Empirical tests of binary search

doubling hypothesis

N	T_N (seconds)	$T_N/T_{N/2}$	*transactions per second*
100,000	1		
200,000	3		
400,000	6	2	67,000
800,000	14	2.35	57,000
1,600,000	33	2.33	48,000
10.28 million	264	2	48,000

Whitelist filter scenario

- Whitelist of size N.
- $10N$ transactions.

```
% java Generator 100000 ...
1 seconds
% java Generator 200000 ...
3 seconds
% java Generator 400000 ...
6 seconds
% java Generator 800000 ...
14 seconds
% java Generator 1600000 ...
33 seconds
```

... = 10 a-z | java TestBS
a-z = abcdefghijklmnopqrstuvwxyz

nearly 50,000 transactions per second, and holding

dobling hypothesis
running time is about a x n^b
with b = lg ratio ==> lg(Tn/Tn/2) ==> lg(2.33)

Validates hypothesis that order of growth is $N \log N$.

Will scale.

Building the Decision Tree

`Binary: yes or no; true or false`

Decision tree is a theoretical tool used to analyze the running time of algorithms. It illustrates the possible executions on inputs.

Use the following implementation of binary search to build the tree for any size array.

Search for every item in the array until the tree is fully built.

```
public static int indexOf (int key, int [] a)
{
   return indexOf (key, a, 0, a.length);
}                                 10


public static int indexOf (int key, int [] a, int lo, int hi)
{

   if ( hi <= lo ) return -1; // key is not present in array a

   int mid = lo + (hi - lo) / 2;  //mid is 5 in first call

   int cmp = a[mid].compareTo(key);
                                mid
   if      ( cmp == 0 )  return a[mid];
   else if ( cmp > 0 )   return indexOf (key, a, lo, mid);
   else return indexOf (key, a, mid+1, hi);
}
```

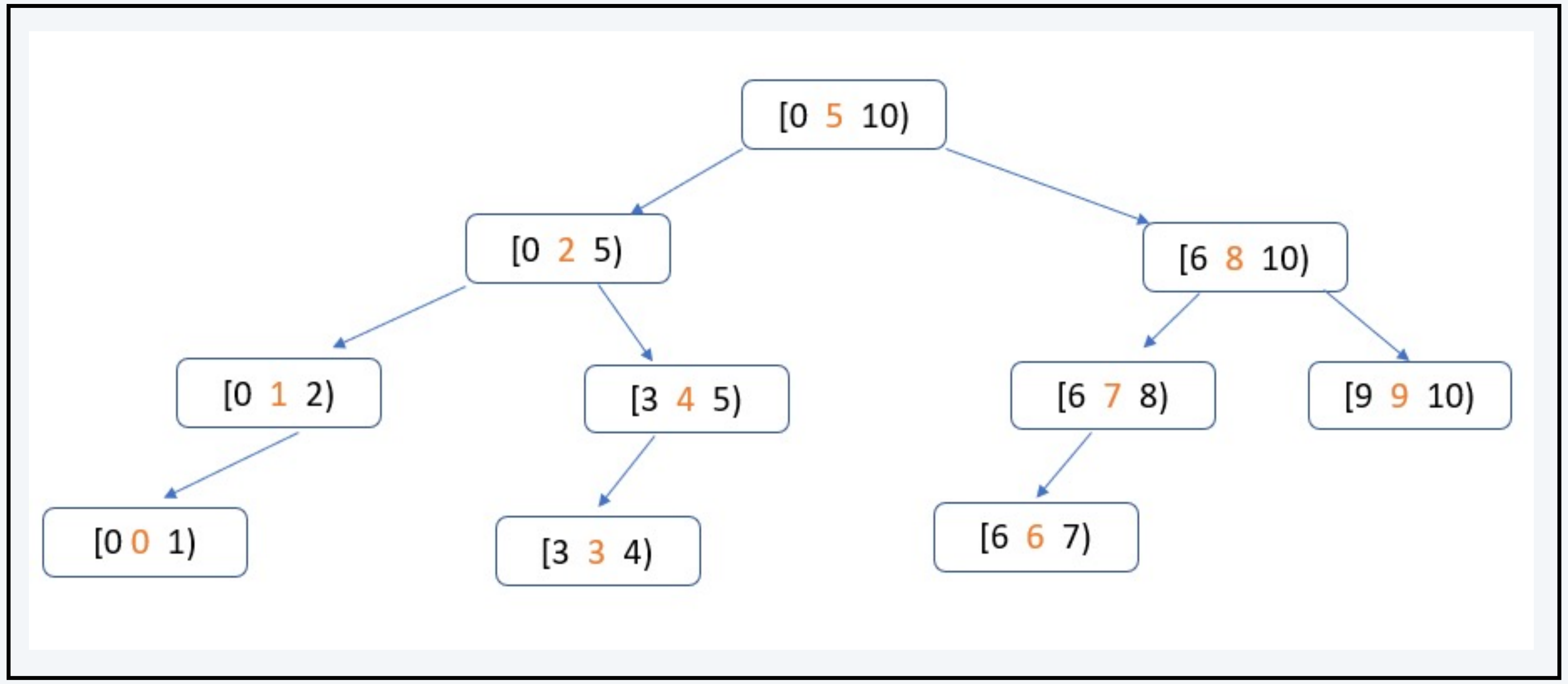
0	1	2	3	4	5	6	7	8	9
11	12	17	19	26	38	45	62	69	83

We'll use an array of size 10 as an example.

Decision Tree based Analysis of Binary Search

LO 4.3

An array of size 10



Mid point in each interval considered marked orange

We assume the interval [lo hi) and left interval [lo mid) and right interval [mid+1, hi) are considered in each step.

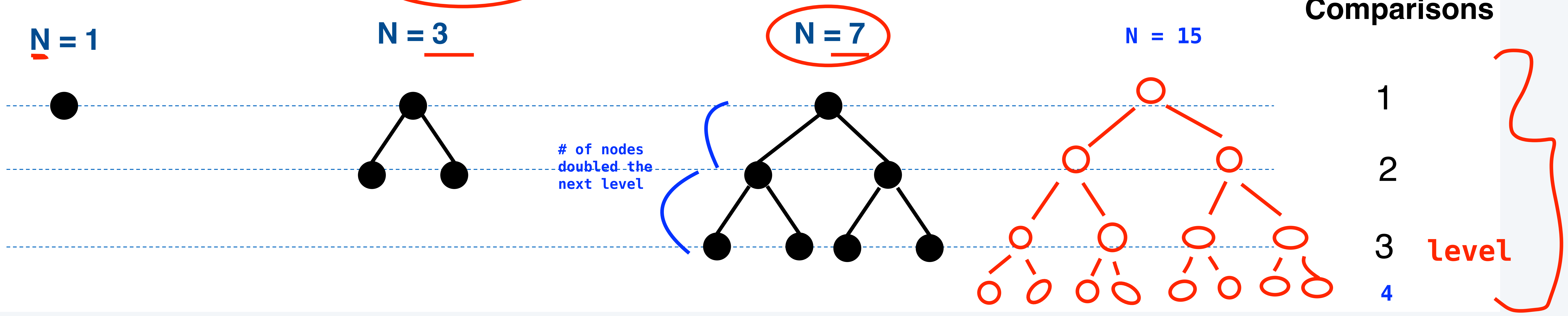
The tree depicts all possible search paths for binary search for keys that may or may not be present.

Q. What is the maximum number of comparisons to find (or not find) an element?
A. Proportional to height of the tree

Decision Tree Height and Maximum Number of Compares

LO 4.3

Full trees of various size



Assume a full tree, array sizes 1, 3, 7, 15, 31, … ← these are all $2^i - 1$ –> total number of nodes of a full tree, where i is the number of levels

The worst-case number of compares for an array of size N is proportional to the height.

~~height~~ level = log (N + 1)

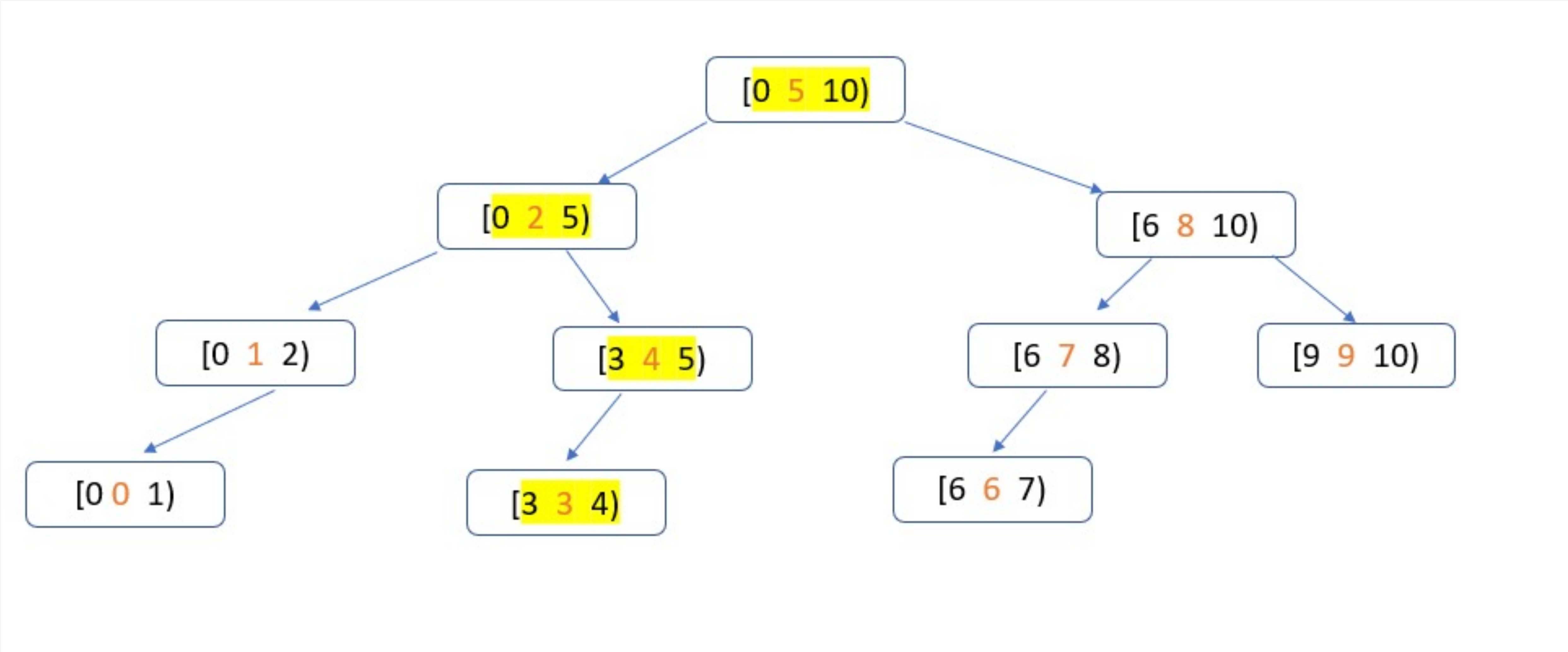
Q. What is the height of a decision tree for an array of size 10?

A. An array of ~~height~~ size 10 has the ~~height~~ level of 4.

Assume a full tree, 10 items do not fit in an array N = 7, so we go to N = 15

~~height~~ level = log (N + 1) = log (15 + 1) = 4

An Example



Searching for Target 19
Found at array index 3

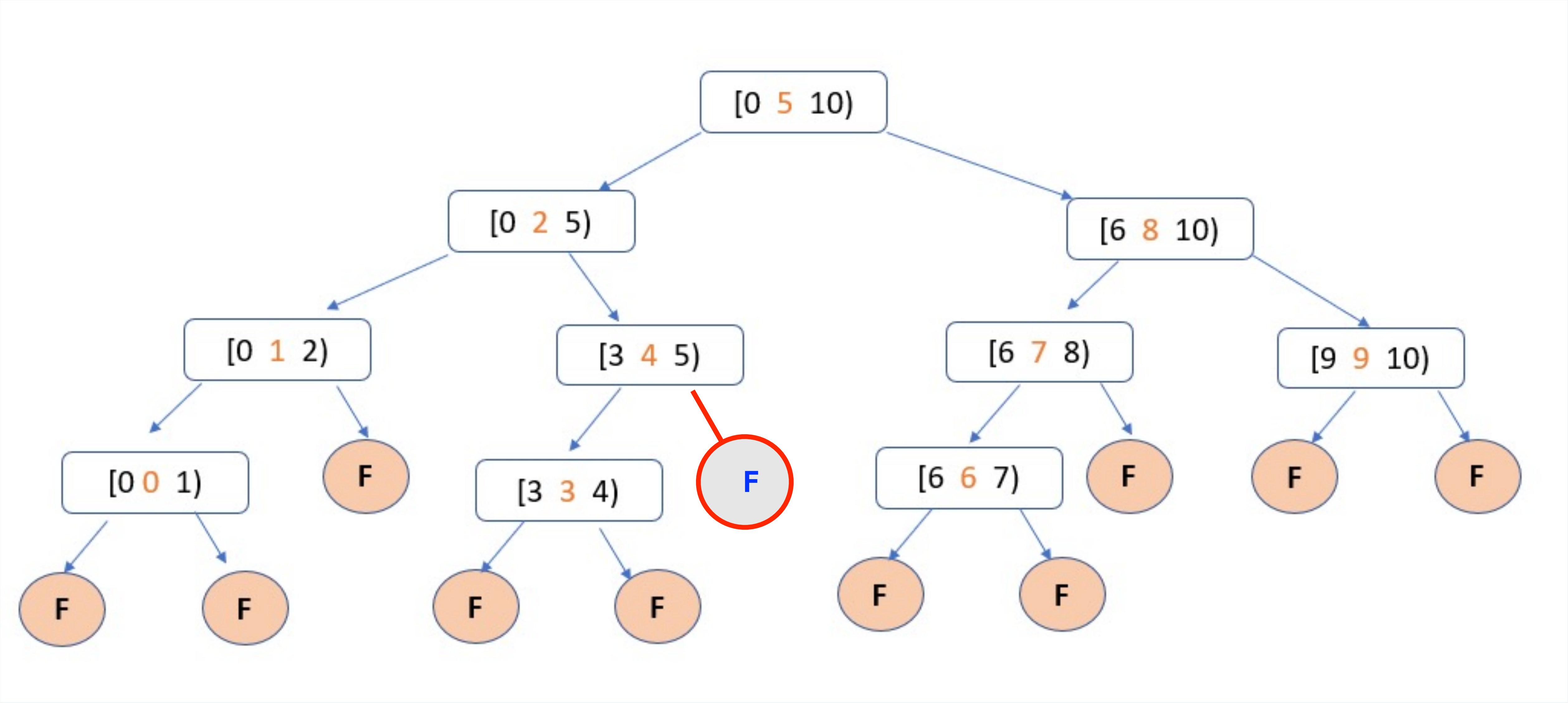
Count the middle points along the search path to find the number of compares.

Q. What is the number of comparisons to find key 19 (at index 3)?

A. 4 compares

Worse Case Comparisons

0	1	2	3	4	5	6	7	8	9
11	12	17	19	26	38	45	62	69	83



Successful Search – index of the search element is an inner node

Unsuccessful Search – occurs at the leaf nodes marked as F

At each node there is ONE comparison to decide which way to go.

Worst-case for successful search

in this example the indices {0, 3, 6} have the longest path of 4 comparisons for successful search.

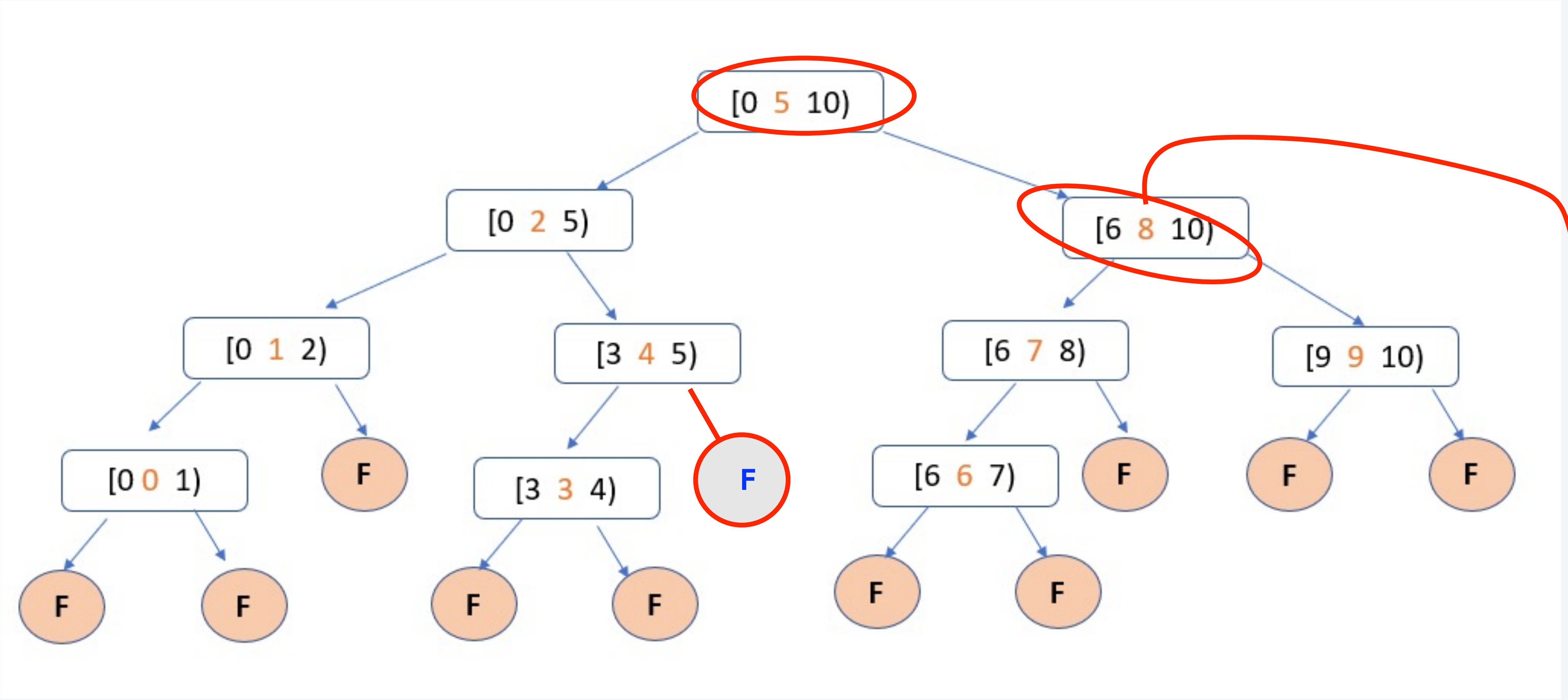
Worst-case for unsuccessful search

items that fail at the nodes that have the worst-case successful search.

Complexity of the search ~ log N

Average Case Comparisons

0	1	2	3	4	5	6	7	8	9
11	12	17	19	26	38	45	62	69	83



Successful Search – average the number of comparisons for successful searches.

Index	#comparisons
5	1
2	2
8	2
1	3
4	3
7	3
9	3
0	4
3	4
6	4
	29 / 10 = 2.9

Average number of compares for success

Unsuccessful Search – if we don't know how many keys are being searched, we can only affirm that the average will be between 3-4.