CSCI 1933 Lab 5

Midterm 1 Review

Rules

This lab is to be completed without use of a computer. **You must use only a pen and paper to complete these steps.**

This lab is to be done individually. Due to the midterm this week, there are no sections to be checked off for this lab. You will get full credit for attending the lab. No portion of the lab may be checked off in office hours.

1 Class Design

In this step you must write the methods to complete a `Train` class. Assume that the Passenger class has already been written.

- Each train has a capacity. A default constructor should initialize the capacity to 100, and another constructor should allow the programmer to specify any positive capacity.
- An `addPassenger(Passenger p)` method should add the given `Passenger` object to the train. Passengers can't be added if the train is full.

  > **Hint:** What new member variable is needed?

- A `numberOfPassengers()` method should return the current number of `Passengers` on board the train.
- An `availableCapacity()` method should return the number of available `Passengers` the train can hold.

2 Cloning an Array

A common operation on data in Java is "cloning". A clone is an identical but separate copy of an object. Note that clones will yield `true` when compared to each other with the `Arrays.equals()` method, but will yield `false` when compared with `==`.

Write a method `public static int[] clone(int[] inputArray)` that will return "clone" of the `inputArray`, and explain to the TA why your method satisfies the properties of cloning listed above.

3 Debugging

The following snippets of code all have at least one bug. Write down fixed code, and explain to a TA what the bugs are.

```
// Constructs a "Whatever" object
private int data;
public Whatever(double data) {
        data = data;
}
public static void setData(int newData) {
        data = newData;
}
```

```
// Computes the sum of the numbers the user enters
Scanner s = new Scanner(System.in);
int sum = 0;
while (s.hasNext()) {
        String data = s.next();
        if (data == "stop") {
                break;
        }
        sum += data;
}
System.out.println("The sum is:" + sum);
```

```
// Cross out the line with the error.
// What are the values of b1, b2, b3, and b4?
Object a = new Object();
Object b = new Object();
Object c = null;
boolean b1 = a.equals(b);
boolean b2 = b.equals(c);
boolean b3 = c.equals(a);
boolean b4 = (a == null || c == null) || c.equals(b);
```

4 Code Comprehension

Give the output (`System.out`) of the following `main` method.

```
public class Whatever {
	private int data;

	public Whatever(int newData) {
		data = newData;
	}

	public void setData(int newData) {
		data = newData;
	}

	public int getData() {
		return data;
	}

	public static void doWhatever(Whatever w, int i, int d) {
		System.out.println("doWhatever(1): w: "+w.getData()+", i: "+i+"; d:
		    "+d);
		w.setData(i);
		d = i;
		i = d;
		System.out.println("doWhatever(2): w: "+w.getData()+", i: "+i+"; d:
		    "+d);
	}

	public static void main(String[] args) {
		Whatever w = new Whatever(1);
		int i = 2;
		double d = 3;
		System.out.println("main(1): w: "+w.getData()+", i: "+i+"; d: "+d);
		doWhatever(w, i, (int)d);
		System.out.println("main(2): w: "+w.getData()+", i: "+i+"; d: "+d);
		w = new Whatever(i);
		d = i / 4;
		System.out.println("main(3): w: "+w.getData()+", i: "+i+"; d: "+d);
	}
```

```
}
```

5 Recursive Algorithms

Write a **recursive** method `public static int productDigits(int n)` to return the product of the digits in the integer `n`. For example, `productDigits(1552)` is 50.

6 Building a 2-D Array

Write a method `public static double[][] make2DArray(int row, int column)` to return an array of dimension $row \times column$ of doubles such that the values in each row start with the row index and go up by 1 for each column. For example, if $row = 4$ and $col = 3$, the array returned will look like this:

0.0	1.0	2.0
1.0	2.0	3.0
2.0	3.0	4.0
3.0	4.0	5.0

7 Least Common Multiple

Write a method `public static int leastCommonMultiple(int a, int b)` to return the least common multiple of two integers. The least common multiple of two positive integers a and b is defined as the smallest positive number that is evenly divisible by both a and b.

8 Polynomials

In this step you will write a class to represent a polynomial of the form:

$$ax^2 + bx + c$$

The class should be called `Polynomial`. It should have the following constructor and methods:

- `public Polynomial(double a, double b, double c)` – the constructor
- `public double getA()` – returns the value of `a`
- `public double getB()` – returns the value of `b`
- `public double getC()` – returns the value of `c`
- `public void add(Polynomial p)` – adds `p` + the current polynomial. To add polynomials, add coefficients of like terms.
- `public double evaluate(double x)` - returns the value of this polynomial evaluated at `x`