NAME-SHAUNAK CHANDRA

ROLL-2005757

BRANCH-CSE (SLOT-1)

https://github.com/Kingsky1t/OOP_Lab_2005757

1) 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 amessage about the destructor being invoked. Create objects using different
constructors and display them.

```
#include <iostream>
using namespace std;

class Complex {
	int real;
	int img;
	public:
	Complex() {
		real=0;
		img=0;
	}
	Complex(int a,int b) {
		real=a;
		img=b;
	}
	Complex(Complex &a) {
		real=a.real;
		img=a.img;
	}
	~Complex() {
		cout<<"destructor called"<<endl;
	}
	void input() {
		cout<<"enter the real and imaginary part:";
		cin>>real>>img;
	}
	void display() {
		cout<<"the complex no. is "<<real<<"+i"<<img<<endl;
	}
};

int main() {
	Complex c1;
	c1.display();
	Complex c2(3,4);
	c2.display();
	Complex c3=c2;
	c3.display();
}
```

OUTPUT:

```
the complex no. is 0+i0
the complex no. is 3+i4
the complex no. is 3+i4
destructor called
destructor called
destructor called
```

2) 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.

```
#include <iostream>
using namespace std;

class Time {
        int hh;
        int mm;
        public:
        Time() {
                hh=0;
                mm=0;
        }
        Time(int h,int m=0) {
                hh=h;
                mm=m;
        }
        Time(Time &a) {
                hh=a.hh;
                mm=a.mm;
        }
        ~Time() {
                cout<<"destructor called"<<endl;
        }
        void input() {
                cout<<"enter the time in mins and sec:";
                cin>>hh>>mm;
        }
        void display() {
                cout<<"the time is "<<hh<<" hours and "<<mm<<" minutes"<<endl;
        }
};

int main() {
        char ch;
        Time t1;
        t1.display();
        cout<<"do you want to enter minutes? y or n;";
        cin>>ch;
        if(ch=='y') {
                t1.input();
                Time t3=t1;
                t3.display();
        }
        else {
                int min;
                cout<<"enter hours:";
                cin>> min;
                Time t2(min);
                Time t3=t2;
                t3.display();
        }


}
```

OUTPUT:

```
the time is 0 hours and 0 minutes
do you want to enter minutes? y or n:n
enter hours:11
the time is 11 hours and 0 minutes
destructor called
destructor called
destructor called
```

3) 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.

```
#include <iostream>
#include <string.h>
using namespace std;

class String {
        int len;
        char *str;
        public:
        String() {
                str=new char[100];
                len=0;
        }

        String(char s[100],int l) {
                str=s;
                len=l;
        }

        void input() {
                cout<<"enter a string:";
                cin.getline(str,100);
        }

        void concatenate(String a,String b) {
                len=a.len+b.len;
                str=strcat(a.str,b.str);
        }
}
        void display() {
                cout<<str;
        }

        ~String() {
        }
};

int main() {
        String s1;
        s1.input();
        char ch[100];
        cout<<"enter a string:";
        cin.getline(ch,100);
        String s2(ch,5);
        String s3;
        s3.concatenate(s1,s2);
        s3.display();
        return 0;
}
```

OUTPUT:

```
enter a string:HELLO
enter a string:SHAUNAK
HELLOSHAUNAK
```

4) WAP to demonstrate the order of call of constructors and destructors for a class.

```
#include <iostream>
using namespace std;

class test {
        public:
        test() {
                cout<<"constructor initialized"<<endl;
        }
        ~test() {
                cout<<"destructor initialized"<<endl;
        }
};
int main() {
        cout<<"object one:"<<endl;
        test ob1;
        cout<<"object two:"<<endl;
        test ob2;
        return 0;
}
```

OUTPUT:

```
object one:
constructor initialized
```

```
object two:
constructor initialized
destructor initialized
destructor initialized
```

5) WAP to count number of objects created from a class using concept of static data members and static member function.

```
#include <iostream>
using namespace std;

class test {
    int n;
    public:
    static int count;
    public:
    test() {
        n=0;
        cout<<"object created"<<endl;
        count++;
    }

    ~test() {
        cout<<"Object destroyed"<<endl;
        count--;
        cout<<"no. of objects left:"<<count<<endl;
    }

    static void display(void)
    {
        cout<<"No. of objects:"<<count<<endl;
    }
};
int test::count;

int main() {
    test ob1;
    test ob2;
    test ob3;
    test::display();
    return 0;
}
```

OUTPUT:

```
object created
object created
object created
No. of objects:3
Object destroyed
no. of objects left:2
Object destroyed
```

```
no. of objects left:1
Object destroyed
no. of objects left:0
```