**CSE208L Object Oriented Programming Lab**

**LAB # 2**

****

**2020**

**Submitted to:**

**Engr. Sumayya Salahuddin**

**Submitted by:**

**TAYYABA**

**Registration No :**

**19PWCSE1854**

**Semester: 3nd**

**Class Section:** **C**

“On my honor, as student of University of Engineering and Technology,

I have neither given nor received unauthorized assistance on this

academic work.”

Dec, 3 , 2020

**Department of Computer Systems Engineering**

**University of Engineering and Technology, Peshawar**

**University of Engineering and Technology, Peshawar**

**Objectives of the Lab:**

Objectives of the lab are to:

* Understand how class object can be passed and returned from class member function.
* Write a class with member function having objects as arguments.
* Write a class with member function that return object.
* Test member function effectively using given test cases.

# **ACTIVITY # 01**

**Title:**

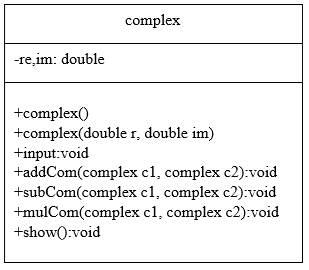
Make a class for complex number and model it uses real and imaginary part.

**Problem analysis:**

Create a class, **complex** that contains a twice double field, **real & imaginary**. Define a constructor that takes no parameters. The **real & imaginary** field should be set to the value 0 in the constructor. Define another constructor that takes two parameters. Define a function **input** for take the complex number from user. **Define** the functions **addCom, subCom** and **mulCom** which is used to add, subtract and multiply the two complex numbers . Define another function **show** for displaythe result of addition, subtraction & multiplication.

**Algorithm:**

UML diagram for the above problem is given below:

* First make class complex.
* Declare real and imaginary as private double field.
* Define no argument constructor to set value of real & imaginary to 0.
* Define argument constructor to set value of real & imaginary to given values.
* Define input function for taking the real and imaginary part of objects to the user.
* Define add, subtract and multiply function for performing some arithmetic operations.
* Define show function to display the result of function.
* In main function, make objects of complex to demonstrate the use of complex number.
* Call each function one after the other and display the show function as shown in the flow chart.

**Flowchart:**

start



Private double re, im

start

re = 0, im =0

end

complex()

start

end

complex(double r,double i)

re = r, im =i

start

re=c1.re+c2.re

im=c1.im+c2.im

end

void addCom(complex c1,complex c2)

start

re=c1.re-c2.re

im=c1.im-c2.im

end

void subCom(complex c1,complex c2)

start

re=((c1.re\*c2.re)-(c1.im\*c2.im)) im=((c1.re\*c2.im)+(c1.im\*c2.re))

end

void addCom(complex c1,complex c2)

start

print re, im

end

void show()

start

end

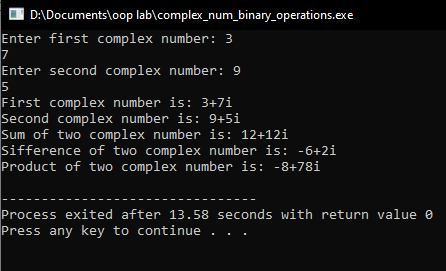
cin>>re>>im

void input()

end

**In C++**

**Source code: Output**

#include<iostream>

using namespace std;

//create a class

class complex{

private:

double re,im;

public:

complex();

complex(double r, double i);

void addCom(complex c1, complex c2);

void subCom(complex c1, complex c2);

void mulCom(complex c1, complex c2);

void input();

void show();

};

complex::complex():re(0),im(0){}; //constructor

complex::complex(double r, double i):re(r),im(i){}; //constructor

void complex::input()

{

cin>>re;

cin>>im;

}

void complex::addCom(complex c1, complex c2)

{

re=c1.re+c2.re;

im=c1.im+c2.im;

}

void complex::subCom(complex c1, complex c2)

{

re=c1.re-c2.re;

im=c1.im-c2.im;

}

void complex::mulCom(complex c1, complex c2)

{

re=((c1.re\*c2.re)-(c1.im\*c2.im));

im=((c1.re\*c2.im)+(c1.im\*c2.re));

}

void complex::show()

{

cout<<re<<"+"<<im<<"i"<<endl;

}

int main()

{

complex c1, c2,c;

cout<<"Enter first complex number: ";

c1.input();

cout<<"Enter second complex number: ";

c2.input();

cout<<"First complex number is: ";c1.show();

cout<<"Second complex number is: ";c2.show();

c.addCom(c1,c2);

cout<<"Sum of two complex number is: ";c.show();

c.subCom(c1,c2);

cout<<"Difference of two complex number is: ";c.show();

c.mulCom(c1,c2);

cout<<"Product of two complex number is: ";c.show();

return 0;

}

**Conclusion:**

This program helps us in understanding the basic concepts of classes and objects in different languages. It acts as a base for us and helps us in preparing ourselves for the higher level of programming. We get to know about how to pass argument to the function of class in OOP with the help of this program.

# **Activity # 02**

**Title:**

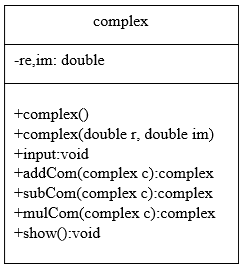
Make a class for complex number and model it uses real and imaginary part.

**Problem analysis:**

Create a class, **complex** that contains a twice double field, **real & imaginary**. Define a constructor that takes no parameters. The **real & imaginary** field should be set to the value 0 in the constructor. Define another constructor that takes two parameters. Define a function **input** for take the complex number from user. **Define** the functions **addCom, subCom** and **mulCom** which is used to add, subtract and multiply the two complex numbers . Define another function **show** for displaythe result of addition, subtraction & multiplication.

**Algorithm:**

UML diagram for the above problem is given below:

* First make class complex.
* Declare real and imaginary as private double field.
* Define no argument constructor to set value of real & imaginary to 0.
* Define argument constructor to set value of real & imaginary to given values.
* Define input function for taking the real and imaginary part of objects to the user.
* Define add, subtract and multiply function for performing some arithmetic operations and return the result.
* Define show function to display the result of function.
* In main function, make objects of complex to demonstrate the use of complex number.
* Call each function one after the other and display the show function as shown in the flow chart.

**Flowchart:**

Start



Private double re, im

start

re = 0, im =0

end

complex()

start

end

complex(double r,double i)

re = r, im =i

start

re=c1.re+c2.re

im=c1.im+c2.im

end

void addCom(complex c1,complex c2)

start

re=c1.re-c2.re

im=c1.im-c2.im

end

void subCom(complex c1,complex c2)

start

re=((c1.re\*c2.re)-(c1.im\*c2.im)) im=((c1.re\*c2.im)+(c1.im\*c2.re))

end

void mulCom(complex c1,complex c2)

start

print re, im

end

void show()

start

end

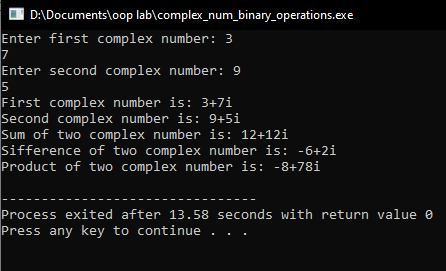
cin>>re>>im

void input()

end

**In C++**

**Source code: Output**

#include<iostream>

using namespace std;

class complex{

private:

double re,im;

public:

complex();

complex(double r, double i);

complex addCom(complex c1);

complex subCom(complex c1);

complex mulCom(complex c1);

void input();

void show();

};

complex::complex():re(0),im(0){}; //constructor

complex::complex(double r, double i):re(r),im(i){}; //constructor

void complex::input()

{

cin>>re;

cin>>im;

}

complex complex::addCom(complex c1)

{

complex temp;

temp.re=c1.re+re;

temp.im=c1.im+im;

return temp;

}

complex complex::subCom(complex c1)

{

complex temp;

temp.re=c1.re-re;

temp.im=c1.im-im;

return temp;

}

complex complex::mulCom(complex c1)

{

complex temp;

temp.re=((c1.re\*re)-(c1.im\*im));

temp.im=((c1.re\*im)-(c1.im\*re));

return temp;

}

void complex::show()

{

cout<<re<<"+"<<im<<"i"<<endl;

}

int main()

{

complex c1, c2,c;

cout<<"enter first complex number: ";

c1.input();

cout<<"enter second complex number: ";

c2.input();

cout<<"first complex number is: ";c1.show();

cout<<"second complex number is: ";c2.show();

c=c2.addCom(c1);

cout<<"sum of two complex number is: ";c.show();

c=c2.subCom(c1);

cout<<"difference of two complex number is: ";c.show();

c=c2.mulCom(c1);

cout<<"product of two complex number is: ";c.show();

return 0;

}

# **Activity # 03**

**Title:**

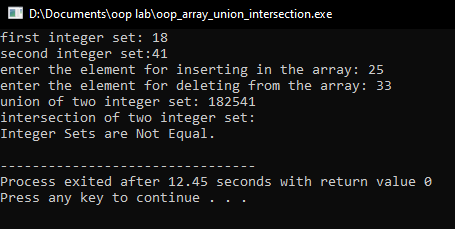
Make a class for Integer Set and model it uses integer array.

**Problem analysis:**

Create a class called ***IntegerSet*** that models how to use integer sets. The program creates an integer array, displays the integer set, makes a NewIntegerSet function to initialize the integer array, then make a union &intersection function for union and intersect the two arrays by making two objects of the class and then displays the union and intersection result. Also make two another function for inserting and deleting the element in the array.

**Source code:**

#include <iostream>

#include <cstdlib> **Output:**

using namespace std;

const int SIZE=50;

class IntegerSet{

private:

int Array[SIZE];

public:

IntegerSet(){

for (int i=0;i<SIZE;i++)

{

Array[i]=0;

}

}

void NewIntegerSet(int n[])

{

for(int i=0;i<SIZE;i++)

{

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

{

if (n[j]==i)

Array[i]=1;

else

Array[i]=0;

}

}

}

void SetPrint()

{

for (int i=0;i<SIZE;i++)

{

if (Array[i]==1)

{

cout<<i;

}

}

cout<<endl;

}

void InsertElement(int k)

{

Array[k]=1;

}

void DeleteElement(int m)

{

Array[m]=0;

}

IntegerSet UnionOfIntegerSet(IntegerSet N)

{

IntegerSet temp;

for(int i=0;i<SIZE;i++)

{

if (N.Array[i] || Array[i])

temp.Array[i]=1;

else

temp.Array[i]=0;

}

return temp;

}

IntegerSet IntersectionOfIntegerSet(IntegerSet N)

{

IntegerSet temp;

for(int i=0;i<SIZE;i++)

{

if ((!(N.Array[i]) || !(Array[i])))

{

temp.Array[i]=0;

}

else

temp.Array[i]=1;

}

return temp;

}

void IsEqualTo(IntegerSet N)

{

int counter=0;

for(int i=0;i<SIZE;i++)

{

if (N.Array[i]==Array[i])

counter++;

}

if (counter==SIZE)

cout<<"Integer Sets are Equal."<<endl;

else

cout<<"Integer Sets are Not Equal."<<endl;

}

};

int main()

{

int num;

int x[SIZE],y[SIZE];

for(int i=0;i<SIZE;i++)

{

x[i]=rand()%50;

}

for(int i=0;i<SIZE;i++)

{

y[i]=rand()%50;

}

IntegerSet i1,i2,i3;

cout<<"first integer set: ";

i1.NewIntegerSet(x);

i1.SetPrint();

i2.NewIntegerSet(y);

cout<<"second integer set:";

i2.SetPrint();

cout<<"enter the element for inserting in the array: ";

cin>>num;

i1.InsertElement(num);

cout<<"enter the element for deleting from the array: ";

cin>>num;

i1.DeleteElement(num);

i3= i1.UnionOfIntegerSet(i2);

cout<<"union of two integer set: ";

i3.SetPrint();

i3 = i1.IntersectionOfIntegerSet(i2);

cout<<"intersection of two integer set: ";

i3.SetPrint();

i1.IsEqualTo(i2);

return 0;

}