OBJECT ORIENTED PROGRAMMING LAB

Experiment-2

-by Chander Jindal

Aim: *Write a program to perform addition of two complex numbers using constructor overloading. The first constructor which takes no argument is used to create objects which are not initialized, second which takes one argument is used to initialize real and imag parts to equal values and third which takes two argument is used to initialized real and imag to two different values.*

Performance Instructions:

A class constructor is a special member function of a class that is executed whenever we create new objects of that class.
A constructor will have exact same name as the class and it does not have any return type at all, not even void. Constructors can be very useful for setting initial values for certain member variables.

```
class Complex{
int real,imag;
public:
Complex() {} // Constructor.
```

Parameterized Constructor
A default constructor does not have any parameter, but if you need, a constructor can have parameters. This helps you to assign initial value to an object at the time of its creation.

```
Complex(int r) // Parameterised constructor for equal values.
{
real=r;
imag=r;
}
Complex(int r,int i) // Parameterised constructor for different values.{
real=r;
imag=i;
}
```

Copy Constructor
The copy constructor is a constructor which creates an object by initializing it with an object of the same class, which has been created previously.

```
Complex(Complex &c) //Copy Constructor.
{
real=c.real;
imag=c.imag;
}
```

Addition of Complex numbers

```
Complex sum(Complex obj1,Complex obj2){
Complex obj3;
obj3.real=obj1.real+obj2.real;
obj3.imag=obj1.imag+obj2.imag;
return obj3;
```

}

#Code

```
#include<bits/stdc++.h>
using namespace std;
class Complex{
	public:
		int real, imag;
	Complex(){}
	Complex(int tempreal, int tempimag){
		real = tempreal;
		imag = tempimag;
	}
	Complex addComp(Complex C1, Complex C2){
		Complex temp;
		temp.real = C1.real + C2.real;
		temp.imag = C1.imag + C2.imag;
		return temp;
	}
};
int main(){
	Complex C1,C2;
	C1.real = rand()%100, C1.imag = rand()%100;
	C2.imag = rand()%100, C2.real = rand()%100;
	cout<<"Complex number 1 : "<<C1.real<< " + i"<<C1.imag<<endl;
	cout<<"Complex number 2 : "<<C2.real<<" + i"<< C2.imag<<endl;
	Complex C3;
	C3 = C3.addComp(C1, C2);
	cout<<"Sum of complex number : "<<C3.real<< " + i"<<C3.imag;
}
```