Q 1.1)

1. Finding the values of the last two entries of the sub-sequence in using given indices and in constant time.
2. Computing a supposed value of the third last entry using a formula in constant time.
3. Using the computed value, search for the value in using binary search and get the index in in time.

Since , if is the last entry then and would be the second last and the third last in order. If knowing any 2 values of those, we would be able to determine another. In this case, we know and , then .

Overall time complexity is .

Q 1.2)

**Subproblems:** for each , Let be the problem of determining , the maximum number of elements that could form a beautiful sub-sequence having as the first element of the sequence and as the last element, and , the number that could be the new first element before .

**Recurrence:** for ,

**Base cases:** and is undefined.

( is initially 0 but is increased by 1 every time we run the whole algorithm again, )

By sorting an array in a descending order in time. Then using this algorithm times, ignoring the first element in the array and assuming the second element to be the first element instead, and so on for every time we run the algorithm again. In order to let every element in having a chance to be the last element of a beautiful sub-sequence. And keep tracking of the highest along with its (to use it in Q1.3) from all times.

The length of the longest beautiful sub-sequence of is the highest that is tracked. The longest length is 0 when the highest tracked is less than 3, because the shortest length of the sub-sequence possible is 3.

Need to add correctness?

Overall time complexity is . Since the dynamic programing method itself takes for iterating through elements, and for each element, we use Q1.1 algorithm to find that is matching with in . Then we do it times, causing it to be .

Note that even though the amount of is decreased by 1 every time (from ignoring the first elements from the array in times of running the algorithm), but the time complexity is still . Since

And .

Q 1.3)

By using of the highest that has been tracked. Then using dynamic programing method from Q1.2 having as the first element in order to determine all and of that set again in time (as explained above). (Note that this is using the sorted )

Then starting with , put as the first element of the list. And search for an index (using binary search in time) then put as the next element of the list. Using the new index as and look for a new index and so on until .

Correctness...

Overall time complexity is . For using binary search times (in the worst case), for every element in causing it to be .