D1 Binary search experiments

A. ***Experiments with Binary Search***

Perform comparison of the running time of Binary Search and naive (brute-force) linear search:

1. Create a class BLTester and add static methods for binary search and linear search.
2. For N=1000, generate an array A of N random numerical values and sort A. Then test both binary and linear search on N different search queries (also generated randomly) and measure the total time each of them took.
3. Now do the above for N=10000, 20000, 40000, 80000, 160000. (If linear search takes too long, you may use smaller numbers.), reporting the total time used by each method for each N.
4. Draw conclusion from this:
   i) How long would you estimate that both methods use when N=320000? Validate your estimate.
   ii) What does this say about the complexity of the two methods?

You may use any code that comes with the book (but not other code).

Hand in to Canvas: a) The Java file, and b) the output of your program, along with your conclusion (in a .txt or .pdf format) as separate files (not zipped).

B. Java practice

Solve the problem „H - Longest run“ in the contest Æfingarverkefni on Mooshak (mooshak.ru.is). (Need not turn in to Canvas). The description follows below.

Since Mooshak has not been fully accessible, it suffices to run the program locally on a input file of your own making. Include then both the input and the output, along with your Java program, on Canvas.

We want to find long *runs* in the input sequence. A run is a *consecutive* non-decreasing subsequence. The first element starts a run, and as long as the numbers keep increasing (or at least not decreasing), the run gets longer. Then, once we see a smaller number, a new run starts. And so it continues, the input gets broken into runs. We want to find the longest one.

Input specification

The first input line gives the number of elements, *n*. The second line contains *n* integers. Each integer is in the range 0...1000.

Output specifications

Output the length of the longest *run* in the input.

Sample input

```
6
1 2 4 3 6 8
```

Sample output

```
3
```