```
BINARY-SEARCH(A, p, r, x)
if p > r
    return NIL
q = ⌊(p + r)/2⌋
if A[q] = x
    return q
if A[q] > x
    return BINARY-SEARCH(A, p, q, x)
else
    return BINARY-SEARCH(A, q + 1, r, x)
```

In the worst case, BINARY-SEARCH keeps running until $p = r$ or $p > r$. Thus the worst case running time of binary search, $T(n)$ where $n = p - r + 1$, can be defined as such

$$T(n) = \begin{cases} c_1 & n \leq 1 \\ T(n/2) + c_2 & n > 1 \end{cases}$$

We'll look at how to solve recursions like these more formally in the upcoming chapters, but for now, notice that this recursion will expand until we have $T(n) = T(n/2^k) + c_2 k = c1 + c_2 k$ where $n/2^k \leq 1 \implies k \geq \lg n$.