1 Javaian Rhapsody

Next to each line, write out what you think the code will do when 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 Fibonacci

Implement this function recursively.

```java
/** fib(N) returns the Nth Fibonacci number, for N>=0.
   * The Fibonacci sequence is 0, 1, 1, 2, 3, 5, 8, 13, 21, ...
   */
public static int fib(int N) {
```
Now implement a function that provides the same results but avoids redundant computations:

```java
public static int fib2(int N) {
```

Now use this modified function header to write a recursive solution that avoids redundant computations and that has a body that is 5 lines or fewer. The inputs are defined as follows: your goal is to compute the $N$th fib number, you are currently computing the $k$th fib number (which is $f_0$), and the $(k+1)$th fib number is $f_1$.

```java
public static int fib3(int N, int k, int f0, int f1) {
```
3 Mystery

```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) {
            x = inputArray[index];
            answer = index;
        }
        index = index + 1;
    }
    return answer;
}

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;
    }
}
```

- What does `mystery` return if `inputArray` is the array `{3, 0, 4, 6, 3}`, and `k` is 2?

- Describe, in English, what `mystery` returns.

- Extra for experts: What does `mystery2` return if `inputArray` is the array `{3, 0, 4, 6, 3}`? Then, describe, in plain English, what `mystery2` does to the array.