Challenge-2: The longest increasing subsequence

Problem:
Given a sequence of elements find the length of the longest increasing subsequence.

Definition:
Let $s : seq <int>$ be a sequence of elements $s_0, s_1, \ldots s_{n-1}$, find the length $(k+1)$ of the longest increasing subsequence $s_{i_0}, s_{i_1}, \ldots, s_{i_k}$, which satisfies that

- $0 \leq i_j < n$ for all $0 \leq j \leq k$
- $i_j < i_{j+1}$ for all $0 \leq j < k$
- $s_{i_j} < s_{i_{j+1}}$ for all $0 \leq j < k$

Remarks:

1. The longest increasing subsequence is not necessarily unique. For example, for

   $$3, 2, 2, 8, 6, 4, 6, 5, 8, 9, 7, 8$$

   there are a lot of longest increasing subsequences (whose length is 5):

   - $3, 4, 6, 8, 9$
   - $3, 4, 6, 7, 8$
   - $3, 4, 5, 8, 9$
   - $3, 4, 5, 7, 8$
   - $2, 4, 6, 8, 9$
   - $2, 4, 6, 7, 8$
   - etc.

2. Simple solutions are $O(n^2)$ worst-case complexity, where n is the length of the sequence, whereas the most efficient solutions are $O(n\ log\ n)$. You can find everywhere different algorithms (even videos in youtube) of both complexities.

3. Verification can be expected to be more intricate as more complicated is the logic of the algorithm. However, some efficient algorithm are logically simple.

4. An efficient, elegant and well explained solution is in the paper `EWD697.pdf` of Dijkstra, dated in 1978: *Some beautiful arguments using mathematical induction* (pp. 5-9). I recommend to carefully read Dijkstra's solution, understand it and implement it in Dafny.

Challenge:
Design a Dafny verified method that, given a sequence s, returns the length of the longest increasing subsequence of s.