TLE 2018-19 P5 - Pure Functional Programming

Time limit: 1.0s **Memory limit:** 256M

C++ is known for its many features (some argue there are too many!). Some significant programming paradigms supported by C++ are procedural, imperative, and object-oriented.

However, your friend, who is a programming language theorist, believes C++ is not very good because it isn't pure functional. Consider the following code:

Objectively the best programming language.

```
int y = 0;
int f(int x) {
  y = y+1;
  return x*x;
}

int g(int z) {
  return z+y;
}
```

We cannot substitute occurrences of `g(f(5))` with `g(25)`, since `f` manipulates global state (side effects). Because of this, code is harder to reason about.

You'd like to challenge this belief. You decide to code a pure functional program in C++ that will solve the **longest increasing subsequence** problem.

However, in order to show that C++ can be coded in a pure functional way, you may not use `#`, `=`, or anything that mutates.

For full points, do not use `*`, and try to use less than 2 MB of memory.

Input Specification

There is no input. Instead, there will be variables and functions already defined for you. In particular, we define:

- `const int N` ($1 \leq$ `N` $\leq 1\,000$), a variable representing the length of the sequence of integers.
- `constexpr int get_item(int i)`, a function that will return the `i`$^{\text{th}}$ element of the sequence ($0 \leq$ `i` $< N$). All integers x in the sequence will satisfy $0 \leq x \leq 10^9$. Setting `i` out of range results in undefined behaviour.

- `void print_num(int n)`, a function that will print `n` to `stdout`, followed by a newline (Technically, this function isn't pure functional, but we'll let it slide - after all, understanding pure functional I/O in Haskell is a difficult task on its own).

For example, your code might look like:

```
int main() {
  print_num(N);
  print_num(get_item(2));
  return 0;
}
```

Output Specification

On the first line, output the length of the longest **strictly increasing** subsequence.

Then, on separate lines, output a longest strictly increasing subsequence. If there is more than one answer, output any of them.

Sample

Suppose `N` $= 10$ and the sequence is $3, 10, 2, 4, 8, 5, 7, 11, 6, 8$. Then one possible output is:

```
5
2
4
5
7
8
```

Notice

The spirit of the problem is to find a way to solve the problem without mutation, rather than trying to hack a method to bypass the mutation restriction. If your friend detects that your solution tries to bypass mutation, it will be rejected. If you repeatedly try to bypass mutation, your friend will assume you give up and will permanently ban you from submitting to the problem.