CHAUDHARY HAMDAN

1905387

OOP LAB-5

Date : 04-09-2020

Lab-5

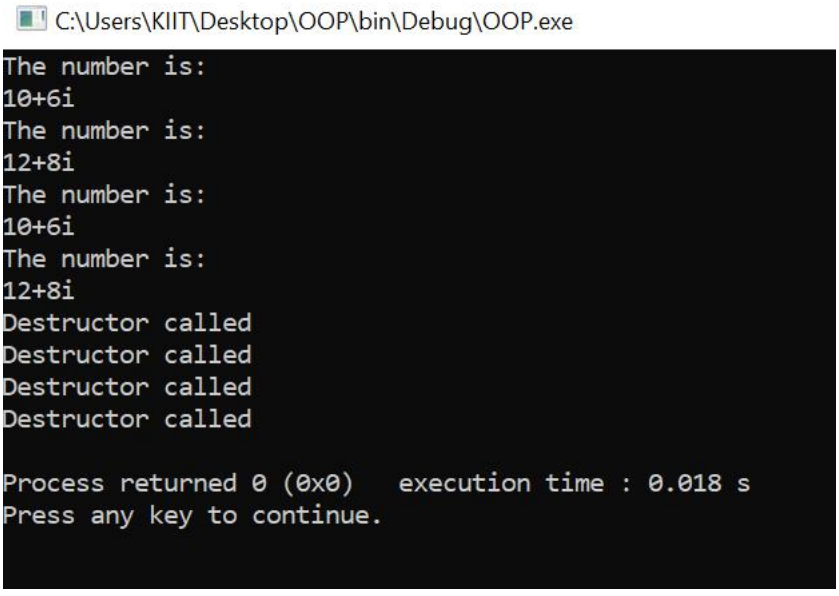
Topic: Constructors and Destructors

i. Create a class complex which stores real and imaginary part of a complex number. Include all types of constructors and destructor. The destructor should display a message about the destructor being invoked. Create objects using different constructors and display them.
ii. Create a class which stores time in hh:mm format. Include all the constructors. The parameterized constructor should initialize the minute value to zero, if it is not provided.
iii. Create a class which stores a sting and its length as data members. Include all the constructors. Include a member function to join two strings and display the concatenated string.
iv. WAP to demonstrate the order of call of constructors and destructors for a class.
v. WAP to count number of objects created from a class using concept of static data members and static member function.

1.

```
#include<iostream>
using namespace std;
class complex
{
	int real,img;
	public:
		complex()
		{
			real=10;
			img=6;
		}
		complex(int a, int b)
		{
			real=a;
			img=b;
		}
		complex(const complex &x,const complex &y)
		{
			real=x.real;
			img=y.img;
		}
		void display()
		{
			cout<<"The number is:\n";
			cout<<real<<"+"<<img<<"i"<<endl;
		}
```

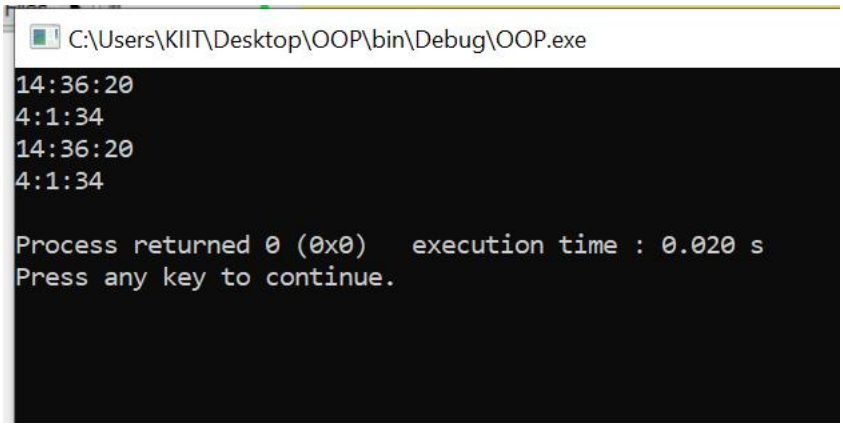
```
        ~complex()
        {
            cout<<"Destructor called"<<endl;
        }
};
int main()
{
    complex c1;
    c1.display();
    complex c2(12,8);
    c2.display();
    complex c3(c1);
    c3.display();
    complex c4(c2);
    c4.display();

    return 0;
}
```



2.

```
#include<iostream>
using namespace std;
class time1{
        int hour,minu,sec;
        public:
        time1()
        {
          hour=14;
          minu=3;
          sec=2000;
        }
        time1(int h,int s,int m=0){
          hour=h;
          minu=m;
          sec=s;
        }
        time1(const time1 &x,const time1 &y,const time1 &z){
          hour=x.hour;
          minu=y.minu;
          sec=z.sec;
        }
        void display(){
          minu=minu+(sec/60);
          sec=sec%60;
          hour=hour+(minu/60);
          minu=minu%60;
          cout<<hour<<":"<<minu<<":"<<sec<<endl;
        }
};
```

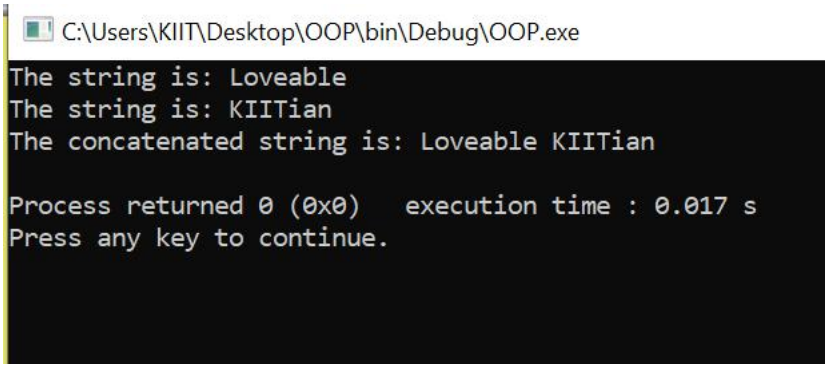
```
int main()
{
	time1 t1;
	t1.display();
	time1 t2(4,94);
	t2.display();
	time1 t3(t1);
	t3.display();
	time1 t4(t2);
	t4.display();
	return 0;
}
```



3.

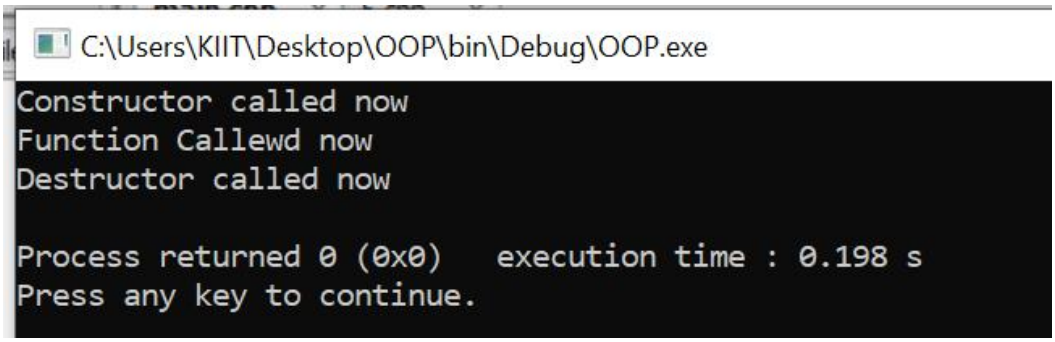
```
#include<iostream>
#include<string.h>
using namespace std;
class mystring
{
	int len;
	char *name;
	public:
	mystring()
	{
		len=0;
		name=new char;
	}
	mystring(char *s)
	{
		name=new char[strlen(s)+1];
		len=strlen(s);
		strcpy(name, s);
	}

	~mystring()
	{
		delete name;
	}
	void join(mystring &s1, mystring &s2)
	{
		len=s1.len+s2.len;
		name=new char[len+1];
```

```
        strcpy(name, s1.name);
        strcat(name, s2.name);
    }
    void display()
    {
        cout<<"The string is: "<<name<<endl;
    }
    void display1()
    {
        cout<<"The concatenated string is: "<<name<<endl;
    }
};

int main()
{
    mystring c1("Loveable "), c2("KIITian"), c3;
    c3.join(c1,c2);
    c1.display();
    c2.display();
    c3.display1();
    return 0;
}
```



4.

```
#include <iostream>

using namespace std;

class A

{

    public:

    A()

    {

        cout << "Constructor called now\n";

    }

    void fun()

    {

        cout << "Function Callewd now\n";

    }

    ~A()

    {

        cout << "Destructor called now\n ";

    }

};


int main()

{

    A ob;

    ob.fun();


    return 0;

}
```



5.

```
#include <iostream>
using namespace std;
class A
{
	static int c;
	public :
	A()
	{
		c++;
		cout << "Object Created " << endl;
	}
	static void display()
	{
		cout << c	<< endl;;
	}
};

int A :: c;

int main()
{
	A ob1;
	A ob2;
	A ob3;
	A ob4;

	cout << "\nObjects Created are " ;
	A :: display();
```

```
    A ob5;
    cout << "\nObjects Created are " ;
    A :: display();

    return 0;
}
```