Computer Programming :: Midterm Exam # 2

April 11, 2013 Max Points: 35

Time: 90 mins

Answer the following questions briefly and to the point.

1. Write three distinct situatididons in which copy constructor of a class is called. **[3]**

Pass by value to a function, return from a function, initialization

2. Consider the following class hierarchy and the corresponding main function. What is the output of

Class Hierarchy

```
class D {
public:
 D() { cout << "D ctor" << endl; }
 D(D&) { cout << "D copy ctor" << endl;}
 ~D() { cout << "D dtor" << endl; }
};

class A {
public:
 A() { cout << "A ctor" << endl; }
 ~A() { cout << "A dtor" << endl; }
};

class B : public A {
public:
 B() { cout << "B ctor" << endl; }
 ~B() { cout << "B dtor" << endl; }
 void test( D d ) { A a; }
};
```

Driver program

```
B globalB; //1
int main()
{
      A a; //1/2
      D d; //1/2
      D d2 = d; //1
      d = d2;
      globalB.test( d ); //2
       return 0;
}
```

```
A ctor
B ctor
A ctor
D ctor
D copy ctor
D copy ctor
A ctor
A dtor
D dtor
D dtor
D dtor
A dtor
B dtor
A dtor
```

3. Given the classes above, what will be the output if *only* the following statement is written in main. Explain your answer. **[2]**

```
D * obj;
```

I'll accept two answers:
1. No output since only a pointer is made. No object is created.
2. Assuming B globalB still remains in the scope

```
A ctor
B ctor
B dtor
A dtor
```

4. Is the piece of code below correct? If yes, what is the output of the function `Bar`. If no, why not? **[3]**

```
class Foo {
int x;
static int count;
public:
	static int Bar(int i) {
		return x*i*count + x*i + 1;
	}
};
```

Would not compile. A non-static member cannot be referenced in a static function.

5. Given the code in question 4, write code to initialize the data member `count` to `10`. Also write the code to call the function `Bar` from main. **[2]**

```
int Foo::count = 10;
Foo::Bar(0);
```

6. Following is a C++ class representing a mathematical fraction, where *n* is the numerator and *d* is the denominator. Implement the post-decrement and pre-decrement operators for this class. **[3]**

```
class Fraction {
   int n,d;
};
```

```
Fraction& Fraction::operator--() {
	n-=d;
	return *this;
}

Fraction Fraction::operator--(int) {
	Fraction f(n,d);
	n-=d;
	return f;
}
```

7. What is an initializer list? Describe two of its uses by giving examples. **[3]**

The initialization list is inserted after the constructor parameters, begins with a colon (:), and then lists each variable to initialize along with the value for that variable separated by a comma. It is executed before the body of the constructor is entered. It can be used for
1. Initializing data members: class Foo { int x; public: Foo():x{0}{} };
2. Calling base class constructors. class Child:public Parent { Child():Parent(param){} };

8. Under which access specifier are friend functions and classes defined? **[1]**

Doesn't matter

9. What is the difference between the keywords `struct` and `class`. **[1]**

struct defaults all members to public access, class defaults it to private

10. The following code on the left side lists a driver for a class `IntegerSet`. When executed, the code prints the output given on the right side.

```
1  int capacity = 10;
2  IntegerSet set1(capacity);
3  cout << "set1 = "<< set1 << endl;
4
5  set1 += 2; // Add an element to the set
6  set1 += 5;
7  cout << "set1 = "<< set1 << endl;
8
9  int arr[] = {1,2,3};
10 IntegerSet set2(arr,3);
11 cout << "set2 = "<< set2 << endl;
12
13 set2 += set1; // Union operation
14 cout << "set2 = "<< set2 << endl;
15
16 IntegerSet set3 = set2;
17 if ( set2 == set3 )
18     cout << set2 << " == " << set3 << endl;
19 else
20     cout << set2 << " != " << set3 << endl;
21
22 set2 = set3 - set1; // Set difference
23
24 if ( set2 == set3 )
25     cout << set2 << " == " << set3 << endl;
26 else
27     cout << set2 << " != " << set3 << endl;
28
29 set2 = 2 + set2; // Add an element to set
30 cout << "set2 = "<< set2 << endl;
```

```
set1 = [ ]

set1 = [ 2 5 ]

set2 = [ 1 2 3 ]

set2 = [ 1 2 3 5 ]

[ 1 2 3 5 ] == [ 1 2 3 5 ]

[ 1 3 ] != [ 1 2 3 5 ]

set2 = [ 2 1 3 ]
```

a. Define the class `IntegerSet`. **[2]**
b. Write declaration of all the functions which will allow the code above to run without any errors. You need to provide the interface only, no implementation is necessary. **[6]**
c. Provide implementation of the functions corresponding to the operations at line 22. **[4]**

```
class IntegerSet { //2
   int *set;
   int size;
   int capacity;
+
public:
   IntegerSet(int initialCap); //0.5
   IntegerSet(int arr[], int len); //0.5
   IntegerSet(IntegerSet& o); //0.5
   IntegerSet& operator=(const IntegerSet& o); //0.5
   IntegerSet& operator+= (int i); //0.5
   IntegerSet& operator+= (const IntegerSet& other); //0.5
   IntegerSet operator- (const IntegerSet& o); //0.5
   bool operator== (const IntegerSet& other); //0.5
   friend ostream& operator<<(ostream& out, const IntegerSet& aset); //1
   friend IntegerSet operator+(int a, const IntegerSet& aset); //1
};

   IntegerSet IntegerSet::operator- (const IntegerSet& o) //2
   {
         IntegerSet ret(size);
         for (int i = 0; i < size; i++) {
               bool notFound = true;
               for (int j=0; j< o.size; j++) {
                     if (set[i]==o.set[j]) {
                           notFound = false;
                           break;
                     }
               }
               if (notFound) {
                     ret.set[ret.size++] = set[i];
               }
         }

         return ret;
   }

   IntegerSet& IntegerSet::operator=(const IntegerSet& o) //2
   {
          if (this==&o) return o;
         delete set;
         size = o.size;
         capacity = o.capacity;
         set = new int[size];
         for (int i =0; i < size; i++)
         {
               set[i] = o.set[i];
         }
         return *this;
   }
```