Analyzing the Running Time of a Simple Recursive Algorithm
Solutions for a Suggested Exercise

This exercise concerned the “Maximal Element in Part of an Integer Array” problem, considered in Reading #2, as well as the following algorithm — which was proved to correctly solve this problem in the exercise for that reading.

```java
maxInRange2 ( integer[] A, integer low, integer high ) {
1. if (low == high) {
2. return A[low]
} else {
3. return max(maxInRange2(A, low, high - 1), A[high])
}
}
```

1. You were first asked to used the uniform cost criterion to write a recurrence for the number $T_{\text{max}}(k)$ of steps used by this algorithm when $0 \leq \text{low} \leq \text{high} \leq \text{A}.\text{length} - 1$ and $\text{high} - \text{low} + 1 = k \geq 1$.

**Solution:** Suppose first that $k = 1$. Then $\text{high} - \text{low} + 1 = k = 1$, so that $\text{low} = \text{high}$ and the test at line 1 is passed when checked (as part of an execution of the algorithm including these inputs). The algorithm then ends after a second step — at line 2 — is executed, so that $T_{\text{max}}(1) = 2$.

Suppose, instead, that $k \geq 2$. Then $\text{high} - \text{low} + 1 = k \geq 2$, so that $\text{high} \geq \text{low} + 1$, and the test at line 1 fails when checked. In this case, execution continues, and ends, with the execution of the step at line 3.
However, this includes a recursive application of this algorithm with inputs \( \text{low}' = \text{low} \) and \( \text{high}' = \text{high} - 1 \). Now

\[
\text{high}' - \text{low}' + 1 = (\text{high} - 1) - \text{low} + 1
\]

\[
= (\text{high} - \text{low} + 1) - 1 \quad \text{(reordering and grouping terms)}
\]

\[
= k - 1
\]

so that the recursive application of the algorithm at line 3 uses \( T_{\text{max}}(k - 1) \) steps.

Thus a recurrence for \( T_{\text{max}}(k) \) is as follows: If \( k \) is a positive integer then

\[
T_{\text{max}}(k) = \begin{cases} 
2 & \text{if } k = 1, \\
T_{\text{max}}(k - 1) + 2 & \text{if } k \geq 2.
\end{cases}
\]

2. You were next asked to guess a solution for this recurrence — that is, guess an expression for \( T_{\text{max}}(k) \) that is not in the form of a recurrence.

**Solution:** Notice that

- \( T_{\text{max}}(1) = 2 \),
- \( T_{\text{max}}(2) = T_{\text{max}}(1) + 2 = 2 + 2 = 4 \),
- \( T_{\text{max}}(3) = T_{\text{max}}(2) + 2 = 4 + 2 = 6 \), and
- \( T_{\text{max}}(4) = T_{\text{max}}(3) + 2 = 6 + 2 = 8 \).

Thus after checking the value of \( T_{\text{max}}(k) \) for small positive values of \( k \) it might be reasonable to guess that \( T_{\text{max}}(k) = 2k \) for every positive integer \( k \).

3. Finally you were asked to prove that your guess is correct.

**Solution:**

**Claim:** Suppose that \( T_{\text{max}} : \mathbb{N} \rightarrow \mathbb{N} \) such that, for every positive integer \( k \),

\[
T_{\text{max}}(k) = \begin{cases} 
2 & \text{if } k = 1, \\
T_{\text{max}}(k - 1) + 2 & \text{if } k \geq 2.
\end{cases}
\]

Then \( T_{\text{max}}(k) = 2k \) for every positive integer \( k \).

**Proof:** By induction on \( k \). The standard form of mathematical induction will be used.

**Basis:** If \( k = 1 \) then

\[
T_{\text{max}}(k) = T_{\text{max}}(1)
\]

\[
= 2
\]

(by the given recurrence for \( T_{\text{max}}(k) \))

\[
= 2 \cdot 1 = 2k
\]
as required to establish the claim in this case.

Inductive Step: Let $h$ be an integer such that $h \geq 1$. It is necessary and sufficient to use the following

Inductive Hypothesis: $T_{\text{max}}(h) = 2h$.

to prove the following

Inductive Claim: $T_{\text{max}}(h + 1) = 2(h + 1)$.

Since $h \geq 1$, $h + 1 \geq 2$. Therefore

$$T_{\text{max}}(h + 1) = T_{\text{max}}((h + 1) - 1) + 2 \quad \text{(by the given recurrence for $T_{\text{max}}(k)$)}$$

$$= T_{\text{max}}(h) + 2$$

$$= 2h + 2 \quad \text{(by the Inductive Hypothesis)}$$

$$= 2(h + 1)$$

— establishing the Inductive Claim, as needed to complete the Inductive Step and complete the proof of the claim. □