1 Our First Java Program

Below is our first Java program of the semester. Next to each line, write out what you think the code will do when run. *This exercise is adapted from Head First Java.*

```java
1 int size = 27; // Declares a variable of type int and assigns it the value 27. In Java, all variables must be declared before they are used
2 String name = "Fido"; // Declares a variable of type String and assigns it the variable "Fido"
3 Dog myDog = new Dog(name, size); // Declares and initializes a new variable of type Dog. Calls the Dog constructor to create a new object of type Dog
4 int x = size - 5; // Declares a new variable of type int and assigns it the value 22
5 if (x < 15) { // If x is less than 15, calls the bark method on an instance of the Dog class. Since x is 22, myDog.bark is not called
6     myDog.bark(8);
7 }
8 while (x > 3) { // Checks if x is greater than 3 and if so calls myDog's play method. Subtracts 1, and as long as x is bigger than 3, goes back to the beginning of the loop. Play happens a total of 19 times.
9     x -= 1;
10    myDog.play();
11 }
12 int[] numList = {2, 4, 6, 8}; // Declares an array of ints and initializes it to {2, 4, 6, 8}
13 System.out.print("Hello "); // Prints the String "Hello " to the standard output
14 System.out.println("Dog: "+ name); // Prints the String "Dog: Fido" to the standard output and then terminates the line
15 System.out.println(numList[1]); // Prints the String "4" to the standard output and then terminates the line. In Java, arrays are indexed from 0
16 if (numList[3] == 8) { // numList[3] is equal to 8
17     System.out.println("potato"); // Prints the String "potato" to the standard output and then terminates the line
18 }
```

2 Mystery

This is a function (a.k.a. method). It takes an array of integers and an integer as arguments, and returns an integer.

```java
public static int mystery(int[] inputArray, int k) {
    int x = inputArray[k];
    int answer = k;
    int index = k + 1;
    while (index < inputArray.length) {
        if (inputArray[index] < x) {
```
 Describe in English what mystery returns if inputArray = [3, 0, 4, 6, 3] and k = 2.

It returns the index of the smallest element that occurs at or after index k in the array, in this case, 4. If k is greater than or equal to the length of the array or less than 0, an ArrayIndexOutOfBoundsException will be thrown, though this exception is not something you’d know without running the program.

The variable x keeps track of the smallest element found so far and the variable answer keeps track of the index of this element. The variable index keeps track of the current position in the array. The while loop steps through the elements of the array starting from index k + 1 and if the current element is less than x, x and answer are updated.
Extra: This is another function. It takes an array of integers and returns nothing.

```java
public static void mystery2(int[] inputArray) {
    int index = 0;
    while (index < inputArray.length) {
        int targetIndex = mystery(inputArray, index);
        int temp = inputArray[targetIndex];
        inputArray[targetIndex] = inputArray[index];
        inputArray[index] = temp;
        index = index + 1;
    }
}
```

Describe what `mystery2` does if `inputArray = [3, 0, 4, 6, 3].`

If `mystery2` is called on the array `3, 0, 4, 6, 3` then after the method runs, the array will be `0, 3, 3, 4, 6`. Given any array, the method `mystery2` sorts the elements of the array in increasing order.

At the beginning of each iteration of the while loop, the first `index` elements of the array are in sorted order. Then the method `mystery` is called to find the index of the smallest element of the array occurring at or after `index`. The element at the index returned by `mystery` is then swapped with the element at position `index` so that the first `index + 1` elements of the array are in sorted order.

This algorithm is called selection sort. We will talk about it more later on in the course.

3 Writing Your First Program

Implement `fib` which takes in an integer `n` and returns the `n`th Fibonacci number.

The Fibonacci sequence is `0, 1, 1, 2, 3, 5, 8, 13, 21, ...`

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

Extra: Implement `fib` in 5 lines or fewer. Your answer must be efficient.

```java
public static int fib2(int n, int k, int f0, int f1) {
    if (n == k) {
        return f0;
    } else {
        return fib2(n, k + 1, f1, f0 + f1);
    }
}
```