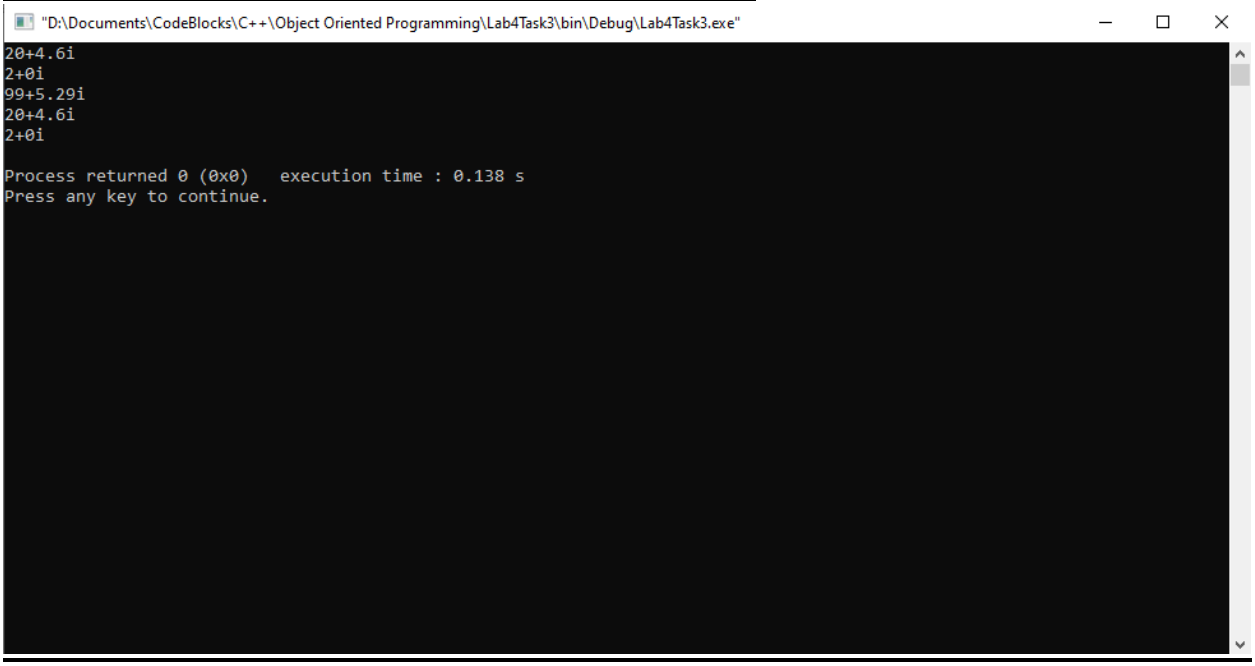
Solution:

I am attaching my code below,

```
#include<iostream>
using namespace std;
class Complex
{
private:
double real, imag;
public:
Complex() // Default Constructor
{
real = 0.0;
imag = 0.0;
}
// Two argument Constructor
Complex (double r, double im)
{
real = r;
imag = im;
}
void show()
{
cout<<real<<"+"<<imag<<"i"<<endl;
}
Complex multiply (Complex &b);
Complex Add (Complex &b)
{
return Complex(real + b.real, imag+ b.imag);
}
```

```
Complex Addwithoutref (Complex b)
{return Complex(real + b.real, imag+ b.imag);
}
Complex subtr (Complex &b)
{
return Complex(real - b.real, imag - b.imag);
}
Complex subtrwithr (Complex b)
{
return Complex(real - b.real, imag - b.imag);
}};
Complex Complex :: multiply(Complex &b)
{
return Complex(real * b.real, imag * b.imag);
}
int main()
{
Complex C1(11, 2.3);
Complex C2(9, 2.3);
Complex C3;
C3 = C1.Add(C2);
C3.show();
C3= C1.subtr(C2);
C3.show();
C3=C1.multiply(C2);
C3.show();
C3=C1.Addwithoutref(C2);
C3.show();
C3=C1.subtrwithr(C2);
}
```

The result for this program is shown below,



5.4 Task:4

Test the distance class example given above.

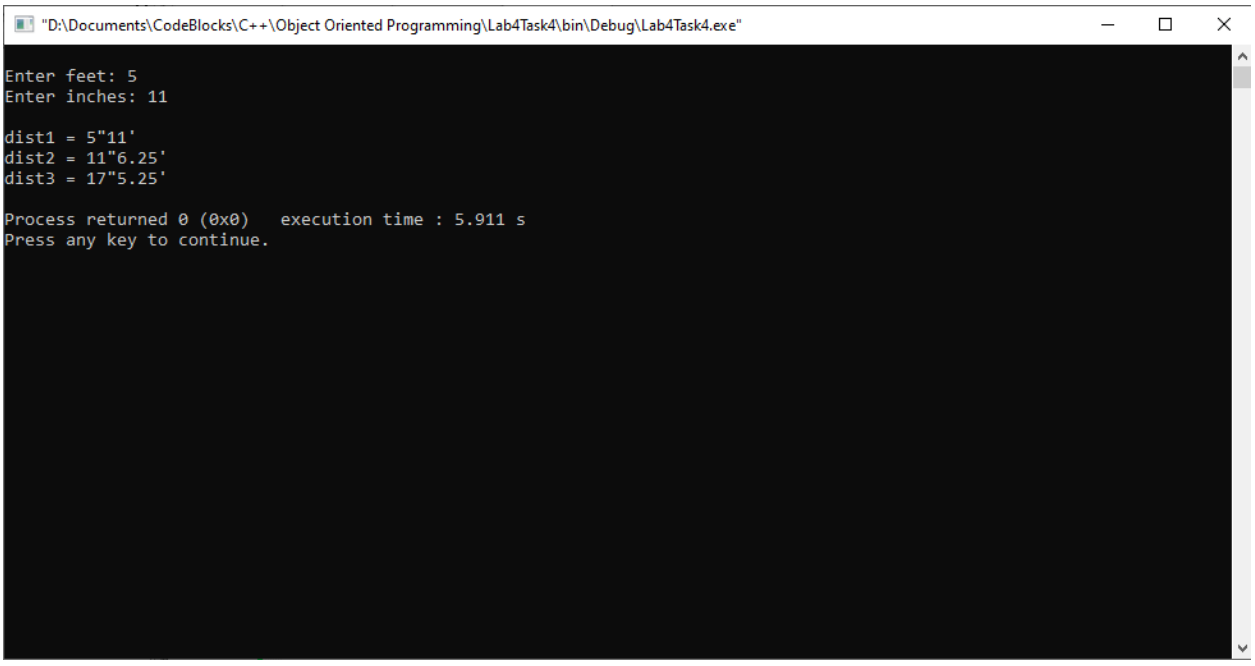
Solution:

I am attaching my code below,

```
#include<iostream>
using namespace std;
class Distance //English Distance class
{
private:
int feet;
float inches;
public:
//constructor (no args)
Distance() : feet(0), inches(0.0)
{ }
//constructor (two args)
Distance(int ft, float in) : feet(ft), inches(in)
{ }
void getdist() //get length from user
{
cout<<"\nEnter feet: ";
cin>>feet;
cout<<"Enter inches: ";
cin>>inches;
}
inline void showdist(); //display distance
void add_dist( Distance, Distance ); //declaration
};
//-----------------------------------------------------------
//definition of inline function which display distance
inline void Distance::showdist()
{
cout<<feet<<"\""<<inches<<"\'";
}
```

```
void Distance::add_dist(Distance d2, Distance d3)
{
inches = d2.inches + d3.inches; //add the inches
feet = 0; //(for possible carry)
if(inches >= 12.0) //if total exceeds 12.0,
{
//then decrease inches
inches -= 12.0; //by 12.0 and
feet++; //increase feet
} //by 1
feet += d2.feet + d3.feet; //add the feet
}
////////////////////////////////////////////////////////////////
int main()
{
Distance dist1, dist3; //define two Distance Objects
Distance dist2(11, 6.25); //define and initialize dist2
dist1.getdist(); //get dist1 from user
dist3.add_dist(dist1, dist2); //dist3 = dist1 + dist2
//display all lengths
cout<<"\ndist1 = ";
dist1.showdist();
cout<<"\ndist2 = ";
dist2.showdist();
cout<<"\ndist3 = ";
dist3.showdist();
cout<<endl;
}
```

The result for this program is shown below,



POST LAB

6.1 Question 1:

Modify the Distance class example of lab task (5.4) by including functions for subtraction and multiplication of distance class objects like addition.

Solution:

I am attaching my code below,

```
void Distance::Sub_dist(Distance d2, Distance d3)
{
inches = d2.inches - d3.inches;
feet = 0;
if(inches >= 12.0)
{

inches -= 12.0;
feet++;
}
feet += d2.feet - d3.feet;
}
void Distance::Mul_dist(Distance d2, Distance d3)
{
inches = d2.inches * d3.inches;
feet = 0;
if(inches >= 12.0)
{

inches -= 12.0;
feet++;
}
feet += d2.feet * d3.feet;
}
```

The result for this program is shown below,

_____THE END_____