Devanshu Surana
PC-23, Batch C1
1032210755

OOCCJ Lab Assignment 3A

Problem Statement:
Define a class complex consisting following:
Data members:
a) real
b) Imaginary part
Member Functions
a) One default constructor
b) function setcomplex() to set the value of real and Imaginary part.
c) function printcomplex() to display and
Four overloaded operator member functions:
1) Operator + to add two complex numbers
2) operator * to multiply two complex numbers
3) Operator - to subtract two complex numbers using friend function
4) Operator / to divide two complex numbers using friend function

Objectives:
1. To learn to create a class in C++
2. To learn constructor function and operator overloading in C++.
3. To learn friend function in C++.

Page No.
Date

FAQ's.

1) What is Inline function?
→ An inline function is a function that is expa
in-line at the point of its call. rather than be
like a regular function. i.e when complier enco
an inline function calls, it replaces the functi
with the entire body of the function.

2) Which operator links class to a member?
→ The scope resolution operator (::) links cla
member function in C++. It is used to defi
member function outside the class definition
to access static members, namespace me
etc. It can also be used to access overload
operators that have been defined as mem
functions.
Ex] void myclass :: myfunction () {
}
In the example "::" links 'myclass' to 'myfu

3) What is default access specifier in C++?
→ In C++, the default access specifier for a
'private'. This means all data members ar
data functions of the class are privat
access within the class only.

Ap
13/3/23

Devanshu Surana
PC-23, 1032210755
Batch C1

OOCJ Lab Assignmen

Problem statement: Wri
Interfaces Motorbike
consists of the attr
-ance (). Cycle inter
These interfaces are
Calculate total d

Objective:
1) To study abstract c
2. To study interfaces

Theory:
1. Java Abstraction:
- A process of hiding
only functionality
shows only essentia
internal details.
Ex: sending SMS
the message. You
the message del
It focuses on wh
does it.

There are 2 w
1) Abstract class (

Name :- Devanshu Surana
Roll No: PC -23
PRN :- 1032210755

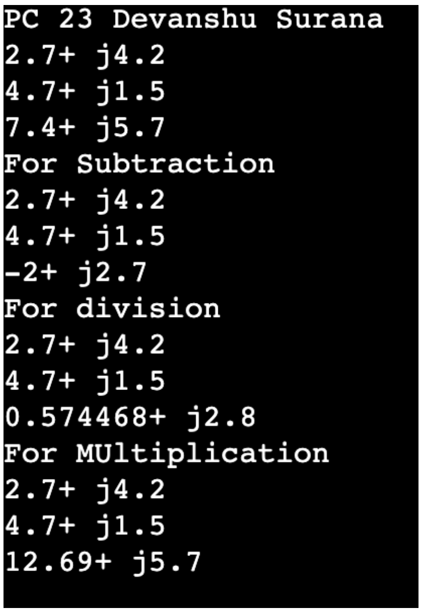
OOCCJ LAB ASSIGNMENT 3A

CODE:-

```
#include <iostream>
using namespace std;
class complex
{
 float real;
 float image;
public:
   complex ()
 {
   real = 0;
   image = 0;
 }

 complex (float x, float y)
 {
   real = x;
   image = y;
 }

 friend complex operator+ (complex & c1, complex & c2);
 friend complex operator- (complex & c1, complex & c2);
 complex operator / (complex com)
 {
   complex t;
   t.real = (real) / (com.real);
   t.image = (image) / (com.image);
   return t;
 }

 complex operator * (complex com)
 {
   complex t;
   t.real = (real) * (com.real);
   t.image = (image) + (com.image);
   return t;
 }

 void display (void);
```

```
};

complex
operator + (complex & ca, complex & cb)
{
  complex t;
  t.real = ca.real + cb.real;
  t.image = ca.image + cb.image;
  return t;
}

complex
operator - (complex & ca, complex & cb)
{
  complex t;
  t.real = ca.real - cb.real;
  t.image = ca.image - cb.image;
  return t;
}

void
complex::display (void)
{
  cout << real << "+ j" << image << "\n";
}

int
main ()
{
  cout << "PC 23 Devanshu Surana" << endl;
  complex ca1, ca2, ca3;
  ca1 = complex (2.7, 4.2);
  ca2 = complex (4.7, 1.5);
  ca3 = ca1 + ca2;
  ca1.display ();
  ca2.display ();
  ca3.display ();
  cout << "For Subtraction" << endl;
  complex cs1, cs2, cs3;
  cs1 = complex (2.7, 4.2);
  cs2 = complex (4.7, 1.5);
  cs3 = cs1 - cs2;
  cs1.display ();

  cs2.display ();
  cs3.display ();
  cout << "For division" << endl;
  complex cd1, cd2, cd3;
  cd1 = complex (2.7, 4.2);
  cd2 = complex (4.7, 1.5);
  cd3 = cd1 / cd2;
  cd1.display ();
  cd2.display ();
  cd3.display ();
  cout << "For MUltiplication" << endl;
  complex cm1, cm2, cm3;
  cm1 = complex (2.7, 4.2);
  cm2 = complex (4.7, 1.5);
  cm3 = cm1 * cm2;
  cm1.display ();
  cm2.display ();
  cm3.display ();
  return 0;
}
```

OUTPUT:-



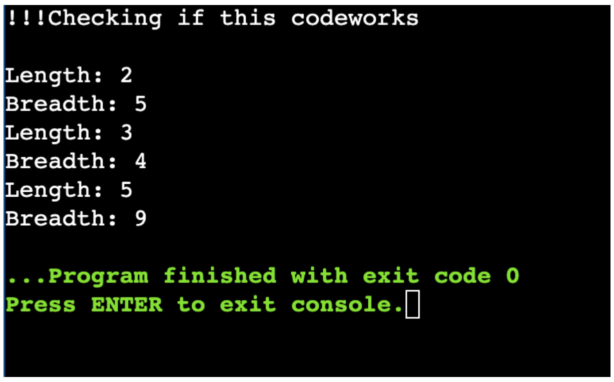
```
PC 23 Devanshu Surana
2.7+ j4.2
4.7+ j1.5
7.4+ j5.7
For Subtraction
2.7+ j4.2
4.7+ j1.5
-2+ j2.7
For division
2.7+ j4.2
4.7+ j1.5
0.574468+ j2.8
For MUltiplication
2.7+ j4.2
4.7+ j1.5
12.69+ j5.7
```

3.1:-

CODE:-

```
#include <iostream>
using namespace std;
class rectangle
{
  int length;
  int breadth;
public:
   rectangle ()
  {
   length = 0;
   breadth = 0;
  }
  rectangle (int l, int b)
  {
   length = l;
   breadth = b;
  }
  rectangle operator + (rectangle rec)
  {
   rectangle r;
   r.length = length + rec.length;
   r.breadth = breadth + rec.breadth;
   return r;
  }
  void display (void);
};

void
rectangle::display (void)
{
  cout << "\nLength: " << length;
  cout << "\nBreadth: " << breadth;
}

int
main ()
{

  cout << "!!!Checking if this codeworks" << endl;
  rectangle r1, r2, r3;
  r1 = rectangle (2, 5);
  r2 = rectangle (3, 4);
  r3 = r1 + (r2);
  r1.display ();
  r2.display ();
  r3.display ();
  return 0;
}
```

OUTPUT:-

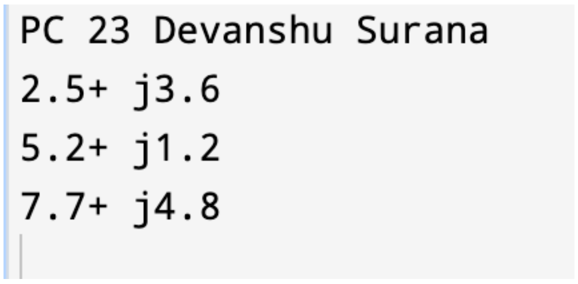


```
!!!Checking if this codeworks

Length: 2
Breadth: 5
Length: 3
Breadth: 4
Length: 5
Breadth: 9

...Program finished with exit code 0
Press ENTER to exit console.
```

3.2:-
CODE:-

```
#include <iostream>
using namespace std;
class complex{
        float real;
        float image;
public:
        complex(){}
        complex(float x,float y){
                real=x;
                image=y;
        }
        friend complex operator+(complex& c1,complex& c2);
        void display(void);
};
complex operator +(complex& ca,complex& cb){
        complex t;
        t.real=ca.real+cb.real;
        t.image=ca.image+cb.image;
        return t;
}

void complex::display(void){
                cout<<real<<"+ j"<<image<<"\n";
}
int main() {
        cout << "PC 23 Devanshu Surana" << endl;
        complex c1,c2,c3;
        c1=complex(2.5,3.6);
        c2=complex(5.2,1.2);
        c3= c1+c2;
        c1.display();
        c2.display();
        c3.display();
        return 0;
}
```

OUTPUT:-



```
PC 23 Devanshu Surana
2.5+ j3.6
5.2+ j1.2
7.7+ j4.8
```

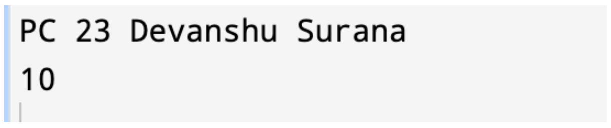
3.3:-

CODE:-

```
                #include <iostream>
using namespace std;

class beta;
class alpha{
                int data;
public:
                alpha(){
                        data = 3;
                }
                friend int frifunc(alpha,beta);
};
class beta{
                int data;
public:
                beta(){
                        data = 7;
                }
                friend int frifunc(alpha,beta);
};

int frifunc(alpha a,beta b){
                return (a.data + b.data);
}
```

```
int main() {
                    cout << "PC 23 Devanshu Surana" << endl;
                    alpha aa;
                    beta bb;
                    cout<<frifunc(aa,bb)<<endl;
                    return 0;
}
```

OUTPUT:-



```
PC 23 Devanshu Surana
10
```