This handout consists of most of the questions from the second test in CS 124 from Fall 2014, with a couple questions added. It is not meant as comprehensive review for our test, but it can give you some idea of the types of questions and topics that you can expect.

1. Some short essay-type questions...

a) The modifier "final" is used to make variables into named constants. Give two reasons for using named constants (as opposed to literal constant values).

b) Suppose that *Foo* is the name of a class and that *p* and *q* are variables of type *Foo*. Carefully explain what is being tested and what is not being tested in a comparison using the operator "==", such as: `if ( p == q )`

c) What are "getters and setters" in the context of object-oriented programming, and why are they used?

2. Write a definition for a static subroutine that satisfies the following description: The name of the subroutine is *almostEqual*. It has two parameters of type *double*, and it returns a *boolean* value. The return value is *true* if the absolute value of the difference of the parameters is less than 0.000005 and otherwise is *false*. (Absolute value can be computed with the function *Math.abs*.)

3. Write a static void subroutine with one parameter of type String. The subroutine should output a count of the number of times that each letter of the alphabet occurs in the string, formatted similarly to the sample output shown here. (Recall that you can use a variable of type *char* in a for loop.)

```
A: 4
B: 0
C: 1
D: 2
E: 11
F: 0
 .
 .
 .
Z: 1
```

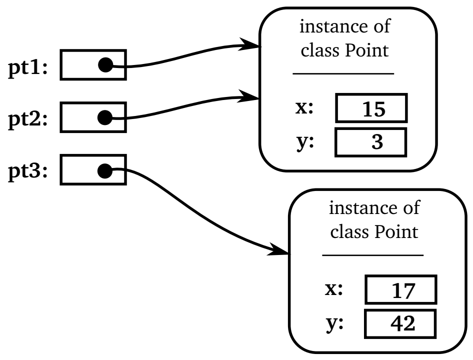
4. Write a static subroutine with one parameter of type *double*`[]`. The subroutine will find and return the average of all the positive numbers (greater than zero) in the array. Ignore numbers that are less than or equal to zero. Note that you will have to count the positive numbers as well as add them up.

5. Suppose that *Foo* is the name of a class. It is possible that both of the following lines of code are syntactically correct. Explain the difference.

(a) x = new Foo[10]; **(b)** y = new Foo(10);

6. The picture below shows three variables (pt1, pt2, and pt3) pointing to two objects. The variables are of type *Point*, where *Point* is the class shown on the left. Write a code segment that creates the situation shown in the picture. You need to declare the variables, create objects containing data as shown, and give pt1, pt2, and pt3 the values shown.

```
public class Point {
    int x;
    int y;
}
```



7. For this problem, you should write a complete Java class definition. The name of the class should be *RandMaker*. An object of type *RandMaker* represents a source of random integers in the range from 1 up to some maximum value. The maximum value is an **instance variable** of the object. The class has a **constructor** that specifies the maximum value. It has a **method** named *make* that returns a random number. For example, the class could be used like this:

```
RandMaker rnd; // A RandMaker to make numbers in the range 1..6
rnd = new RandMaker(6);
int d1 = rnd.make(); // get a random number
int d2 = rnd.make(); // get another random number
```

Write a class that meets these specifications and that could be used as shown.

8. Suppose that a class *InventoryItem* is defined as shown on the left below. Suppose that the variable *inventory* is defined as shown on the right below. And suppose that the *inventory* array has already been filled with 100 objects containing information about 100 inventory items. Write a code segment that will print the *name* of every inventory item for which the *numberInStock* is zero.

```
public class InventoryItem {
    String name;
    int itemID;
    int numberInStock;
}
```

```
InventoryItem[] inventory;
inventory = new InventoryItem[100];
```

9. Explain what it means for one class to be a *subclass* of another class.

10. A subroutine is a black box, but it can interact with the rest of the program through *parameters*, *return values*, and *global variables*. Explain these terms and explain how parameters, return values, and global variables are used for communication between subroutine and program.