2 Mystery

```
1    /** This is a function (a.k.a. method). It takes an array
2     * of integers as an argument, and returns an integer. */
3    public static int mystery(int[] inputArray, int k) {
4        int x = inputArray[k];
5        int answer = k;
6        int index = k + 1;
7        while (index < inputArray.length) {
8            if (inputArray[index] < x) {
9                x = inputArray[index];
10               answer = index;
11           }
12           index = index + 1;
13       }
14       return answer;
15   }
16
17   /** Extra for experts. This is another function. It takes an
18    * array of integers and returns nothing at all. */
19   public static void mystery2(int[] inputArray) {
20       int index = 0;
21       while (index < inputArray.length) {
22           int targetIndex = mystery(inputArray, index);
23           int temp = inputArray[targetIndex];
24           inputArray[targetIndex] = inputArray[index];
25           inputArray[index] = temp;
26           index = index + 1;
27       }
28   }
```

- What does `mystery` return if `inputArray` is the array 3, 0, 1, 6, 3, and `k` is 2?
  It returns 2

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

  It returns the index of the smallest element that occurs at or after index `k` in the array. If `k` is greater than or equal to the length of the array or less than 0, an `ArrayIndexOutOfBoundsException` will be thrown.

  The variable `x` keeps track of the smallest element found so far and the variable `answer` keeps track of the index of this element. The variable `index` keeps track of the current position in the array. The while loop steps through the elements of the array starting from index `k` + 1 and if the current element is less than `x`, `x` and `answer` are updated.

- Extra for experts: What does `mystery2` do if `inputArray` is the array 3, 0, 1, 6, 3? Describe, in English, what `mystery2` does to the array.

  If `mystery2` is called on the array 3, 0, 1, 6, 3 then after the method runs, the array will be 0, 1, 3, 3, 6. Given any array, the method `mystery2` sorts the elements of the array in increasing order.

  At the beginning of each iteration of the while loop, the first `index` elements of the array are in sorted order. Then the method `mystery` is called to find the index of the smallest

element of the array occurring at or after `index`. The element at the index returned by `mystery` is then swapped with the element at position `index` so that the first `index + 1` elements of the array are in sorted order.

3   Writing Your First Program

```
/** 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) {
    if (N <= 0) {
        return N;
    } else {
        return fib(N - 1) + fib(N - 2);
    }
}
```

We can also write this iteratively:

```
/** 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) {
    int f0 = 0;
    int f1 = 1;
    while (N > 0) {
        int temp = f1;
        f1 = f0 + f1;
        f0 = temp;
        N -= 1;
    }
    return f0
}
```

Extra for experts: Complete `fib2` in 5 lines or less. Your answer must be efficient.

```
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);
    }
}
```

To compute the Nth fibonacci number using `fib2`, call `fib2(N, 0, 0, 1)`.