CS 61B Summer 2021 — Small Group Tutoring, Section 2: Loops and Arrays — Worksheet 2

1 Read Me

Describe what each of the following methods does. You may assume that `values` contains at least one element.

Boolean - probably a check

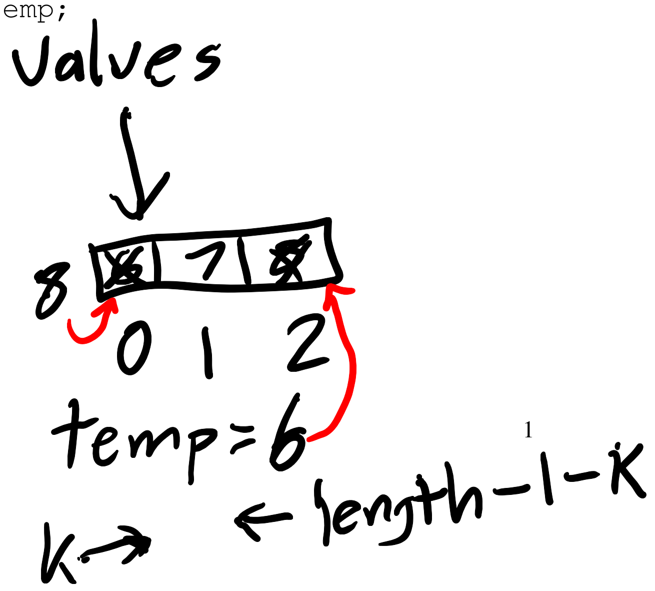
```
private static boolean method1 (int[] values) {
    int k = 0;                            length - 1 prevents an index out of bound
    while (k < values.length - 1) {
        if (values[k] > values[k+1]) { Compares current el with next one
            return false;
        }
        k = k + 1; Increment - cycle
    }              through the array
    return true;
}
```

The method returns true when each element is smaller or equal to the element after it.

Not returning anything

```
private static void method2 (int[] values) {
    int k = 0;
    while (k < values.length / 2) { iterate over the first half of indices
        int temp = values[k]; store the value at k
        values[k] = values[values.length - 1 - k]; going in reverse
        values[values.length - 1 - k] = temp;
        k = k + 1;
    }
}
```

The method reverses values in place.



2 Flatten

Write a method flatten that takes in a 2-D int array x and returns a 1-D int array that contains all of the arrays in x concatenated together. For example, `flatten([[1, 3, 7], [], [9]])` should return `[1, 3, 7, 9]`.

```
public static int[] flatten(int[][] x) {
    int newArraySize = 0;
    for (int I = 0; I < x.length; I ++) {
    newArraySize += x[I].length;
    }
    int[] newArray = new int[newArraySize];
    int newArrayIndex = 0;
    for (int I = 0; I < x.length; I++) {
        for (int j = 0; j < x[I].length; j++) {
newArray[newArrayIndex] = x[I][j];
        newArrayIndex += 1;
        }
    }
    return newArray
}
```

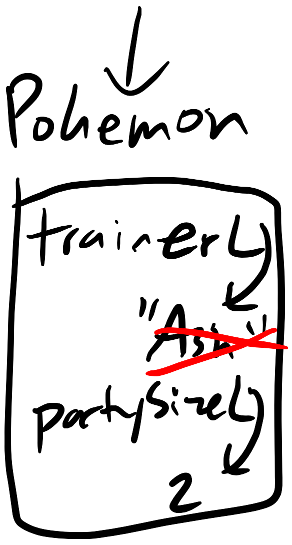
3 Bugged Out

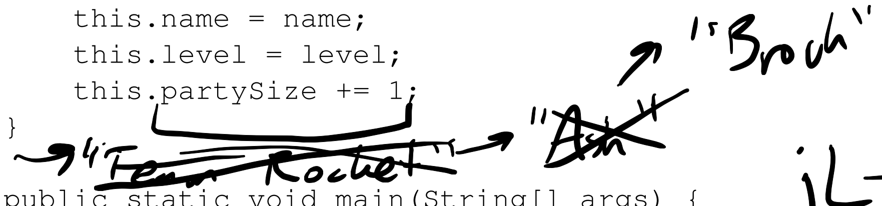
We have a class arrFunctions, and we decide that we want to write it a method with the following signature: `public static int arr_multiply(int[] arr)`. This method takes in an `int[]` and returns all the **non-zero** values in the array multiplied together. If there is a zero in the array, we want to ignore it. The only time we should return 0 is if the array is empty. We want our array to work in all sorts of odd edge cases without any errors.

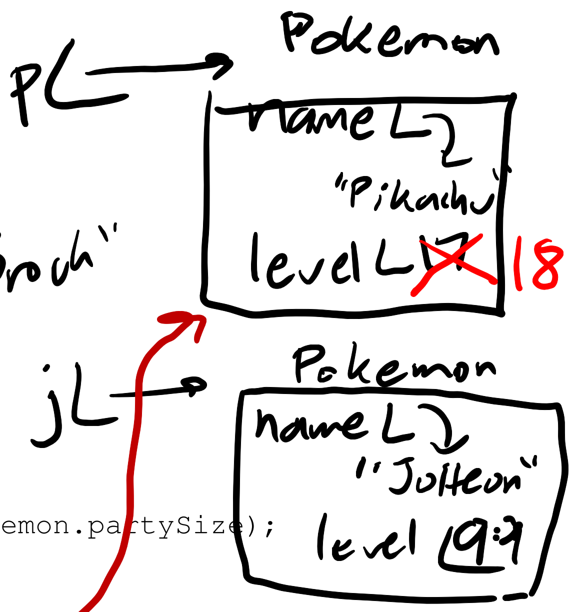
Write 3 unit tests that each target cases for this method. You do not need to write the method, just the tests (don't you love test driven development?!).

```
@Test
public void testRegular() {
    int[] arr = new int[]{1, 2, 3};
    int result = arrFunctions.arr_multiply(arr);
    assertEquals(6, result);
}

@Test
public void testZero() {
    int[] arr = new int[]{0, 0, 0};  // new int[]{0, 1, 2};
    int result = arrFunctions.arr_multiply(arr);
    assertEquals(0, result);
}

@Test
public void testEmpty {
    int[] arr = new int[]{};
    int result = arrFunctions.arr_multiply(arr);
    assertEquals(0, result);
}
```

4 Extra: Static Electricity







```java
public class Pokemon {
    public String name;
    public int level;
    public static String trainer = "Ash";
    public static int partySize = 0;

    public Pokemon(String name, int level) {
        this.name = name;
        this.level = level;
        this.partySize += 1;
    }

    public static void main(String[] args) {
        Pokemon p = new Pokemon("Pikachu", 17);
        Pokemon j = new Pokemon("Jolteon", 99);
        System.out.println("Party size: " + Pokemon.partySize);
        p.printStats();
        int level = 18;
        Pokemon.change(p, level);
        p.printStats();
        Pokemon.trainer = "Ash";
        j.trainer = "Brock";
        p.printStats();
    }

    public static void change(Pokemon poke, int level) {
        poke.level = level;
        level = 50;
        poke = new Pokemon("Voltorb", 1);
        poke.trainer = "Team Rocket";
    }

    public void printStats() {
        System.out.println(name + " " + level + " " + trainer);
    }

}
```

a) Write what would be printed after the main method is executed.

Party Size: 2
Pikachu 17 Ash
Pikachu 18 Team Rocket
Pikachu 18 Brock

b) On line 28, we set `level` equal to `50`. What `level` do we mean? An instance variable of the `Pokemon` class? The local variable containing the parameter to the `change` method? The local variable in the `main` method? Something else?

The parameter

c) If we were to call `Pokemon.printStats()` at the end of our main method, what would happen?

Errors - can't call instance method on the class (needs to be called on an instance)