Discussion 1: Intro to Java

Introductions!

Hi everyone! My name is Adel and I will be your CS 61B TA this semester 🙂

Discussion: Wed 4-5pm 120 Wheeler Hall
Lab: Thurs 3-5pm 275 Soda Hall
OH: Thurs 1-2pm in 220 Jacobs Hall

email: asetoodehnia@berkeley.edu
website: asetoodehnia.github.io (or find it through the CS 61B staff webpage)

Today's Goals:

1. Familiarizing ourselves with Java Syntax

2. Feeling comfortable reading *and* writing Java code

3. Getting to know each other and making new friends/study buddies!

Today's Agenda:

1. Understand and complete
    - question 1
    - first part of question 2
    - first part of question 3

2. Extra time
    - *extra* part of question 2
    - *extra* part of question 3

Question 1: Javaian Rhapsody

Next to each line, write out what you think the code will do when it is run. Assume the `Singer` class exists and that the code below compiles.

```
1  String disagree = "no";
2  int x = 7;
3  Singer queen = new Singer("Queen");
4
5  while (x > 0) {
6      x -= 1;
7      queen.sing(disagree);
8  }
9
10 String[] phrases = {"Oh", "mamma mia", "let me go"};
11 System.out.print(phrases[0]);
12 for (int i = 0; i < 3; i += 1) {
13     System.out.print(" " + phrases[1]);
14 }
15 System.out.print(" " + phrases[2]);
```

String disagree → "no"

int x 7

Singer queen → String ? → "Queen"

↳ calling constructor of Singer obj.

↳ queen will sing "no" 7 times

String[] phrases → "Oh" | "mama mia" | "let me go"

loops 3 times

Std out:

Oh mama mia mama mia mama mia let me go

Question 2 (Part 1): Mystery

Below is a function (or method) called `mystery1`. It takes an array of integers called `inputArray` and an integer `k` as arguments and returns an integer.

```
1  public static int mystery1(int[] inputArray, int k) {
2      int x = inputArray[k];
3      int answer = k;
4      int index = k + 1;
5      while (index < inputArray.length) {
6          if (inputArray[index] < x) {
7              x = inputArray[index];
8              answer = index;
9          }
10         index = index + 1;
11     }
12     return answer;
13 }
```

int[] inputArray → | 3 | 0 | 4 | 6 | 3 | (0, 1, 2, 3, 4)

int k 2

int x 4 3

int answer 2 4

int index 5 4

Write the return value of `mystery1` if `inputArray` is the array {3, 0, 4, 6, 3} and `k` is 2. Then, describe in English what `mystery1` returns.

returns index of smallest element w/ index ≥ k

Question 3 (Part 1): Fibonacci

Implement `fib1` recursively. `fib1` takes in an integer `N` and returns an integer representing the `N`th Fibonacci number. The Fibonacci sequence is 0, 1, 1, 2, 3, 5, 8, 13, 21, . . . , where 0 is the 0th Fibonacci number.

```
public static int fib1(int N) {
        if (N <= 1) {
            return N;
        } else {
            return fib1(N-1) + fib1(N-2);
        }
}
```

Question 3 (Extra): Fibonacci

Extra: Implement `fib2` in 5 lines or fewer that avoids redundant computation. `fib2` takes in an integer `N` and helper arguments `k`, `f0`, and `f1` and returns an integer representing the `N`th Fibonacci number. If you're stuck, try implementing `fib1` iteratively and then see how you can transform your iterative approach to implement `fib2`.

```
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);
    }
}
```

↳ we avoid redundant computation by passing in intermediate values needed to compute fib2(N). i.e. no need for 2 recursive calls.

Question 2 (Extra): Mystery

Extra: Below is another function called `mystery2`. It takes an array of integers called `inputArray` as an argument and returns nothing.

```
1  public static void mystery2(int[] inputArray) {
2      int index = 0;
3      while (index < inputArray.length) {
4          int targetIndex = mystery1(inputArray, index);
5          int temp = inputArray[targetIndex];
6          inputArray[targetIndex] = inputArray[index];
7          inputArray[index] = temp;
8          index = index + 1;
9      }
10 }
```

5

int[] inputArray → | 0 | 3 | 3 | 4 | 6 |

int index: 0 1 2 3 4 5

int targetIndex: 1 4

int temp: 0 3 4

Write what `mystery2` will do if `inputArray` is the array {3, 0, 4, 6, 3}. Then, describe in English what `mystery2` does.

Sorts inputArray in increasing order.