CS 180

Exam Review

Generics

```
public class Box<E> {
   E data;
   public Box( E data ) {
      this.data = data;
   }
   public E getData() {
      return data;
   }
}

Box<Integer> intBox = new Box<Integer>( 42 )
Box<String> intBox = new Box<String>( “Hi” )

What will be the 'type' of 'data' for each of
the above declarations?
```

Inheriting From the Superclass

```
public class A {
     private int x;

     public A() { set(0); }
     public A(int x) { set(x); }

     public void set(int x) { this.x = x; }
     public int get() { return x; }
     private void increase() {
          x++;
     }
}

public class B extends A {
     public B() { super(); }
     public B(int x) { super(x); }

     @Override
     public void increase() {
          set(get()+2);
     }
}
```

```
public class A {
     protected int x;

     public A() { set(0); }
     public A(int x) { set(x); }

     public void set(int x) { this.x = x; }
     public int get() { return x; }
     private void increase() {
          x++;
     }
}

public class B extends A {
     public B() { super(); }
     public B(int x) { super(x); }

     @Override
     public void increase() {
          x += 2;
     }
}
```

Inheriting From the Superclass

On the superclass constructor calls, constructors wind up being executed top-down on the hierarchy.

```
public class A {
     public A() { SOP("A"); }
}

public class B extends A {
    public B() { SOP("B"); }
}

public class C extends A {
    public C() { SOP("C"); }
}

public class D extends B {
    public D() { SOP("D"); }
}
```

```
A a = new A();
Output: A

B b = new B();
Output: AB

C c = new C();
Output: AC

D d = new D();
Output: ABD
```

Recursion

```
public int fib(int n) {
    if (n == 0)
        return 0;
    if (n == 1)
        return 1;
    else
        return fib(n - 1) + fib(n - 2);
}

public static void main(String args[]) {
    Exam2Review e = new Exam2Review();
    for (int i = 0; i < 10; i++) {
        System.out.print(e.fib(i) + "  ");
    }
}

//Very bad way of computing fibonacci numbers. Why?
```

Recursion

```
public int bs(int arr[], int start, int end, int val) {
    int mid = (start + end) / 2;
    if (arr[mid] == val) {
        return mid;
    } else {
        if (start >= end) {
            return -1;
        }
        if (arr[mid] > val) {
            end = mid - 1;
        } else {
            start = mid + 1;
        }
        return bs(arr, start, end, val);
    }
}
```

What does this code do?

Part I. Multiple Choice Questions (2 points each):

1. Use the code segment below for the first three questions:

```
public class Exam
{
   private int x;

   public Exam() {x = 0;}
   public Exam(int x) {setX(x);}
   public Exam(Exam e) {x = e.getX();}
   public int getX() {return x + 10;}
   public void setX(int x) {this.x = x;}
   public void modifyX(Exam e) {this.x = e.getX() + 10;}

   public static void main(String[] args) {
      Exam e1 = new Exam();
      Exam e2 = new Exam(e1);
      e2.modifyX(e1);
      System.out.println(e1.getX() + "::" + e2.getX());
   }
}
```

   What is the output when the given program is run?

   (a) `20::10`
   (b) `10::20`
   (c) `10::30`
   (d) `30::10`

2. Refer to the code in question 1, what is the output if `x` is declared as `'static'`?

   (a) No change
   (b) `10::10`
   (c) `40::40`
   (d) `30::30`

3. Refer to the code in question 1, what is the output if the method `modifyX` is declared as `'static'`?

   (a) `10::10`
   (b) There is no output due to compile time error
   (c) `30::30`
   (d) `40::40`

8. What is a valid data type for the following set of data?

```
{ null, {"1"}, {"1", "2"}, {"1", "2", "6"}, {"1", "2", "6", "24"} }
```

   (a) String[]
   (b) String[][]
   (c) int[][]
   (d) int[]

9. Which of the following is FALSE about the "`throws`" clause?

   (a) If a method throws an exception that is not caught within the method, the method ends immediately after the exception is thrown.
   (b) A throws clause in an overriding method can declare to throw an exception different from that which is thrown by the corresponding method in the super-class.
   (c) A method can declare to throw an exception even if that exception is never explicitly thrown from the method.
   (d) A method can declare to throw multiple types of exceptions.

10. What is the output of the following program?

```
public class Parent
{
   private void f()
   {
      System.out.print("parent f()");
   }
   public static void main(String[] args)
   {
      Parent p = new Derived();
      p.f();
   }
}

class Derived extends Parent
{
   public void f()
   {
      System.out.print("derived f()");
   }
}
```

   (a) `parent f() derived f()`
   (b) `parent f()`
   (c) `derived f()`
   (d) `derived f() parent f()`