Examples of Recurrence Equations

MERGE-SORT (worst case)
Solve two subproblems of size $n/2$ each and perform a linear amount of work to merge the sorted arrays.

QUICK-SORT (best case). Solve two subproblems of size $n/2$ each and perform a linear amount of work to partition the sorted arrays.

$$T(n) = \begin{cases} c & \text{if } n \leq 1 \\ 2T(\frac{n}{2}) + cn & \text{if } n > 1 \end{cases}$$

BINARY-SEARCH (worst case)
Compare the key to the mid element (constant amount of time) and then solve one subproblem of size $n/2$.

$$T(n) = \begin{cases} c & \text{if } n \leq 1 \\ T(\frac{n}{2}) + c & \text{if } n > 1 \end{cases}$$

QUICK-SELECT (an excellent case!)
Partition the array (linear amount of time) and then solve one subproblem of size $n/2$.

$$T(n) = \begin{cases} c & \text{if } n \leq 1 \\ T(\frac{n}{2}) + cn & \text{if } n > 1 \end{cases}$$

```
FIND-MAX(a[], first, last):
    if (first == last) return a[first]

    mid = first + (last-first) / 2

    left  = FIND-MAX(a, first, mid)
    right = FIND-MAX(a, mid+1, last)

    return MAX(left, right)
```

$$T(n) = \begin{cases} c & \text{if } n \leq 1 \\ 2T(\frac{n}{2}) + c & \text{if } n > 1 \end{cases}$$

```
FIND-MAX(a[], first, last):
   if (first == last) return a[first]

   return MAX(a[first],
              FIND-MAX(a, first+1, last)
```

$$T(n) = \begin{cases} c & \text{if } n \leq 1 \\ T(n-1) + c & \text{if } n > 1 \end{cases}$$

```
SELECTION-SORT(a[], first, last):
   if (first >= last) return

   min_index = FIND-MIN(a, first, last)
   SWAP(a[min_index], a[first])

   SELECTION-SORT(a, first+1, last)
```

$$T(n) = \begin{cases} c & \text{if } n \leq 1 \\ T(n-1) + cn & \text{if } n > 1 \end{cases}$$

The same recurrence applies also to:
QUICK-SORT (worst case)
QUICK-SELECT (worst case)