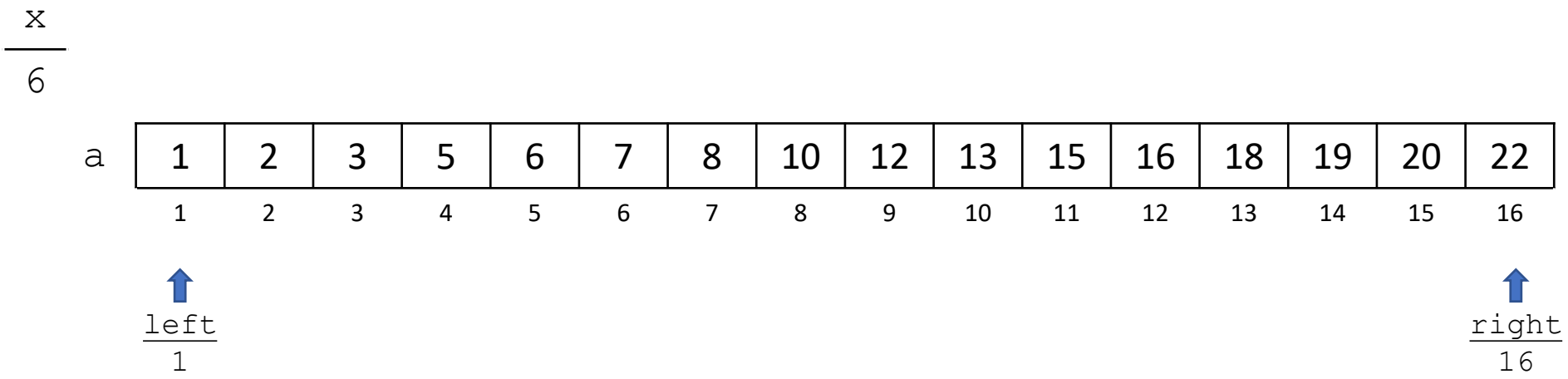
Section 8.14
Divide-and-Conquer Algorithms: Binary Search

Binary Search

- Searching for a value in a sorted sequence
- Works like looking for a word in the dictionary

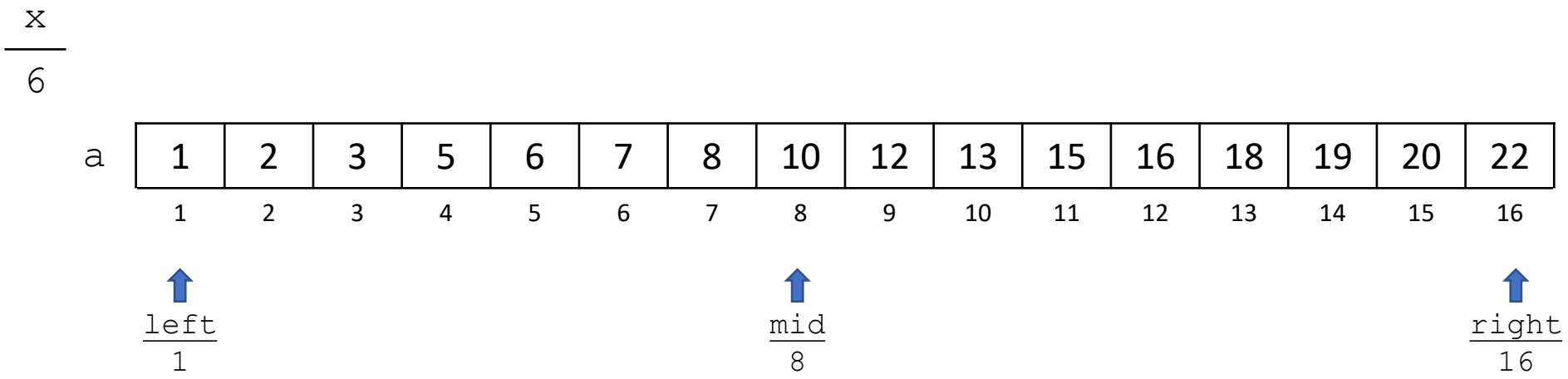
Binary Search

- Example: Find 6 in the ordered list of 16 integers



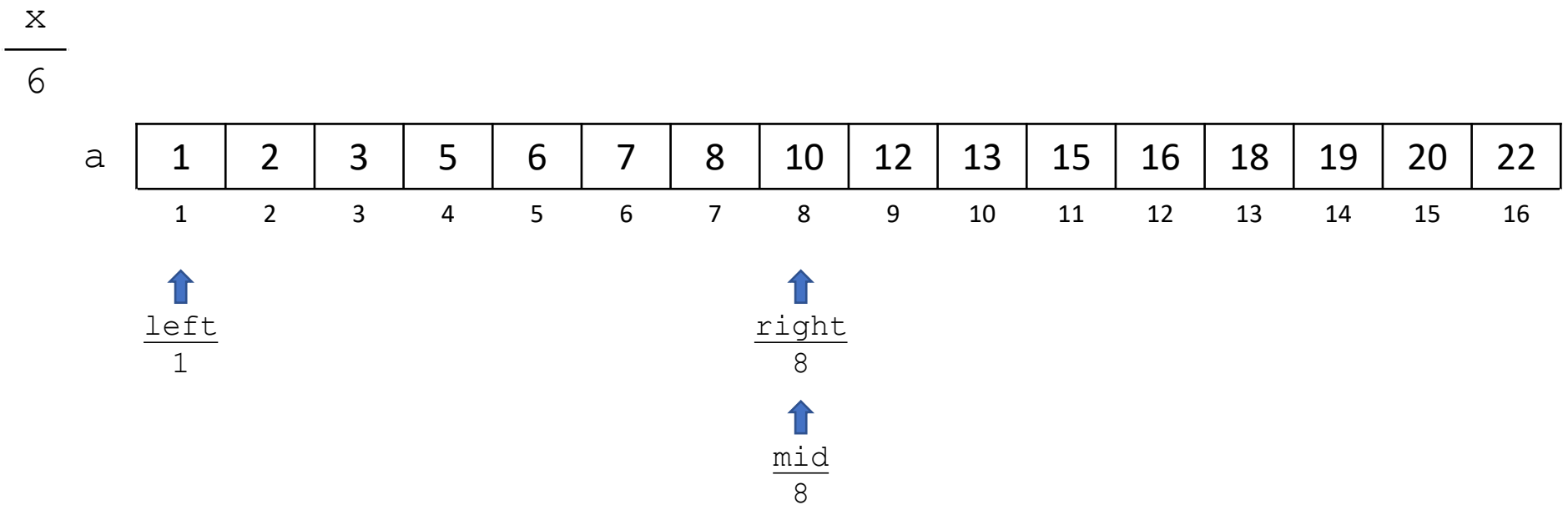
Binary Search

- Example: Find 6 in the ordered list of 16 integers



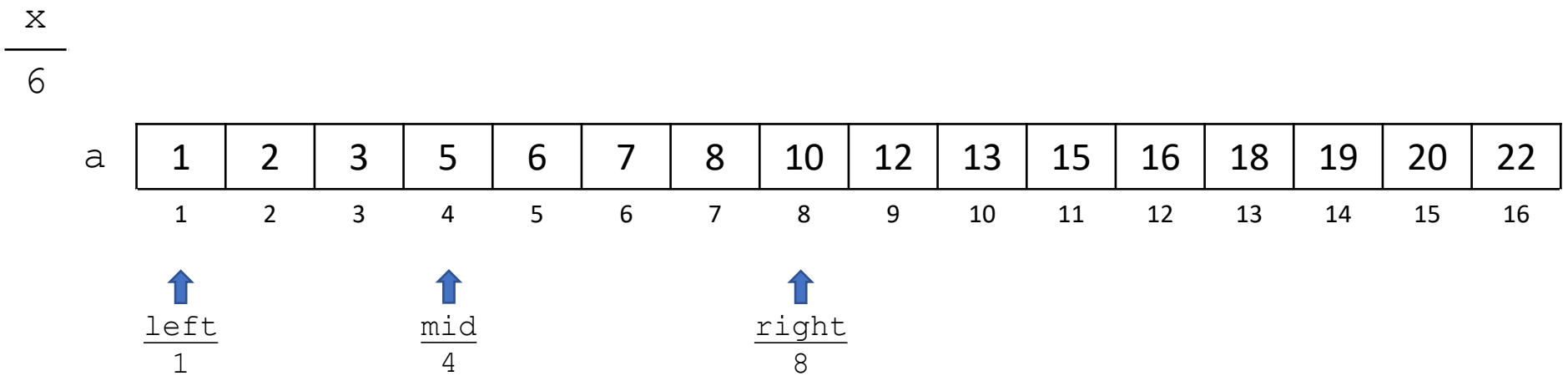
Binary Search

- Example: Find 6 in the ordered list of 16 integers



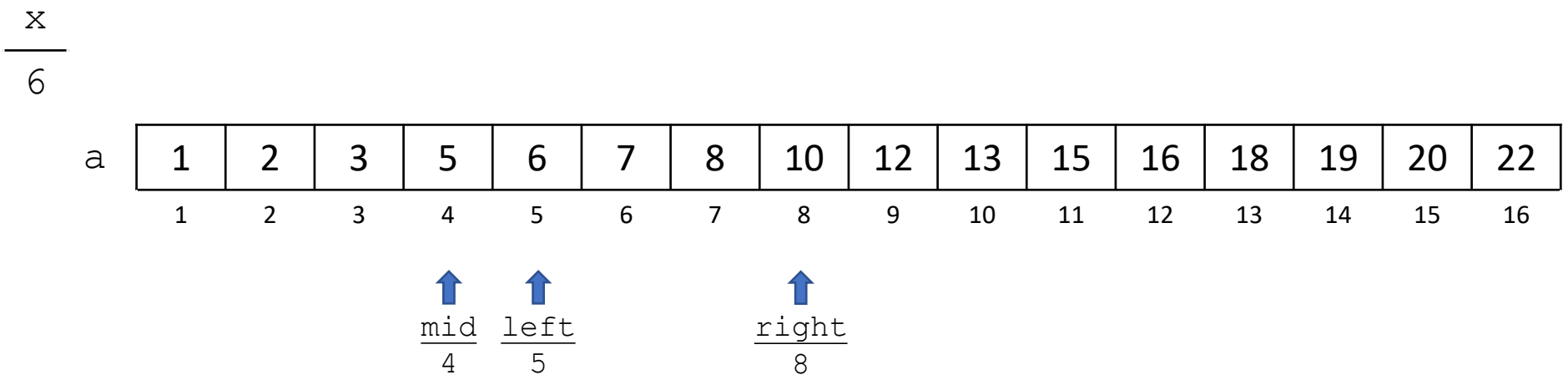
Binary Search

- Example: Find 6 in the ordered list of 16 integers



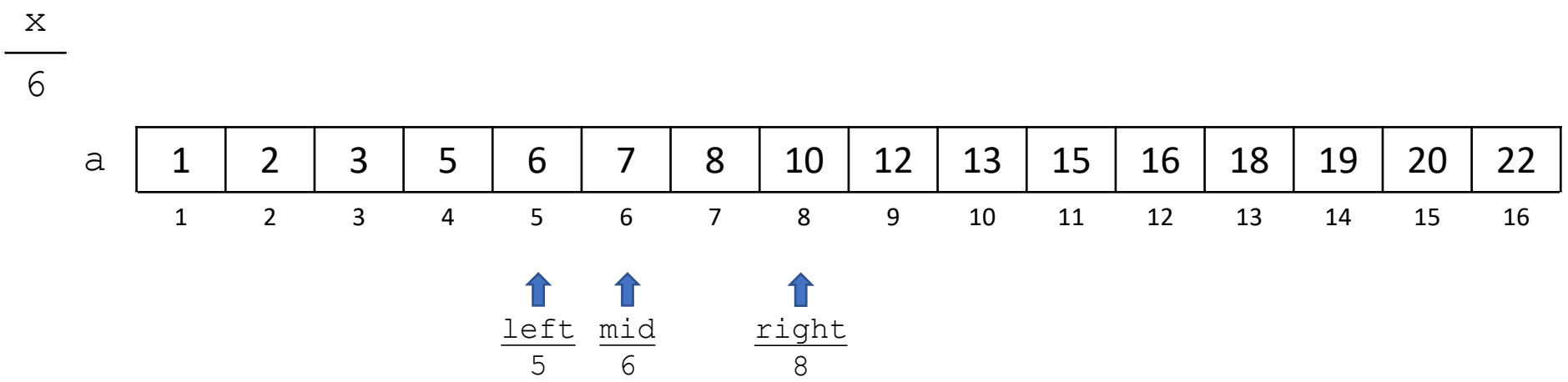
Binary Search

- Example: Find 6 in the ordered list of 16 integers



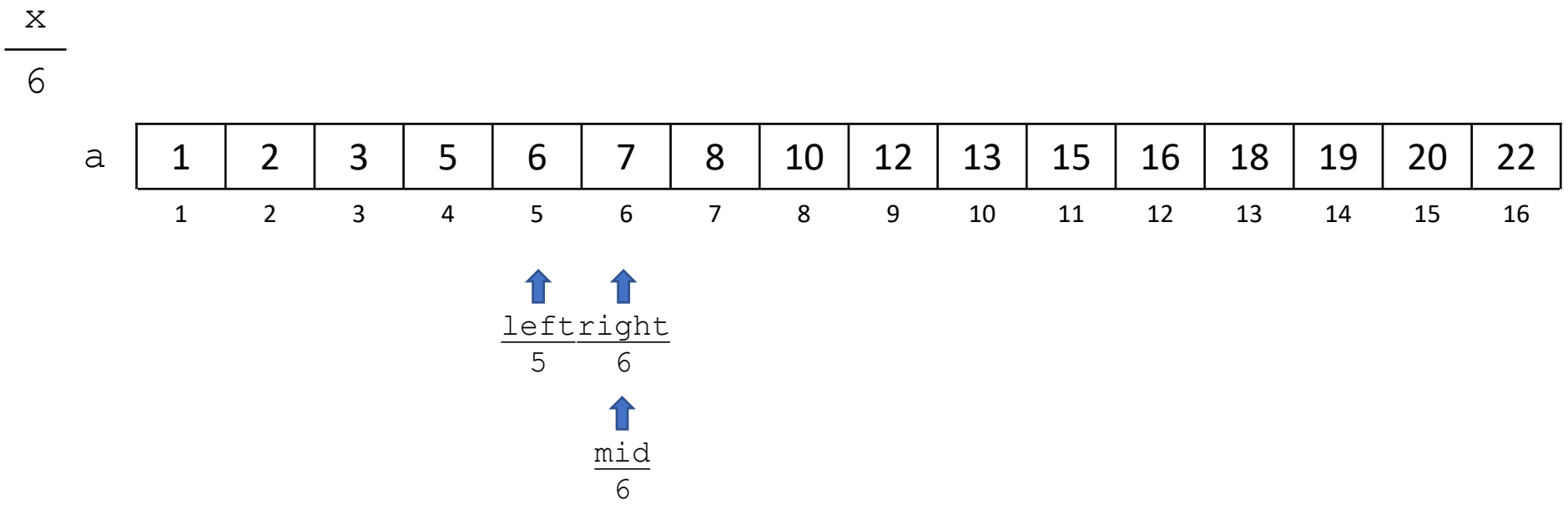
Binary Search

- Example: Find 6 in the ordered list of 16 integers



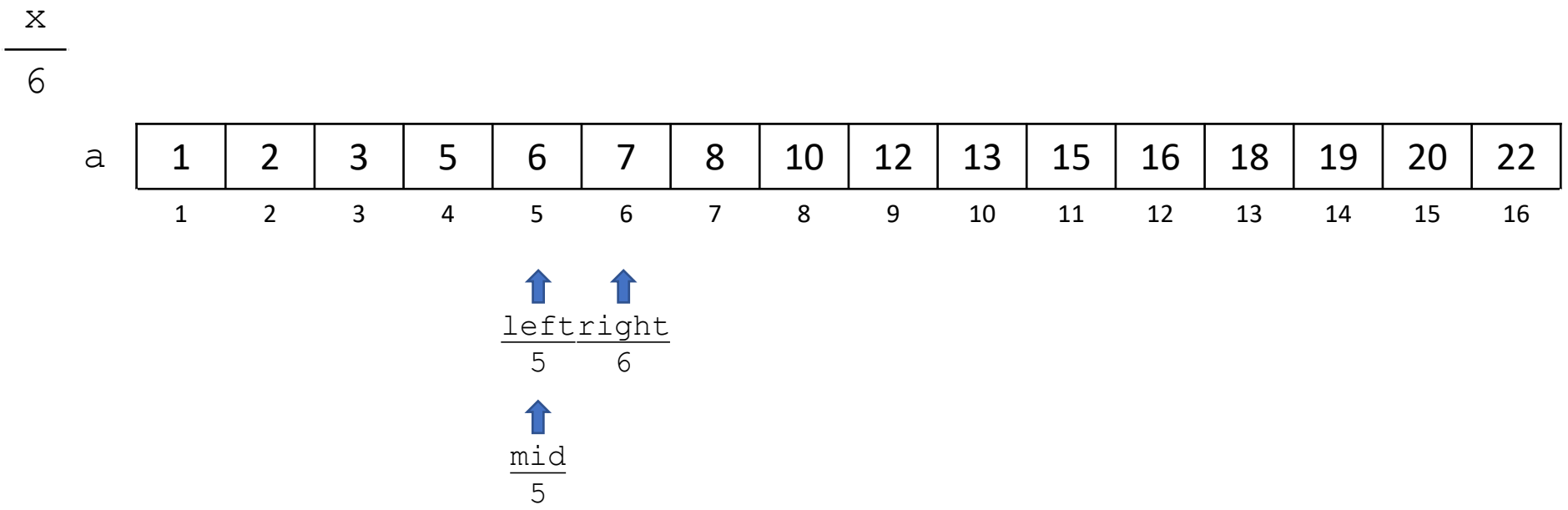
Binary Search

- Example: Find 6 in the ordered list of 16 integers



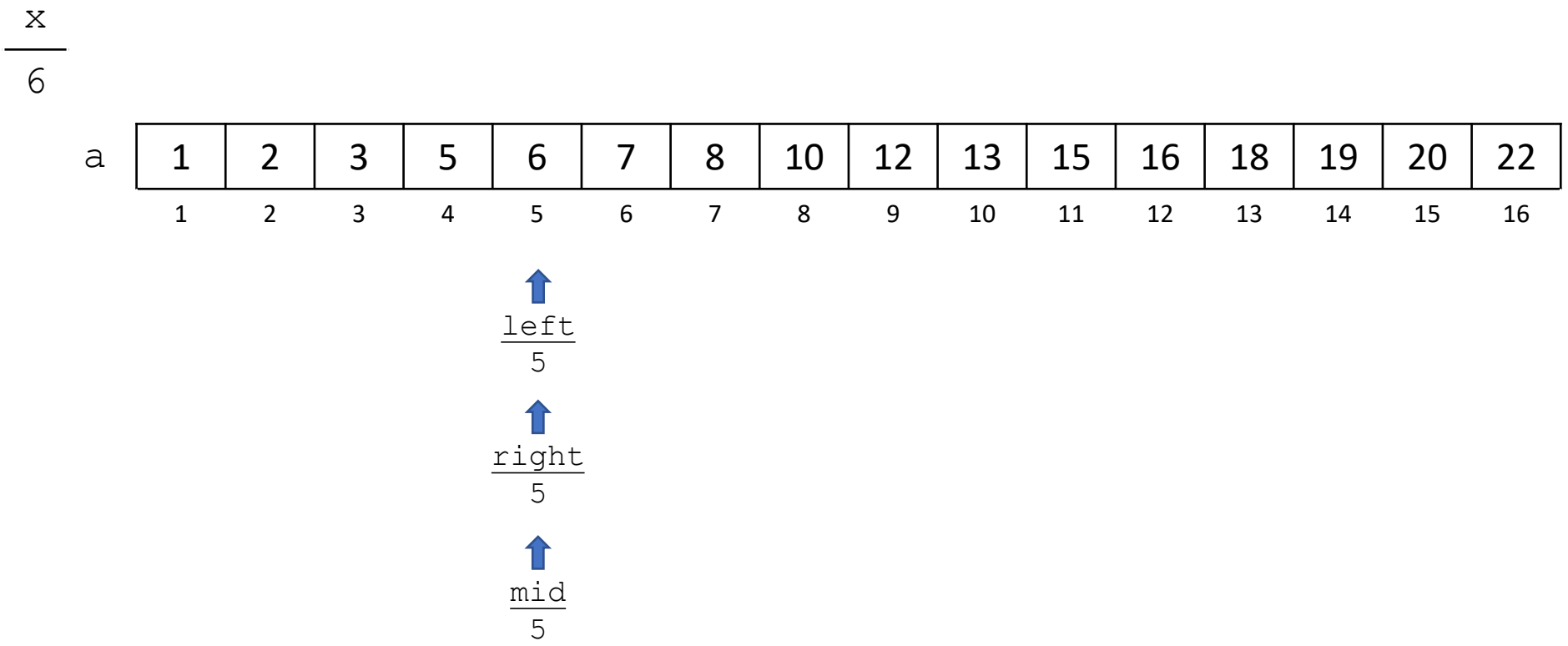
Binary Search

- Example: Find 6 in the ordered list of 16 integers



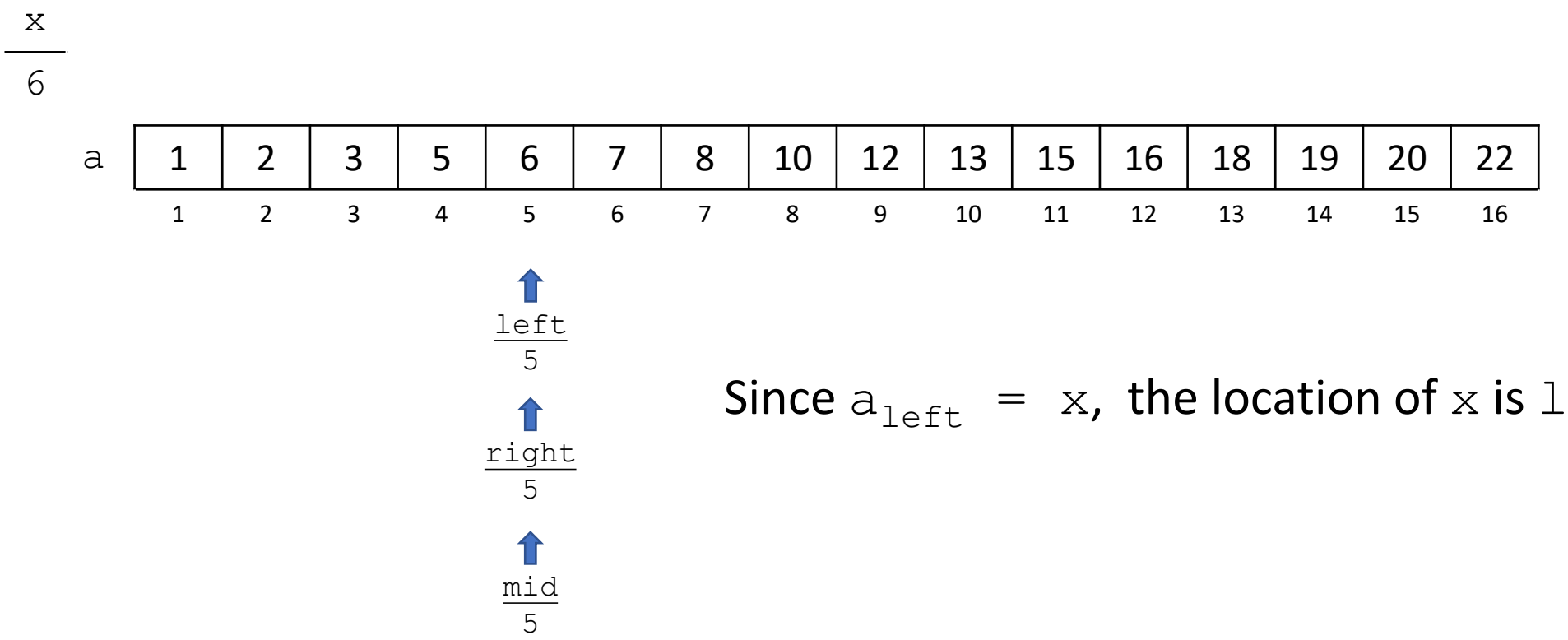
Binary Search

- Example: Find 6 in the ordered list of 16 integers



Binary Search

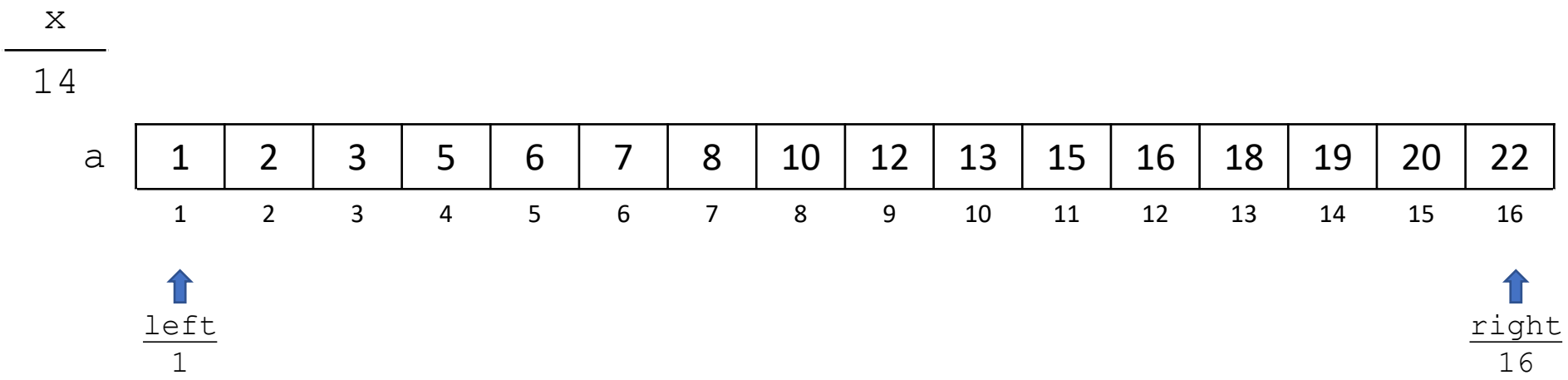
- Example: Find 6 in the ordered list of 16 integers



Since $a_{left} = x$, the location of x is left

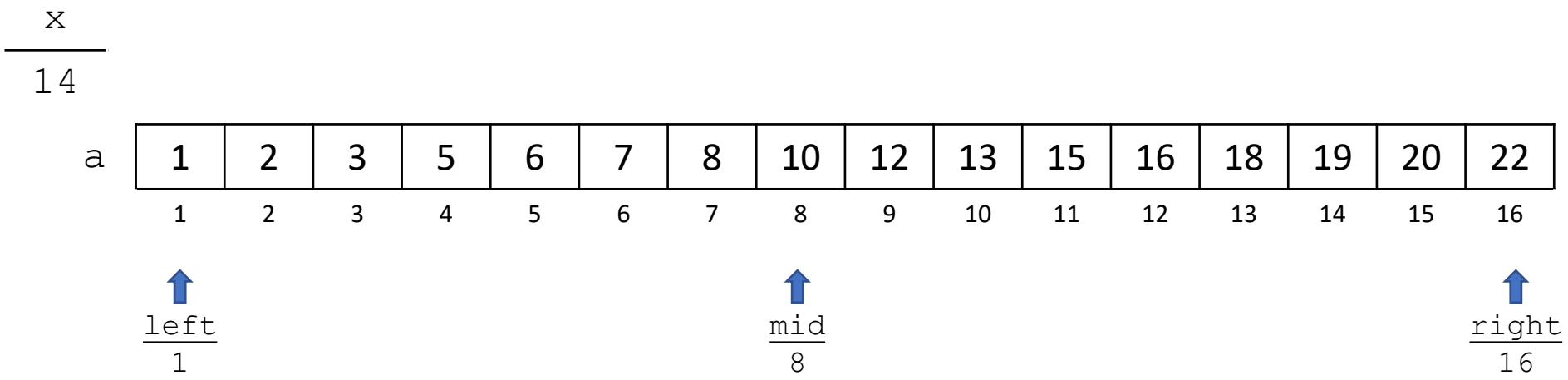
Binary Search

- Another example: Find 14 in the ordered list of 16 integers



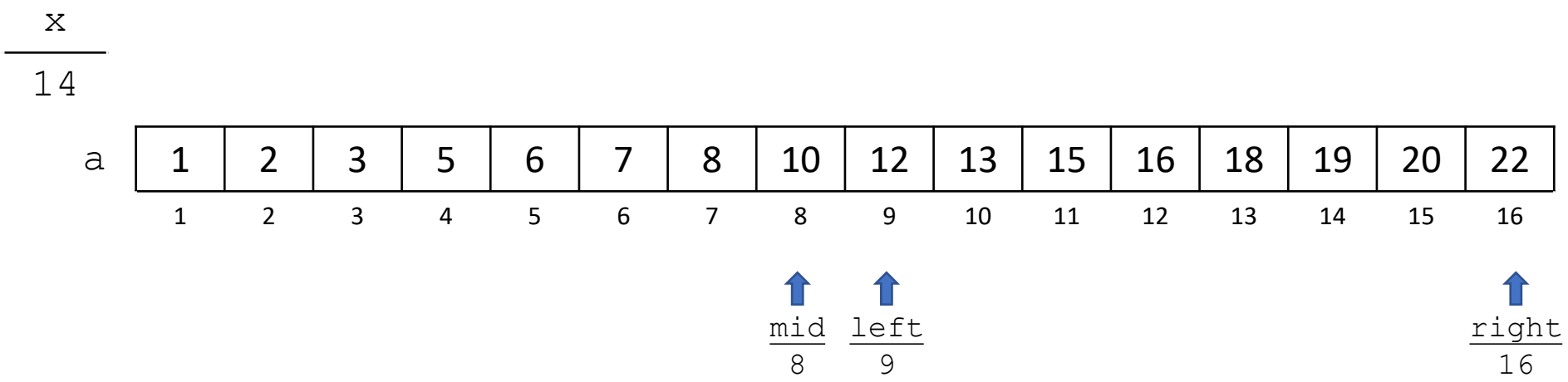
Binary Search

- Another example: Find 14 in the ordered list of 16 integers



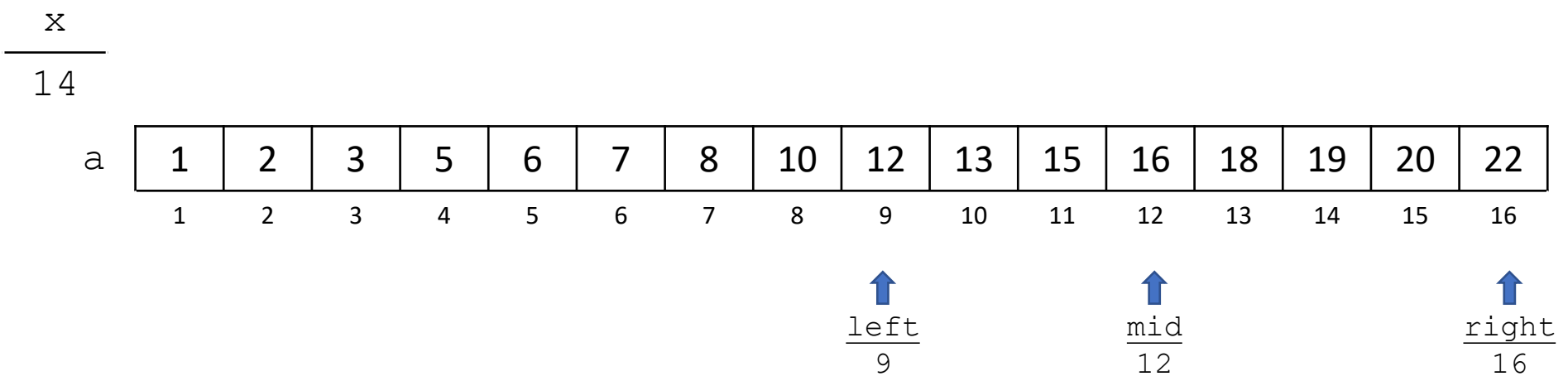
Binary Search

- Another example: Find 14 in the ordered list of 16 integers



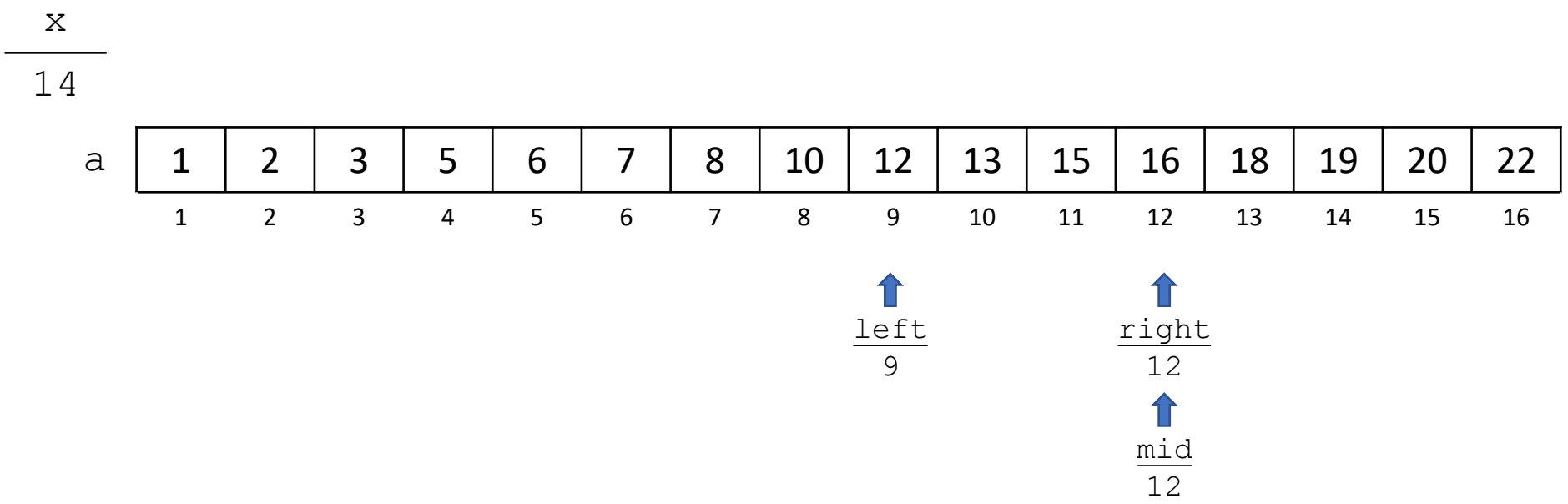
Binary Search

- Another example: Find 14 in the ordered list of 16 integers



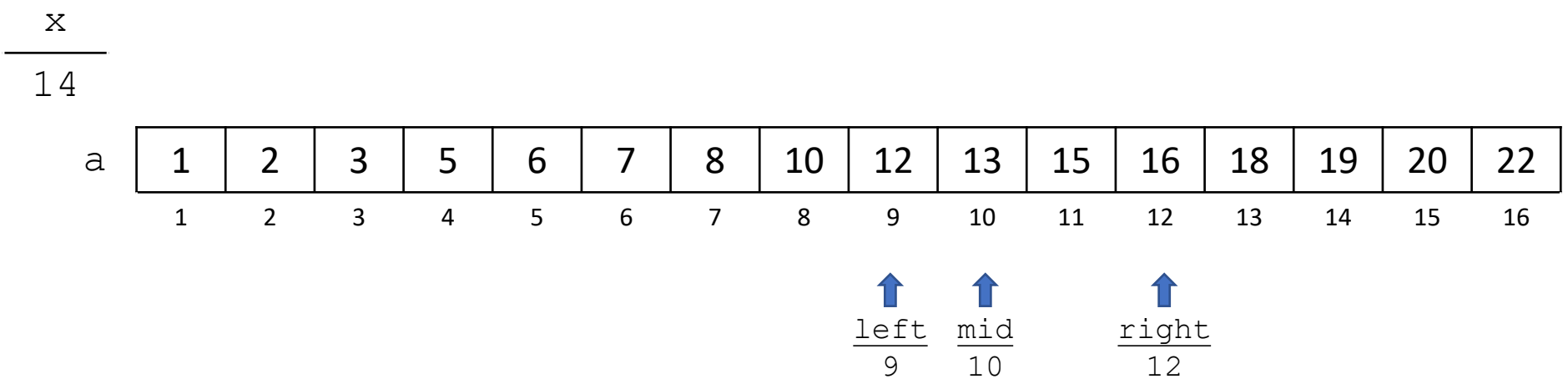
Binary Search

- Another example: Find 14 in the ordered list of 16 integers



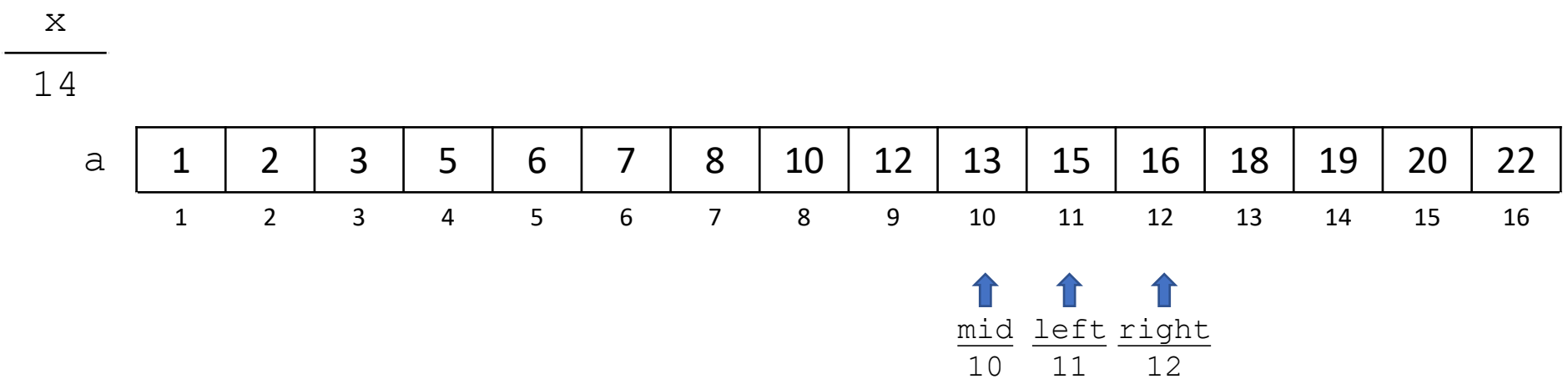
Binary Search

- Another example: Find 14 in the ordered list of 16 integers



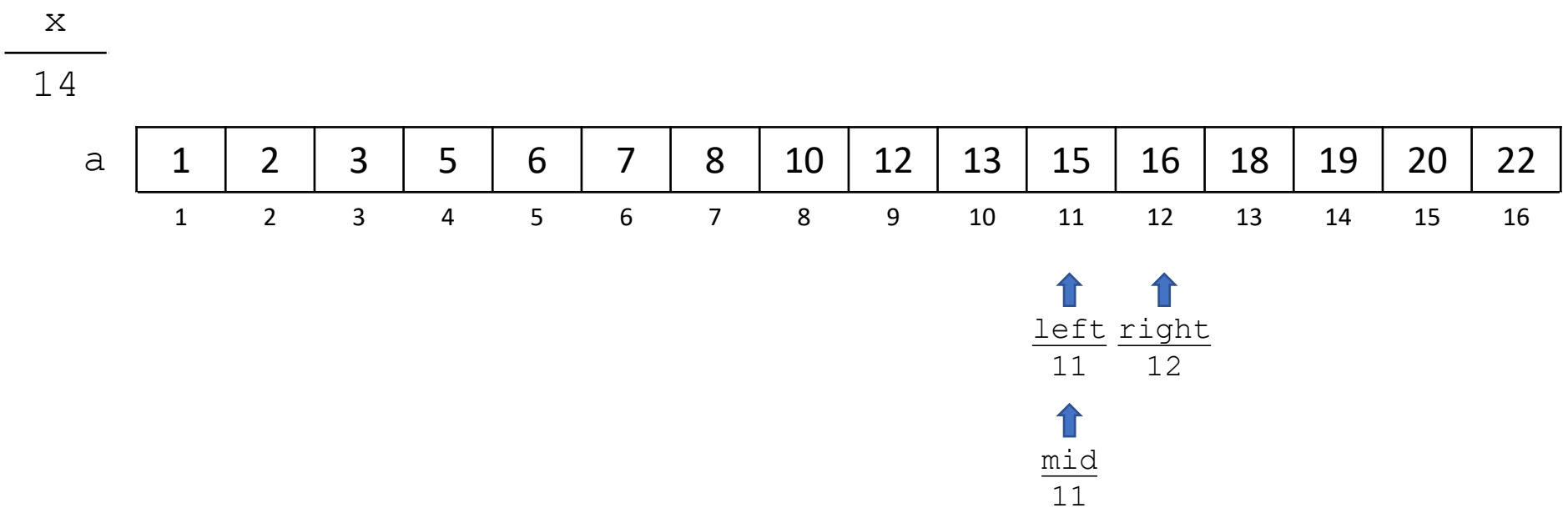
Binary Search

- Another example: Find 14 in the ordered list of 16 integers



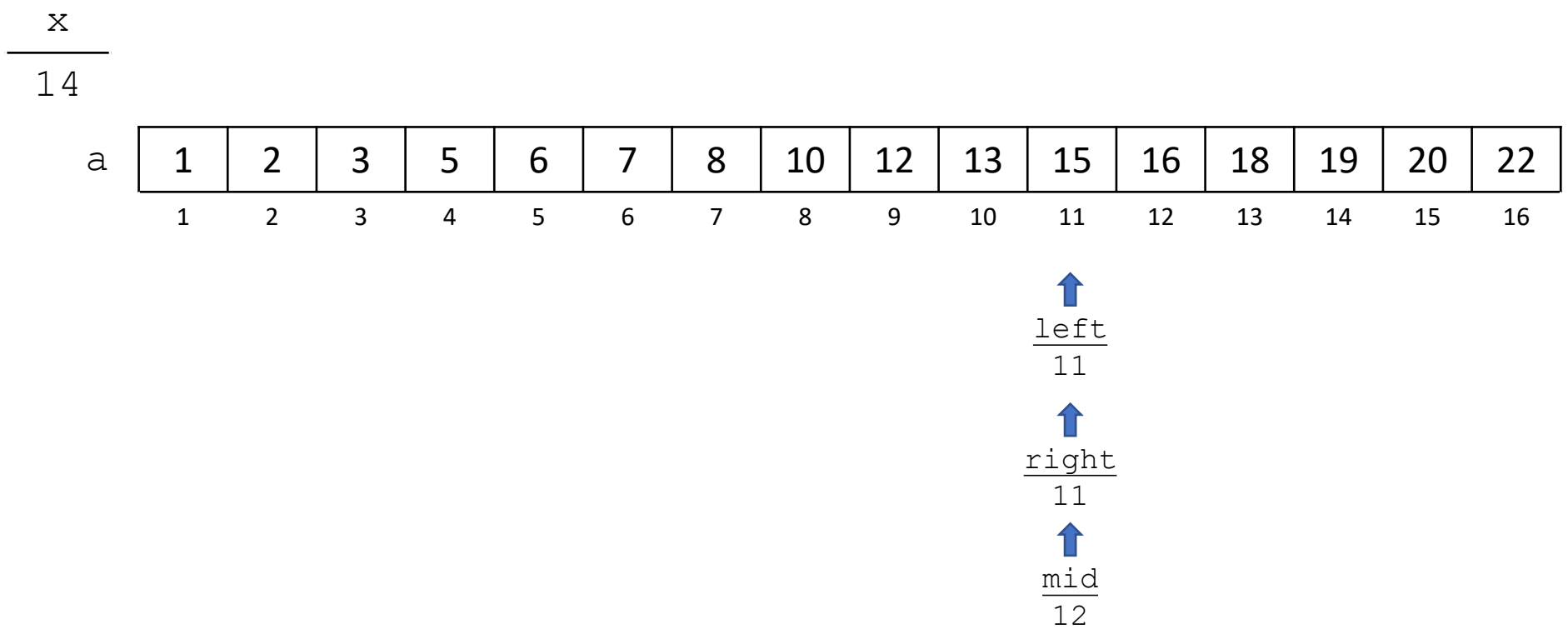
Binary Search

- Another example: Find 14 in the ordered list of 16 integers



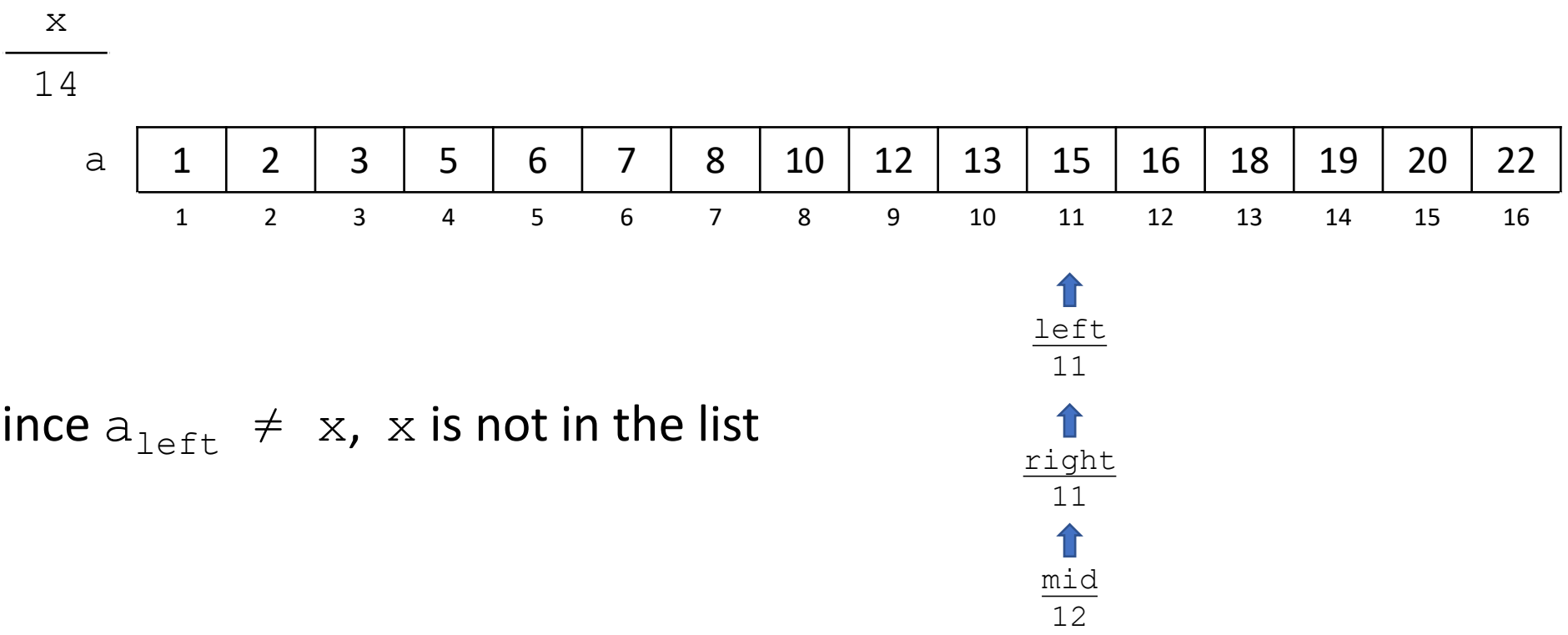
Binary Search

- Another example: Find 14 in the ordered list of 16 integers



Binary Search

- Another example: Find 14 in the ordered list of 16 integers



Since $a_{left} \neq x$, x is not in the list

Binary Search

- The search repeats the following steps
  1. `mid := ⌊(left+right)/2⌋`
  2. If $x \leq a_{mid}$ then `right := mid`
  3. Otherwise, `left := mid+1`

- The search ends when `left = right`
- At which point, if $x=a_{left}$ then `x` was found at index `left`

Binary Search

```
BinarySearch(A, x, left, right)
Input: a sorted sequence of numbers: A=a_1…a_n,
    a number to search for: x
    the boundaries within which to search: left and right
Output: The index of x in A if it is in A, otherwise -1

if (left = right)
  if (x = a_left)
    return left
  else
    return -1
  end-if
end-if

mid := ⌊(left+right)/2⌋
if (x ≤ a_mid)
  right := mid
else
  left := mid+1
end-if
return binarysearch(A, x, left, right)
```

Binary Search Analysis

- If the size of the sequence of numbers, A, is a power of two, i.e. 2^n, then how many recursive calls to binarysearch are made?
- Each recursive call to BinarySearch reduces the portion of sequence A that is to be searched by half.
- Recursive calls are made until `left=right`, i.e., the size of the region of A to be searched is 1
- How many times can 2^n be divided by 2 until the result is 1?
  - n times
  - $\log_2(2^n) = n$

Binary Search Analysis

- Another way to analyze the BinarySearch algorithm is to describe the number of steps required to compute binarysearch when the size of the region of A to be searched is n
- T(1) = 3
- T(n) = 9 + T(n/2)

```
if (left = right)
  if (x = a_left)
    return left
  else
    return -1
  end-if
end-if

mid := ⌊(left+right)/2⌋
if (x ≤ a_mid)
  right := mid
else
  left := mid+1
end-if
return binarysearch(A, x, left, right)
```