John Rollinson

CS151 Spring 2020

Module 5 Review Questions:

1. If a member variable is declared **STATIC**, all objects of that class share that variable.

5. A(n) **FRIEND** function is not a member of a class, but has access to the private members of the class.

6. A(n) **FORWARD DECLARATION** tells the compiler that a specific class will be declared later in the program.

11. When the **POSTFIX (++)** operator is overloaded, its function must have a dummy parameter.

15. The **class** Stuff has both a copy constructor and an overloaded **=** operator. Assume that **blob** and **clump** are both instances of the **Stuff** class. For each of the statements, indicate whether the copy constructor or the overloaded = operator will be called:

**A. Staff blob = clump ;  
B. clump = blob ;  
C. blob.operator =(clump) ;  
D. showValues(blob) ; // blob is passed by value**

Given data, Stuff has both a copy constructor and an overloaded = operator. Assume blob and clump are both instances of the Stuff class.

1. This is a copy constructor.
2. This is an overloaded = operator.
3. This is an overloaded = operator.
4. This is a copy constructor.

18. Consider the following class declaration:

**class Thing {  
 private:  
 int x ;  
 int y ;  
 static int z ;  
 public:  
 Thing()  
 { x = y = z ; }   
 static void putThing(int a)   
 { z = a ; }  
 } ;  
 int Thing:: z = 0 ;**

Assume a program containing the class declaration defines three **Thing** objects with the following statement:

**Thing one, two, three ;**

1. How many separate instances of the x member exist?

Three separate instances of the x member exist.

1. How many separate instances of the y member exist?

Three separate instances of the y member exist.

1. How many separate instances of the z member exist?

One separate instance of the z member exist.

1. What value will be stored in the x and y members of each object?

Zero will be stored in the x and y members of each object.

1. Write a statement that will call the **putThing** member function before the **Thing** objects are defined.

This statement follows that will call the putThing member function before the Thing objects are defined.

Thing::putThing(2); assigns z the value of 2.

23. Explain why the parameter of a copy constructor must be a reference.

The parameter of a copy constructor must be a reference object because if an object were passed to the copy constructor by value, the copy constructor would create a copy of the augment and store it in the parameter object. When the parameter object is created, its copy constructor will be called, causing another parameter object to be created. To prevent the copy constructor from calling itself an infinite number of times, we pass a parameter to be referencing an object.