Small Group Tutorial 1, 23-27/1/2017

1. Fibonacci numbers can also be computed using the following recursive formula:

$$
\begin{array}{rcl}
fib(0) & = & 0, \\
fib(1) & = & 1, \\
fib(2) & = & 1, \\
fib(n) & = & \begin{cases} (fib((n+1)/2))^2 + (fib((n-1)/2))^2, & n \geq 3 \text{ and } n \text{ is odd}, \\ (fib((n/2)+1) + fib((n/2)-1)) * fib(n/2), & n \geq 3 \text{ and } n \text{ is even}. \end{cases}
\end{array}
$$

   (a) Using the above formula, calculate $fib(3)$, $fib(4)$, $fib(5)$, $fib(6)$, and $fib(11)$.
   Verify if the calculated value $fib(11)$ is correct by writing down the elements of the Fibonacci sequence until $fib(11)$, using the recursive formula "$fib(n) = fib(n-1) + fib(n-2)$".

   (b) Write a recursive Java method

```
public static long fib(int k) { ... }
```

   which computes Fibonacci number $fib(k)$ using the above recursive formula.

   (c) Draw the recursion tree (the recursion trace) of the computation `fib(11)`.

   (d) In the tests of the recursive and iterative methods for computing Fibonacci numbers presented in Lecture 1, slide 43, the following computational times were observed:

```
> java FibonacciTest 42
fib(42) = 267914296 [computed iteratively in 0 ms]
fib(42) = 267914296 [computed by linear recursion in 0 ms]
fib(42) = 267914296 [computed by binary recursion in 2785 ms]
```

   What do you think about the efficiency of your recursive method from Question 1b? How much time do you think this method would take to compute $fib(42)$?

2. Write a recursive method and an iterative method to determine whether a given element x occurs in the array `list` at index f or higher. The recursive method should check if x is in the array at index f (the base case), and if not, then it should recursively check if x is at index f+1 or higher.

   Compare two objects using method "`equals`," not the operator "`==`" (since we are not interested whether two objects are physically the same, but whether they represent the same data).

```
1  public class Search {
2
3    public static boolean searchRecursive (Object[] list, int f, Object x) {
4      // recursive method for checking if x is in array list at index f or higher
5      // GIVE YOUR CODE HERE
6    }
7
8    public static boolean searchIterative (Object[] list, int f, Object x) {
9      // iterative method for checking if x is in array list at index f or higher
10     // GIVE YOUR CODE HERE
11   }
12
13   public static void main(String[] args) {
14     // test the methods
15     for (int i = 0; i < args.length; i++) {
16       System.out.print(args[i] + " ");
17     }
18     System.out.println();
19
20     // check if the first argument in the command line is repeated
21     System.out.println("the 1st argument is repeated: ");
22     System.out.println(searchRecursive(args, 1, args[0]));
23     System.out.println(searchIterative(args, 1, args[0]));
24   }
25 }
```

3. Consider the following Java method:

```
1       public static int geom(int x, int n) {
2           // assume n is at least 0
3           if ( n <= 0 ) return 1;
4           else return 1 + (x * geom(x, n-1));
5       }
```

What is the number returned by the call $geom(2, 3)$?

What is the number returned by $geom(x, n)$?