CS 61B Discussion 1 Fall 2018

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.

```java
String disagree = "no";
int x = 7;
Singer queen = new Singer("Queen");

while (x > 0) {
    x -= 1;
    queen.sing(disagree);
}

String[] phrases = {"Oh", "mamma mia", "let me go"};
System.out.print(phrases[0]);
for (int i = 0; i < 3; i += 1) {
    System.out.print(" " + phrases[1]);
}
System.out.print(" " + phrases[2]);
```

2 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.

```java
public static int mystery1(int[] inputArray, int k) {
    int x = inputArray[k];
    int answer = k;
    int index = k + 1;
    while (index < inputArray.length) {
        if (inputArray[index] < x) {
            x = inputArray[index];
            answer = index;
        }
        index = index + 1;
    }
    return answer;
}
```

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.

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

```java
public static void mystery2(int[] inputArray) {
    int index = 0;
    while (index < inputArray.length) {
        int targetIndex = mystery1(inputArray, index);
        int temp = inputArray[targetIndex];
        inputArray[targetIndex] = inputArray[index];
        inputArray[index] = temp;
        index = index + 1;
    }
}
```

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

3 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.

```java
public static int fib1(int N) {




}
```

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`.

```java
public static int fib2(int N, int k, int f0, int f1) {




}
```