Question 2. [12 MARKS]

Given an array of n integers $[x(1), \ldots, x(n)]$ define an **increasing subsequence** of x, to be a subsequence $x(j(1)) < x(j(2)) < \ldots < x(j(p))$ where the indicies $j(i)$ are also strictly increasing with i, that is, $1 \leq j(1) < j(2) < \ldots < j(p) \leq n$. We are interested in a **longest increasing subsequence** of x, i.e., with the maximum length p. Define $LLIS(x)$ to be the **length** of a longest increasing subsequence of x (there may be more than one increasing subsequence with this length).

For example, given the array $x = [22, 5, 8, -3, 10, 1]$ the longest increasing subsequence is $[5, 8, 10]$ which has length is 3. Therefore, for this example, $LLIS(x) = 3$.

In addition, define the array $[q(1), \ldots, q(n)]$ to have elements $q(k)$ which equal the length of the longest increasing subsequence of x **ending with the value** $x(k)$. In particular, such a subsequence only uses values from the first k elements of x, namely $[x(1), \ldots, x(k)]$ and must end with $x(k)$ itself.

For the example array x above, $[q(1), \ldots, q(6)] = [1, 1, 2, 1, 3, 2]$.

Part (a) [5 MARKS]

Give an equation which expresses $q(k)$ for $1 < k \leq n$ in terms of the values of $q(j)$ for various $j < k$ (and possibly other simple expressions involving elements of the array x). Explain.

Solution:
$q(k) \ = \ 1 + \max\left[\{0\} \cup \{q(j) \mid 1 \leq j < k \text{ and } x(j) < x(k)\}\right]$

Explanation: Two cases for the longest increasing subsequence ending at x(k):
Case 1: The subsequence is length 1, consisting of x(k) alone. This case is dealt with
by including the term $\{0\}$ in the max above (the second set could be empty).
Case 2: The longest subsequence ending at $x(k)$ has length larger than 1.
Let the second last item in the subsequence be x(j) for some $j < k$.
For the subsequence to be increasing, we require that $x(j) < x(k)$.
Since we are looking for the longest subsequence, we must use the longest
subsequence ending at $x(j)$, which has length $q(j)$.
With $x(k)$ appended, the length of the resulting subsequence is $1 + q(j)$.
Therefore $q(k) \geq 1 + q(j)$ for each such carefully chosen j (if any).
Finally, since we are looking for the longest increasing subsequence
we want to maximize over all possible choices from these two cases.

Part (b) [2 MARKS]

Express $LLIS(x)$ in terms of the array q. Explain.

Solution:
$LLIS(x) = \max\{q(k) \mid 1 \leq k \leq n\}$.
Explanation: The longest increasing subsequence has to end at some $x(k)$, in which case it has length $q(k)$. So we should choose the max $q(k)$.

Question 2. (CONTINUED)

Part (c) [5 MARKS]

Given the arrays x and q defined above, provide pseudo-code for an algorithm which extracts a longest increasing subsequence. (Precisely worded English sentences describing simple steps are allowed in your pseudo-code.) In the previous notation, the algorithm should return $[x(j(1)), x(j(2)), \ldots, x(j(p))]$ for the maximum p. The $j(k)$'s themselves do not need to be returned, just the corresponding x values (in order). Briefly explain why your algorithm is correct.

Solution:

```
[s] = LIS(x, q, n)
   Find an index k such that q(k) is the maximum of [q(1), ..., q(n)].
   s = [ ]          // s is initially an empty list.
   while q(k) >= 1
      s = [x(k) s]      // Prepend s with x(k).
      if q(k) == 1
         return s
      Find (any) j in the range 1 ≤ j < k such that q(j) = q(k) - 1 and x(j) < x(k).
         Raise an exception if such a j does not exist.
      k = j
   end
```

Explanation:

As in part (b) above, the longest inceasing subsequence ends at x(k), for a k where q(k) is maximal.
For $q(k) > 1$, the second-last item x(j) must have q(j) = q(k) - 1, since the subsequence ending at $x(j)$ is just one shorter.
This second-last item x(j) must also satisfy $x(j) < x(k)$ since the subsequence is increasing.
The algorithm finds exactly such a j and iterates.
Each iteration decreases q(k) by one, until the q(k) = 1.
For q(k) = 1 we know the first element of the sub-sequence is x(k).