NAME: SATVIK DANDALE

BATCH: B1

DIV: TY-E

GR No: 1710797

Roll No: 13

/\*

Write a program to implement Complex class as follows:

Private: real\_value

imag\_value

Public:

Parameterized constructor(1 and 2 parameter)

Copy constructor

Getter and setter functions.

(Use of constructors and destructors)

\*/

#include<bits/stdc++.h>

using namespace std;

class Complex{

private:

float real\_value, img\_value;

public:

// Constructors

Complex(/\* DEFAULT \*/){

this->real\_value = this->img\_value = 0;

}

Complex(double single){

// In this case, the real and imaginary part will be set to the given number.

this->real\_value = this->img\_value = single;

}

Complex(int single){

// In this case, the real and imaginary part will be set to the given number.

this->real\_value = this->img\_value = single;

}

Complex(double real\_value, double img\_value){

this->real\_value = real\_value;

this->img\_value = img\_value;

}

Complex(int real\_value, double img\_value){

this->real\_value = real\_value;

this->img\_value = img\_value;

}

Complex(double real\_value, int img\_value){

this->real\_value = real\_value;

this->img\_value = img\_value;

}

Complex(int real\_value, int img\_value){

this->real\_value = real\_value;

this->img\_value = img\_value;

}

// Copy Constructor

Complex(Complex &c){

this->real\_value = c.real\_value;

this->img\_value = c.img\_value;

}

// Destructor

~Complex(){

delete this;

}

// SETTERS

void setReal(int real\_value){

this->real\_value = real\_value;

}

void setImg(int img\_value){

this->img\_value = img\_value;

}

void setReal(double real\_value){

this->real\_value = real\_value;

}

void setImg(double img\_value){

this->img\_value = img\_value;

}

// GETTERS

float getReal(){ return this->real\_value; }

float getImg(){ return this->img\_value; }

// Displaying the complex number

void display(){

cout<<this->real\_value<<" + "<<this->img\_value<<"i";

}

};

int main(){

// Creating two different Complex Numbers

Complex \*c1 = new Complex(1, 22);

Complex \*c2; // Getter

cout<<"c1 is (using getters): "<<c1->getReal()<<" + "<<c1->getImg()<<"i\n";

// Setter

c1->setImg(4.5);

cout<<"c1 now is (using setters): ";

c1->display();

cout<<endl;

// Copy Constructor

c2 = new Complex(\*c1);

cout<<"c2 after copying from c1 is (using copy constructor and display function): ";

c2->display();

cout<<endl;

cout<<"c1 is (using getters): "<<c1->getReal()<<" + "<<c1->getImg()<<"i\n";

// Setter

c1->setImg(4.5);

cout<<"c1 now is (using setters): ";

c1->display();

cout<<endl;

// Copy Constructor

c2 = new Complex(\*c1);

cout<<"c2 after copying from c1 is (using copy constructor and display function): ";

c2->display();

cout<<endl;

}

OUTPUT:

c1 is (using getters): 1 + 22i

c1 now is (using setters): 1 + 4.5i

c2 after copying from c1 is (using copy constructor and display function): 1 + 4.5i