Object Oriented Programming

Practice Session 1

Dr. Naveed Anwar Bhatti

Webpage: naveedanwarbhatti.github.io

Question 1

Write a program with a **mother** class and an inherited **daugther** class. Both of them should have a method void **display ()** that prints a message (different for mother and daugther). In the main define a daughter and call the **display()** method on it.

Question 1: Solution

```
class mother{
public:
        void display ()
        {
                cout << "mother: display function\n";
        }
};

class daughter : public mother{
public:
        void display ()
        {
                cout << "daughter: display function\n\n";
        }
};

int main ()
{
        daughter hina;
        hina.display();
        return 0;
}
```

Question 1 (modified)

Write a program with a **mother** class and an inherited **daugther** class. Both of them should have a method void **display ()** that prints a message (different for mother and daugther). In the main define a daughter and call the **display()** method of **mother** class.

Question 1 (modified): Solution

```
class mother{
public:
        void display ()
        {
                cout << "mother: display function\n";
        }
};

class daughter : public mother{
public:
        void display ()
        {
                cout << "daughter: display function\n\n";
        }
};

int main ()
{
        daughter hina;
        hina.mother::display();
        return 0;
}
```

Question 2

Develop a class **Counter** that represents a simple **integer counter**. The class should satisfy the following requirements:

a) A constructor should be provided that takes a single int argument that is used to initialize the counter value. The argument should default to zero.
b) The **prefix increment** and **postfix increment** operators should be overloaded in order to provide a means by which to increment the counter value.
c) A member function **getValue** should be provided that returns the current counter value.

In addition, the class **must track how many Counter objects are currently in existence**. A means for querying this count should be provided. The code must not use any global variables.

Question 2: Solution

```
class Counter {
public:
 Counter(int initialValue = 0) {
        value= initialValue
        ++numCounters; }

~Counter() {
        --numCounters;
    }

Counter& operator++() ;
Counter operator++(int) ;

    int getValue() const ;
    static int getNumCounters() ;
    }

private:
    int value; Counter objects
    static int numCounters;
};

int Counter::numCounters = 0;
```

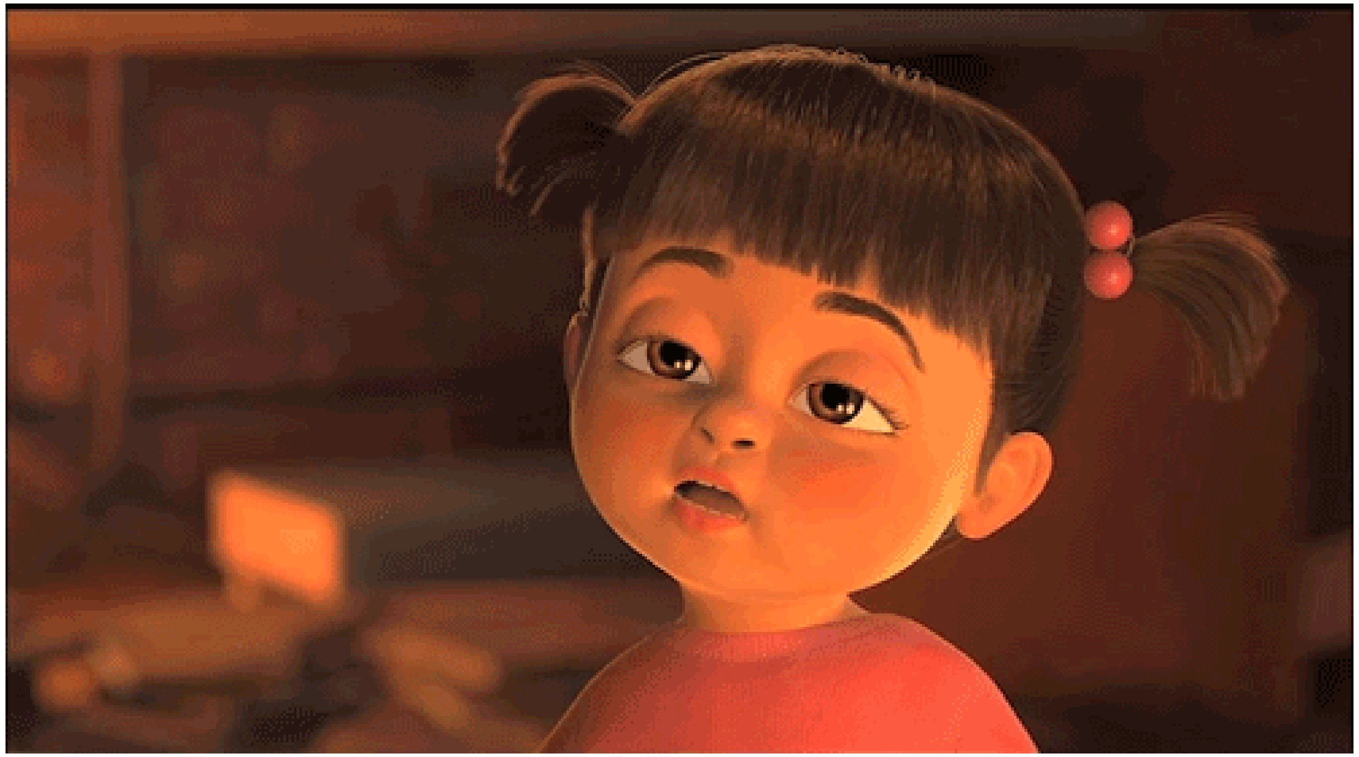
```
Counter& Counter::operator++() {
        ++value;
        return *this;
}


Counter Counter::operator++(int) {
        Counter temp = *this;
        ++value;
        return temp;
    }


int Counter::getValue() const {
        return value;
    }


static int Counter::getNumCounters() {
    return numCounters;
  }
```

Thanks a lot



If you are taking a Nap, **wake up**........Lecture Over