```

\[
\text{depth} \begin{cases}
    \text{Binary-Search} (A, 1, 8, k) \\
        \text{BS}(A, 1, 4, k) \\
        \text{BS}(A, 3, 4, k) \\
        \text{BS}(A, 3, 3, k) \\
    \end{cases}
\]

8 \to 3
16 \to 4
32 \to 5
64 \to 6
128 \to 7

\text{Binary-Search}(A, p, r, \text{key})
\text{if } p == r
    \text{return } \text{key} == A[p]
\text{else}
    q = \lfloor (p + r)/2 \rfloor
    \text{if } A[q] < \text{key}
        \text{return } \text{Binary-Search}(A, q + 1, r, \text{key})
    \text{else}
        \text{return } \text{Binary-Search}(A, p, q, \text{key})
\text{BS}(A, 1, n, k)
\text{depth of recursion tree}
\text{is } \log n
\text{C}(n) = \# \text{ of BS calls for subarray of size } n
\text{C}(n) = 1 + \left( \frac{C(n)}{2} \right)
\text{in a call to binary search}
\text{one call to BS on a subarray half the size}
```
\[ \text{depth of recursion tree is } \log n \]

\[ C(n) = \text{# of MS calls for size } n \text{ subarray} \]

\[ C(n) = 1 + 2 \cdot C\left(\frac{n}{2}\right) \]

- Makes 2 recursive calls in the code
- Subarray of recursive call is half the size

\[ C(1) = 1 \] because no recursion for subarray of size 1
# of nodes in recursion tree

\[ MS(A,1,1) = 1 \]
\[ MS(A,1,2) = 3 \]
\[ MS(A,1,4) = 7 \]
\[ MS(A,1,8) = 15 \]
\[ MS(A,1,16) = 31 \]
\[ MS(A,1,32) = 63 \]

**Note**
\[ MS(A,1,n) = 2n-1 \text{ MS calls} \]
\[ C(n) = 1 + 2 C \left( \frac{n}{2} \right) \]
\[ C(1) = 1 \]

Solution is \[ C(n) = 2n - 1 \]

Mathematical induction to show
\[ C(1) = 1, \quad C(n) = 1 + 2 C \left( \frac{n}{2} \right) \]
implies \[ C(n) = 2n - 1 \text{ for powers of two} \]

**Base Case**
\[ 2(1) - 1 = 1 = C(1) \]

**Induction**
\[ \text{Assume true for values } \lt n \]
\[ C(n) = 1 + 2 C \left( \frac{n}{2} \right) = 1 + 2 \left( 2^{n/2} - 1 \right) = 2n - 1 \]
\[ C(1) = 1 \]

\[ C(l) = a + 1 \]

\[ C(l^2) = a(a+1) + 1 \]

\[ C(l^3) = a(a^2+a+1) + 1 \]

\[ C(l^d) = a^d + a^{d-1} + a^{d-2} + \ldots + 1 \]

Depth of recursion tree:

\[ a > 1 \]

\[ a^d = a^{\left\lfloor \log_b n \right\rfloor} = n \]
Master method for number of recursive calls

\[ C(n) = 1 + a \cdot C\left(\frac{n}{b}\right) \]

- In a call to the algorithm
- Number of recursive calls by this call
  - \( a = 2 \) for MS
  - \( a = 1 \) for BS

- Work in recursive calls are on subproblems of \( \frac{1}{b} \) of the original size
  - \( \frac{1}{b} = \frac{1}{2} \) for MS & BS
  - \( b = 2 \)

Recursion tree
- \( a \) branches from each internal node

\[ \log_b n \text{ is depth of recursion tree} \]

\[ \log_b n = \frac{\log n}{\log b} \]