Unit 10 PBL 1: `binarySearch()`

1. Write methods according to the following specifications. Test your code.

   a) Write a method `printStars()` that takes an integer parameter and prints that number of stars to the console. For example, `printStars(5)` should print `"*****"` to the console, while `printStars(12)` should print `"************"`.
   Write this method using a `while` loop. You may assume the precondition `n>=0` is true.

   b) Write the same method as above, but **using *recursion***. Again, you may assume the precondition `n>=0` is true.

   c) Write the code for method `factorial()` that uses recursion to implement:
   $F_n = n \times F_{n-1}$ where $F_0 = 1$.

   d) Write the code for method `fibonacci()` that uses recursion to implement:
   $F_n = F_{n-1} + F_{n-2}$ where $F_0 = 0$ and $F_1 = 1$.

2. Write a recursive method that takes in an integer parameter and prints to the console the binary equivalent. For example, `printBinary(10)` should display `1010`, and `printBinary(100)` should display `1100100`.
   You may assume the precondition `n>=0` is true.
   Hint: you will need both division (`/`) and modulus (`%`).

3. The code below implements an iterative version of the *binary search* algorithm.

```
public static int binarySearch(int[] arr, int target) {
   int left = 0;
   int right = arr.length - 1;
   while (left <= right) {
      int mid = (left + right) / 2;
      if (arr[mid] == target) {
         return mid;
      } else if (arr[mid] < target) {
         left = mid + 1;
      } else {
         right = mid – 1;
      }
   }
    return -1;
}
```

   Using the code segment below as a starting point, implement a recursive version of the *binary search* algorithm. Test your code.

```
public static int binarySearch(int[] arr, int target) {
    return binarySearch(arr, target, 0, arr.length-1);
}
private static int binarySearch(int[] arr, int target,
                                int low, int high)
```