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.

**Solution (#1 of $\infty$):** Add a sentinel value $A[0] = -\infty$. Let $LIS(i, j)$ denote the length of the longest increasing subsequence of $A[j..n]$ where every element is larger than $A[i]$. This function obeys the following recurrence:

$$LIS(i, j) = \begin{cases} 
0 & \text{if } j > n \\
LIS(i, j + 1) & \text{if } j \leq n \text{ and } A[i] \geq A[j] \\
\max\{LIS(i, j + 1), 1 + LIS(j, j + 1)\} & \text{otherwise}
\end{cases}$$

We need to compute $LIS(0, 1)$.

**Solution (#2 of $\infty$):** Add a sentinel value $A[n + 1] = -\infty$. Let $LIS(i, j)$ denote the length of the longest increasing subsequence of $A[1..j]$ where every element is smaller than $A[j]$. This function obeys the following recurrence:

$$LIS(i, j) = \begin{cases} 
0 & \text{if } i < 1 \\
LIS(i - 1, j) & \text{if } i \geq 1 \text{ and } A[i] \geq A[j] \\
\max\{LIS(i - 1, j), 1 + LIS(i - 1, i)\} & \text{otherwise}
\end{cases}$$

We need to compute $LIS(n, n + 1)$.

**Solution (#3 of $\infty$):** Let $LIS(i)$ denote the length of the longest increasing subsequence of $A[i..n]$ that begins with $A[i]$. This function obeys the following recurrence:

$$LIS(i) = \begin{cases} 
1 & \text{if } A[j] \leq A[i] \text{ for all } j > i \\
1 + \max\{LIS(j) \mid j > i \text{ and } A[j] > A[i]\} & \text{otherwise}
\end{cases}$$

(The first case is actually redundant if we define $\max\emptyset = 0$.) We need to compute $\max_i LIS(i)$.

**Solution (#4 of $\infty$):** Add a sentinel value $A[0] = -\infty$. Let $LIS(i)$ denote the length of the longest increasing subsequence of $A[i..n]$ that begins with $A[i]$. This function obeys the following recurrence:

$$LIS(i) = \begin{cases} 
1 & \text{if } A[j] \leq A[i] \text{ for all } j > i \\
1 + \max\{LIS(j) \mid j > i \text{ and } A[j] > A[i]\} & \text{otherwise}
\end{cases}$$

(The first case is actually redundant if we define $\max\emptyset = 0$.) We need to compute $LIS(0) - 1$; the $-1$ removes the sentinel $-\infty$ from the start of the subsequence.

**Solution (#5 of $\infty$):** Add sentinel values $A[0] = -\infty$ and $A[n + 1] = \infty$. Let $LIS(j)$ denote the length of the longest increasing subsequence of $A[1..j]$ that ends with $A[j]$. This function obeys the following recurrence:

$$LIS(j) = \begin{cases} 
1 & \text{if } j = 0 \\
1 + \max\{LIS(i) \mid i < j \text{ and } A[i] < A[j]\} & \text{otherwise}
\end{cases}$$

We need to compute $LIS(n + 1) - 2$; the $-2$ removes the sentinels $-\infty$ and $\infty$ from the subsequence.
2. Given an array $A[1..n]$ of integers, compute the length of a longest decreasing subsequence.

**Solution (one of many):** Add a sentinel value $A[0] = \infty$. Let $LDS(i, j)$ denote the length of the longest decreasing subsequence of $A[j..n]$ where every element is smaller than $A[i]$. This function obeys the following recurrence:

$$
LDS(i, j) = \begin{cases}
0 & \text{if } j > n \\
LDS(i, j + 1) & \text{if } j \leq n \text{ and } A[i] \leq A[j] \\
\max\{LDS(i, j + 1), 1 + LIS(j, j + 1)\} & \text{otherwise}
\end{cases}
$$

We need to compute $LDS(0, 1)$.

**Solution (clever):** Multiply every element of $A$ by $-1$, and then compute the length of the longest increasing subsequence using the algorithm from problem 1.
3. Given an array $A[1..n]$ of integers, compute the length of a **longest alternating subsequence**.

**Solution (one of many):** We define two functions:

- Let $LAS^+(i, j)$ denote the length of the longest alternating subsequence of $A[j..n]$ whose first element (if any) is larger than $A[i]$ and whose second element (if any) is smaller than its first.
- Let $LAS^-(i, j)$ denote the length of the longest alternating subsequence of $A[j..n]$ whose first element (if any) is smaller than $A[i]$ and whose second element (if any) is larger than its first.

These two functions satisfy the following mutual recurrences:

\[
LAS^+(i, j) = \begin{cases} 
0 & \text{if } j > n \\
LAS^+(i, j + 1) & \text{if } j \leq n \text{ and } A[j] \leq A[i] \\
\max \{LAS^+(i, j + 1), 1 + LAS^-(j, j + 1)\} & \text{otherwise}
\end{cases}
\]

\[
LAS^-(i, j) = \begin{cases} 
0 & \text{if } j > n \\
LAS^-(i, j + 1) & \text{if } j \leq n \text{ and } A[j] \geq A[i] \\
\max \{LAS^-(i, j + 1), 1 + LAS^+(j, j + 1)\} & \text{otherwise}
\end{cases}
\]

To simplify computation, we consider two different sentinel values $A[0]$. First we set $A[0] = -\infty$ and let $\ell^+ = LAS^+(0, 1)$. Then we set $A[0] = +\infty$ and let $\ell^- = LAS^-(0, 1)$. Finally, the length of the longest alternating subsequence of $A$ is $\max \{\ell^+, \ell^-\}$.

**Solution (one of many):** We define two functions:

- Let $LAS^+(i)$ denote the length of the longest alternating subsequence of $A[i..n]$ that starts with $A[i]$ and whose second element (if any) is larger than $A[i]$.
- Let $LAS^-(i)$ denote the length of the longest alternating subsequence of $A[i..n]$ that starts with $A[i]$ and whose second element (if any) is smaller than $A[i]$.

These two functions satisfy the following mutual recurrences:

\[
LAS^+(i) = \begin{cases} 
1 & \text{if } A[j] \leq A[i] \text{ for all } j > i \\
1 + \max \{LAS^-(j) \mid j > i \text{ and } A[j] > A[i]\} & \text{otherwise}
\end{cases}
\]

\[
LAS^-(i) = \begin{cases} 
1 & \text{if } A[j] \geq A[i] \text{ for all } j > i \\
1 + \max \{LAS^+(j) \mid j > i \text{ and } A[j] < A[i]\} & \text{otherwise}
\end{cases}
\]

We need to compute $\max_i \max \{LAS^+(i), LAS^-(i)\}$.
To think about later:


**Solution:** Let $LCS(i, j)$ denote the length of the longest convex subsequence of $A[i..n]$ whose first two elements are $A[i]$ and $A[j]$. This function obeys the following recurrence:

$$LCS(i, j) = 1 + \max\{LCS(j, k) \mid j < k \leq n \text{ and } A[i] + A[k] > 2A[j]\}$$

Here we define $\max\emptyset = 0$; this gives us a working base case. The length of the longest convex subsequence is $\max_{1 \leq i < j \leq n} LCS(i, j)$.

**Solution (with sentinels):** Assume without loss of generality that $A[i] \geq 0$ for all $i$. (Otherwise, we can add $|m|$ to each $A[i]$, where $m$ is the smallest element of $A[1..n]$.) Add two sentinel values $A[0] = 2M + 1$ and $A[-1] = 4M + 3$, where $M$ is the largest element of $A[1..n]$.

Let $LCS(i, j)$ denote the length of the longest convex subsequence of $A[i..n]$ whose first two elements are $A[i]$ and $A[j]$. This function obeys the following recurrence:

$$LCS(i, j) = 1 + \max\{LCS(j, k) \mid j < k \leq n \text{ and } A[i] + A[k] > 2A[j]\}$$

Here we define $\max\emptyset = 0$; this gives us a working base case.

Finally, we claim that the length of the longest convex subsequence of $A[1..n]$ is $LCS(-1, 0) - 2$.

**Proof:** First, consider any convex subsequence $S$ of $A[1..n]$, and suppose its first element is $A[i]$. Then we have $A[-1] - 2A[0] + A[i] = 4M + 3 - 2(2M + 1) + A[i] = A[i] + 1 > 0$, which implies that $A[-1] \cdot A[0] \cdot S$ is a convex subsequence of $A[-1..n]$. So the longest convex subsequence of $A[1..n]$ has length at most $LCS(-1, 0) - 2$.

On the other hand, removing $A[-1]$ and $A[0]$ from any convex subsequence of $A[-1..n]$ leaves a convex subsequence of $A[1..n]$. So the longest subsequence of $A[1..n]$ has length at least $LCS(-1, 0) - 2$. □
5. Given an array \( A[1..n] \), compute the length of a longest \textit{palindrome} subsequence of \( A \).

\textbf{Solution (naïve)}: Let \( LPS(i, j) \) denote the length of the longest palindrome subsequence of \( A[i..j] \). This function obeys the following recurrence:

\[
LPS(i, j) = \begin{cases} 
0 & \text{if } i > j \\
1 & \text{if } i = j \\
\max\left\{LPS(i + 1, j), LPS(i, j - 1)\right\} & \text{if } i < j \text{ and } A[i] \neq A[j] \\
2 + LPS(i + 1, j - 1) & \text{otherwise}
\end{cases}
\]

We need to compute \( LPS(1, n) \).

\textbf{Solution (with greedy optimization)}: Let \( LPS(i, j) \) denote the length of the longest palindrome subsequence of \( A[i..j] \). Before stating a recurrence for this function, we make the following useful observation.\(^1\)

\textbf{Claim 1}. If \( i < j \) and \( A[i] = A[j] \), then \( LPS(i, j) = 2 + LPS(i + 1, j - 1) \).

\textbf{Proof}: Suppose \( i < j \) and \( A[i] = A[j] \). Fix an arbitrary longest palindrome subsequence \( S \) of \( A[i..j] \). There are four cases to consider:

- If \( S \) uses neither \( A[i] \) nor \( A[j] \), then \( A[i] \cdot S \cdot A[j] \) is a palindrome subsequence of \( A[i..j] \) that is longer than \( S \), which is impossible.
- Suppose \( S \) uses \( A[i] \) but not \( A[j] \). Let \( A[k] \) be the last element of \( S \). If \( k = i \), then \( A[i] \cdot A[j] \) is a palindrome subsequence of \( A[i..j] \) that is longer than \( S \), which is impossible. Otherwise, replacing \( A[k] \) with \( A[j] \) gives us a palindrome subsequence of \( A[i..j] \) with the same length as \( S \) that uses both \( A[i] \) and \( A[j] \).
- Suppose \( S \) uses \( A[j] \) but not \( A[i] \). Let \( A[h] \) be the first element of \( S \). If \( h = j \), then \( A[i] \cdot A[j] \) is a palindrome subsequence of \( A[i..j] \) that is longer than \( S \), which is impossible. Otherwise, replacing \( A[h] \) with \( A[i] \) gives us a palindrome subsequence of \( A[i..j] \) with the same length as \( S \) that uses both \( A[i] \) and \( A[j] \).
- Finally, \( S \) might include both \( A[i] \) and \( A[j] \).

In all cases, we find either a contradiction or a longest palindrome subsequence of \( A[i..j] \) that uses both \( A[i] \) and \( A[j] \).

Claim 1 implies that the function \( LPS \) satisfies the following recurrence:

\[
LPS(i, j) = \begin{cases} 
0 & \text{if } i > j \\
1 & \text{if } i = j \\
\max\{LPS(i + 1, j), LPS(i, j - 1)\} & \text{if } i < j \text{ and } A[i] \neq A[j] \\
2 + LPS(i + 1, j - 1) & \text{otherwise}
\end{cases}
\]

We need to compute \( LPS(1, n) \). \( \blacksquare \)

\(^1\)And yes, optimizations like this require a proof of correctness, both in homework and on exams. Premature optimization is the root of all evil.