CSCI 201 Fall 2007: Sample Questions for Exam 3

Problem 1: One-dimensional arrays

Be able to:

1) Declare an array,
2) Initialize an array to the default values,
3) Set the elements of an array to interesting values within a loop.

In particular, review the array "modifier" loops within the Array lab and the array creation and modification code of Homework 4. Except for the "swap" and "relaxation" modifications of the Array lab, you should be able to easily recreate the Java used in the lab and homework assignment.

Be able to:

1) Perform an interesting operation on all the elements of an array.

Review the array "operator" loops within the Array lab. Again, you should be able to recreate similar Java code with little problem.

Multiple-dimensional arrays

Be able to answer questions such as the following

Question MDA1

Initialize a 3 by 4 two-dimensional array so that it contains the values stored in the following table:

201	17	30	97
-5	0	0	0
191	192	2007	42

Answer MDA1

```
double[][] MDA1Exp = {
          {  201,   17,   30,   97},
          {   -5,    0,    0,    0},
          {  191,  192, 2007,   42}
} ;
```

Question MDA2

Create a 9 by 17 two-dimensional array called `MDA2Exp` and initialize it so that `MDA2Exp[i][j]` is `3*i + 5`.

Answer MDA2

```
double[][] MDA2Exp = new double[9][] ;
for (int i=0; i<MDA2Exp.length; ++i) {
     MDA2[i] = new double[17] ;
     for (int j= 0; j<MDA2Exp[i]; ++j) {
          MDA2[i][j] = 3*i + 5 ;
     }
}
```

Problem 2: Calling Methods

Below is the documentation for 7 methods taken from various classes in the Java Standard Library. Immediately below each method's documentation, write a segment of code to call that method. If the method must be called from an object, create the necessary object before calling the method. If the method returns a value, store that value in a variable of the appropriate type. If the method has parameters, provide values of the appropriate type for those parameters.

From the **Math** class:

static double	**max**(double a, double b) Returns the greater of two double values.

Answer: `double m = Math.max(5.5, 13.6) ;`

static double	**random**() Returns a double value with a positive sign, greater than or equal to 0.0 and less than 1.0.

Answer: `double r = Math.random() ;`

static long	**round**(double a) Returns the closest long to the argument.

Answer: `double n = Math.round(20.1) ;`

From the **String** class:

String	**concat**(String str) Concatenates the specified string to the end of this string.

Answer:
String letters1 = new String("hello");
String letters2 = new String("bye");
String letters3 = letters1.concat(letters2);

boolean	**equalsIgnoreCase**(String anotherString) Compares this String to another String, ignoring case considerations.

Answer:
String letters1 = new String("hello");
String letters2 = new String("bye");
boolean theTruth = letters1.equalsIgnoreCase(letters2);

int	**indexOf**(int ch) Returns the index within this string of the first occurrence of the specified character.

Answer:
String letters1 = new String("hello");
int index = letters1.indexOf(99);

From the **Random** class:

int	**nextInt**(int n) Returns a pseudorandom, uniformly distributed int value between 0 (inclusive) and the specified value (exclusive), drawn from this random number generator's sequence.

Answer:
Random ran = new Random();
int number = ran.nextInt(99);

Problem 3: Classes

Below are 7 problems each containing a program that defines a class in addition to the class containing the method main(). Answer the question associated with each problem below.

Problem a.
What will be output by the program below?

```
public class Dog {
      private String name;

      public Dog (String dogName) {
         name = dogName;
      }

      public String getName() {
         return name;
      }

      public void setName(String nombre) {
         name = nombre;
      }
}

public class TestQuestion {
      public static void main(String [] args) {
         Dog dawg1 = new Dog (”fido”);
         Dog dawg2 = new Dog (”spike”);
         System.out.println (”dawg1’s name is ” + dawg1.getName()
            + ” dawg2’s name is ” + dawg2.getName());
      }
}
```

Problem b.
What will be output by the program below?

```
public class Cat  {
  private int weight;

  public Cat () {
    weight = 10;
  }

  public Cat (int weight) {
    this.weight = weight;
  }

  public void setWeight (int weight) {
    this.weight = weight;
  }

  public int getWeight () {
    return weight;
  }
}

public class TestQuestion {
      public static void main(String[] args) {
            int catWeight = 25;
            Cat tom = new Cat(catWeight);
            System.out.println(”the value of catWeight is ”
                  + catWeight );
            tom.setWeight(0);
            System.out.println(”the cat’s weight is ”
                  + tom.getWeight() );
      }

}
```

Problem c.
Is the program below legal, and if not, why not?

```
public class Cat  {
  private int weight;

  public Cat () {
    weight = 10;
  }

  public Cat (int weight) {
    this.weight = weight;
  }

  public void setWeight (int weight) {
    this.weight = weight;
  }
```

```
  public int getWeight () {
    return weight;
  }
}

public class TestQuestion {
      public static void main(String[] args) {
            int catWeight = 25;
            Cat tom = new Cat(catWeight);
            tom.weight = 0;
            System.out.println(”the value of catWeight is ”
                  + catWeight );
      }
}
```

Problem d.

What will be output by the program below?

```
public class PositiveInteger {
      private int posInteger;

      public PositiveInteger(int number) {
            setInteger(number);
      }

      public int getInteger() {
         return posInteger;
      }
      public void setInteger(int number) {
         posInteger = Math.abs(number);
      }
}

public class TestQuestion {
      public static void main(String [] args) {
         int num = 10;
         PositiveInteger posNum = new PositiveInteger(-1);
         System.out.println(”my integer = ” + num +
            ” and my positive integer = ” + posNum.getInteger());
      }
}
```

Problems e through g all refer to the Star class defined below.

```
public class Star {
    public double x ;
    public double y ;
    public int points ;
    public double radius ;
    public double rotation ;

    public Star() {
        x = 1.0 ;
        y = 1.0 ;
        points = 5 ;
        radius = 1.0 ;
        rotation = 0.0 ;
    }

    public Star(double myX, double myY) {
        x = myX ;
        y = myY ;
        points = 5 ;
        radius = 1.0 ;
        rotation = 0.0 ;
    }

    public Star(double myX, double myY, int myPoints) {
        x = myX ;
        y = myY ;
        points = myPoints ;
        radius = 1.0 ;
        rotation = 0.0 ;
    }

    public Star(double myX, double myY, int myPoints,
            double myRadius, double myRotation) {
        x = myX ;
        y = myY ;
        points = myPoints ;
        radius = myRadius ;
        rotation = myRotation ;
    }
}
```

Problem e.
What will be output by the program below assuming it also contains the star class.

```
   public class TestQuestion {
      public static void main(String [] args) {
         Star bright = new Star();
         System.out.println(”bright star has ” + bright.points + ” points”);
      }
   }
```

Problem f.
What will be output by the program below?

```
public class TestQuestion {
   public static void main(String [] args) {
      Star bright = new Star(1.0,1.0);
      System.out.println(”bright star has ” + bright.points + ” points”);
   }
}
```

Problem g.
What will be output by the program below?

```
public class TestQuestion {
   public static void main(String [] args) {
      Star bright = new Star(1.0, 1.0, 4);
      System.out.println(”bright star has ” + bright.points + ” points”);
   }
}
```

Problem 3: Reference Variables

Below are 6 problems each containing a program that generates output. For each problem, state what would be printed if the code were run.

Problem a.

```
public class TestQuestion {
   public static void main(String [] args) {
      int[] numbers = {1,2,3,4};
      int[] moreNumbers = {5,6,7,8,9,10};
      numbers = moreNumbers;
      for(int i=0; i<number.length; i++) {
         System.out.println (”number at index ” + i + ” = ” + numbers[i]);
      }
   }
}
```

Problem b.

```
   public class TestQuestion {
      public static void main(String [] args) {
         int[] numbers = {1,2,3,4};
         getNumbers(numbers);
         for(int i=0; i<number.length; i++) {
            System.out.println (”number at index ” + i + ” = ” + numbers[i]);
         }
      }

      public static void getNumbers(int[] array) {
         for(int i=0; i<array.length; i++) {
            array[i]=i;
         }
      }
   }
```

Problem c.

```
   public class TestQuestion {
      public static void main(String [] args) {
         int[] numbers = {1,2,3,4};
         int[] moreNumbers = {5,6,7,8,9,10};
         int[] temp = numbers;
         numbers = moreNumbers;
         moreNumbers = temp;
         for(int i=0; i<number.length; i++) {
            System.out.println (”number at index ” + i + ” = ” + numbers[i]);
         }
      }
   }
```

Problem d.

```
public class Dog {
      private String name;

      public Dog (String dogName) {
         name = dogName;
      }

      public String getName() {
         return name;
      }

      public void setName(String nombre) {
         name = nombre;
      }
}
```

```
public class TestQuestion {
      public static void main(String [] args) {
         Dog dawg1 = new Dog (”fido”);
         Dog dawg2 = new Dog (”spike”);
         dawg1 = dawg2;
         System.out.println (”dawg1’s name is ” + dawg1.getName()
            + ” dawg2’s name is ” + dawg2.getName());
      }
}
```

Problem e.

```
public class Dog {
      private String name;

      public Dog (String dogName) {
         name = dogName;
      }

      public String getName() {
         return name;
      }

      public void setName(String nombre) {
         name = nombre;
      }
}

public class TestQuestion {
      public static void main(String [] args) {
         Dog dawg1 = new Dog (”fido”);
         Dog dawg2 = new Dog (”spike”);
         Dog temp = dawg1;
         dawg1 = dawg2;
         dawg2 = temp;
         System.out.println (”dawg1’s name is ” + dawg1.getName()
            + ” dawg2’s name is ” + dawg2.getName());
      }
}
```

Problem f.

```
public class Dog {
      private String name;

      public Dog (String dogName) {
         name = dogName;
      }

      public String getName() {
         return name;
      }
```

```
      public void setName(String nombre) {
         name = nombre;
      }
}

public class TestQuestion {
      public static void main(String [] args) {
         Dog dawg1 = new Dog (”fido”);
         Dog dawg2 = new Dog (”spike”);
         Dog temp = dawg1;
         temp.setName(”badDog”);
         System.out.println (”dawg1’s name is ” + dawg1.getName()
            + ” dawg2’s name is ” + dawg2.getName());
      }
}
```