CS 61B Discussion 3: {O,b,j,e,c,t,s} Fall 2019

1 Objects Refresher

Answer the following questions about the `Avatar` class.

```
1  public class Avatar {
2      public static String electricity; public String fluid;
3
4      public Avatar(String str1, String str2) {
5          Avatar.electricity = str1;
6          this.fluid = str2;
7      }
8
9      public static void main(String[] args) {
10         Avatar foo1 = new Avatar("one ", "two");
11         Avatar foo2 = new Avatar("three ", "four");
12         System.out.println(foo1.electricity + foo1.fluid);
13         foo1.electricity = "I declare ";
14         foo1.fluid = "a thumb war";
15         System.out.println(foo2.electricity + foo2.fluid);
16     }
17 }
```

(a) Determine what would be printed after executing the main method of class `Avatar`.

The main method will print the following: `three two`
`I declare four`

(b) If we changed only line 2 such that `electricity` is an instance variable and `fluid` is a class variable instead, would this code still compile or which other lines would also need to be changed and in what way?

`Avatar` on line 5 will no longer work if `electricity` was no longer static; it would cause a compile-time error because we cannot reference instance variables using a static class reference. But, `this` would still work on line 6 even if `fluid` is made static since an instance variable can be used to reference a static class reference.

(c) Reverting our changes from part (b) and starting from the original code, will adding the following method to class `Avatar` cause any errors during compilation or execution?

```
public static String getFluid() {
    return fluid;
}
```

The method will cause a compile-time error because we can not reference an instance variable (in this case, `fluid`) from inside a static context.

When the object is not specified (the thing before the period) in a field access or method call, Java will use `this.` by default. However, since the new method is static, `this` does not exist and therefore an error is thrown.

2 Min/Max

Given an array `A`, return a 2 element array `B` where `B[0]` is the minimum element of `A` and `B[1]` is the maximum element of `A`.

```
import static java.lang.Math.max; // max(a, b) returns max of a, b
import static java.lang.Math.min; // min(a, b) returns min of a, b

public static int[] minMax(int[] A) {
    int maxVal = Integer.MIN_VALUE; // smallest int in Java
    int minVal = Integer.MAX_VALUE; // largest int in Java

    int[] B = new int[2];

    for (int i = 0; i < A.length; i+= 1) {
        maxVal = max(maxVal, A[i]);
        minVal = min(minVal, A[i]);
    }
    B[0] = minVal;
    B[1] = maxVal;
    return B;

}
```

3 Reverse

Given an array `A`, reverse its elements in place (do not create any new arrays; this should be a destructive method).

```
public static void reverse(int[] A) {

    for (int i = 0; i < A.length / 2; i++) {
        int temp = A[A.length - i - 1];
        A[A.length - i - 1] = A[i];
        A[i] = temp;
    }

}
```