A ***subsequence*** of a sequence (for example, an array, linked list, or string), obtained by removing zero or more elements and keeping the rest in the same sequence order. A subsequence is called a ***substring*** if its elements are contiguous in the original sequence. For example:

- `SUBSEQUENCE`, `UBSEQU`, and the empty string ε are all substrings (and therefore subsequences) of the string `SUBSEQUENCE`;
- `SBSQNC`, `SQUEE`, and `EEE` are all subsequences of `SUBSEQUENCE` but not substrings;
- `QUEUE`, `EQUUS`, and `DIMAGGIO` are not subsequences (and therefore not substrings) of `SUBSEQUENCE`.

Describe recursive backtracking algorithms for the following problems. *Don't worry about running times.*

1. Given an array $A[1 .. n]$ of integers, compute the length of a ***longest increasing subsequence***. A sequence $B[1 .. \ell]$ is *increasing* if $B[i] > B[i-1]$ for every index $i \geq 2$.

   For example, given the array

   $$\langle 3, \underline{\mathbf{1}}, \underline{\mathbf{4}}, 1, \underline{\mathbf{5}}, 9, 2, \underline{\mathbf{6}}, 5, 3, 5, \underline{\mathbf{8}}, \underline{\mathbf{9}}, 7, 9, 3, 2, 3, 8, 4, 6, 2, 7 \rangle$$

   your algorithm should return the integer 6, because $\langle 1, 4, 5, 6, 8, 9 \rangle$ is a longest increasing subsequence (one of many).

2. Given an array $A[1 .. n]$ of integers, compute the length of a ***longest decreasing subsequence***. A sequence $B[1 .. \ell]$ is *decreasing* if $B[i] < B[i-1]$ for every index $i \geq 2$.

   For example, given the array

   $$\langle 3, 1, 4, 1, 5, \underline{\mathbf{9}}, 2, \underline{\mathbf{6}}, 5, 3, \underline{\mathbf{5}}, 8, 9, 7, 9, 3, 2, 3, 8, \underline{\mathbf{4}}, 6, \underline{\mathbf{2}}, 7 \rangle$$

   your algorithm should return the integer 5, because $\langle 9, 6, 5, 4, 2 \rangle$ is a longest decreasing subsequence (one of many).

3. Given an array $A[1 .. n]$ of integers, compute the length of a ***longest alternating subsequence***. A sequence $B[1 .. \ell]$ is *alternating* if $B[i] < B[i-1]$ for every even index $i \geq 2$, and $B[i] > B[i-1]$ for every odd index $i \geq 3$.

   For example, given the array

   $$\langle \underline{\mathbf{3}}, \underline{\mathbf{1}}, \underline{\mathbf{4}}, \underline{\mathbf{1}}, \underline{\mathbf{5}}, 9, \underline{\mathbf{2}}, \underline{\mathbf{6}}, \underline{\mathbf{5}}, 3, 5, \underline{\mathbf{8}}, 9, \underline{\mathbf{7}}, \underline{\mathbf{9}}, \underline{\mathbf{3}}, 2, 3, \underline{\mathbf{8}}, \underline{\mathbf{4}}, \underline{\mathbf{6}}, \underline{\mathbf{2}}, \underline{\mathbf{7}} \rangle$$

   your algorithm should return the integer 17, because $\langle 3, 1, 4, 1, 5, 2, 6, 5, 8, 7, 9, 3, 8, 4, 6, 2, 7 \rangle$ is a longest alternating subsequence (one of many).

To think about later:

4. Given an array $A[1 .. n]$ of integers, compute the length of a longest ***convex*** subsequence of A. A sequence $B[1 .. \ell]$ is *convex* if $B[i] - B[i-1] > B[i-1] - B[i-2]$ for every index $i \geq 3$.

   For example, given the array

   $$\langle \underline{\mathbf{3}}, \underline{\mathbf{1}}, 4, \underline{\mathbf{1}}, 5, 9, \underline{\mathbf{2}}, 6, 5, 3, \underline{\mathbf{5}}, 8, \underline{\mathbf{9}}, 7, 9, 3, 2, 3, 8, 4, 6, 2, 7 \rangle$$

   your algorithm should return the integer 6, because $\langle 3, 1, 1, 2, 5, 9 \rangle$ is a longest convex subsequence (one of many).

5. Given an array $A[1 .. n]$, compute the length of a longest ***palindrome*** subsequence of A. Recall that a sequence $B[1 .. \ell]$ is a *palindrome* if $B[i] = B[\ell - i + 1]$ for every index i.

   For example, given the array

   $$\langle \underline{\mathbf{3}}, 1, 4, 1, 5, 9, \underline{\mathbf{2}}, 6, 5, \underline{\mathbf{3}}, 5, 8, \underline{\mathbf{9}}, \underline{\mathbf{7}}, \underline{\mathbf{9}}, \underline{\mathbf{3}}, \underline{\mathbf{2}}, \underline{\mathbf{3}}, 8, 4, 6, 2, 7 \rangle$$

   your algorithm should return the integer 9, because $\langle 3, 2, 3, 9, 7, 9, 3, 2, 3 \rangle$ is a longest palindrome subsequence (there may be others).