**Question 3**

**Program Code**

#include<iostream>

using namespace std;

class complex

{

float real,img;

public:

complex() //default constructor

{

real=0;

img=0;

}

complex(float a,float b) //parameterized constructor

{

real=a;

img=b;

}

complex operator+(complex c1) //operator overloading i.e use of binary operator+

{

complex temp;

temp.real=real+c1.real;

temp.img=img+c1.img;

return temp;

}

friend ostream &operator<<(ostream &out,complex &c) // use of friend function

{

out<<c.real<<"+"<<c.img<<"i";

return out;

}

friend istream &operator>>(istream &in,complex &c) // use of friend function

{

in>>c.real>>c.img;

return in;

}

};

int main()

{

complex c1,c2,c3;

cout<<"\nEnter the 1st complex number:"<<endl;

cin>>c1;

cout<<"\nEnter the 2nd complex number:"<<endl;

cin>>c2;

c3=c1+c2;

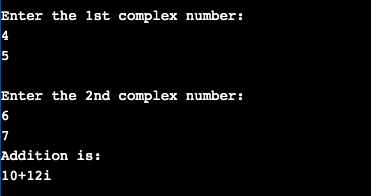
cout<<"Addition is:"<<endl;

cout<<c3<<endl;

return 0;

}

**Output**



**Question 4**

**Program Code**

#include<iostream>

#include<conio.h>

using namespace std;

class Example

{

// Member Variable Declaration

int a, b;

public:

//Normal Constructor with Argument

Example(int x, int y)

{

// Assign Values In Constructor

a = x;

b = y;

cout << "\nIm Constructor";

}

//Copy Constructor with Obj Argument

Example(const Example& obj)

{

// Assign Values In Constructor

a = obj.a;

b = obj.b;

cout << "\nIm Copy Constructor";

}

void Display()

{

cout << "\nValues :" << a << "\t" << b;

}

};

int main()

{

//Normal Constructor Invoked

Example Object(10, 20);

//Copy Constructor Invoked - Method 1

Example Object2(Object);

//Copy Constructor Invoked - Method 2

Example Object3 = Object;

Object.Display();

Object2.Display();

Object3.Display();

// Wait For Output Screen

getch();

return 0;

}

**Output**

