LINEAR SEARCH!

WHAT IS AN ALGORITHM?

An Algorithm is a set of instructions that are followed in order to solve a problem or complete a task.

ALGORITHMS IN PROGRAMMING

In programming, an algorithm is a well-defined set of instructions or steps that a computer follows to solve a specific problem or perform a task.

WHY IS IT IMPORTANT?

Algorithms provide a structured approach to solving problems. Understanding algorithms will help you succeed in tech interviews and on the job.

QUICK FACTS FOR NEWBIES!

Array = List

Index = the location of a single item in an array

Indexes start at 0

Pseudocode = notation resembling code, but is human readable and used for the purpose of planning

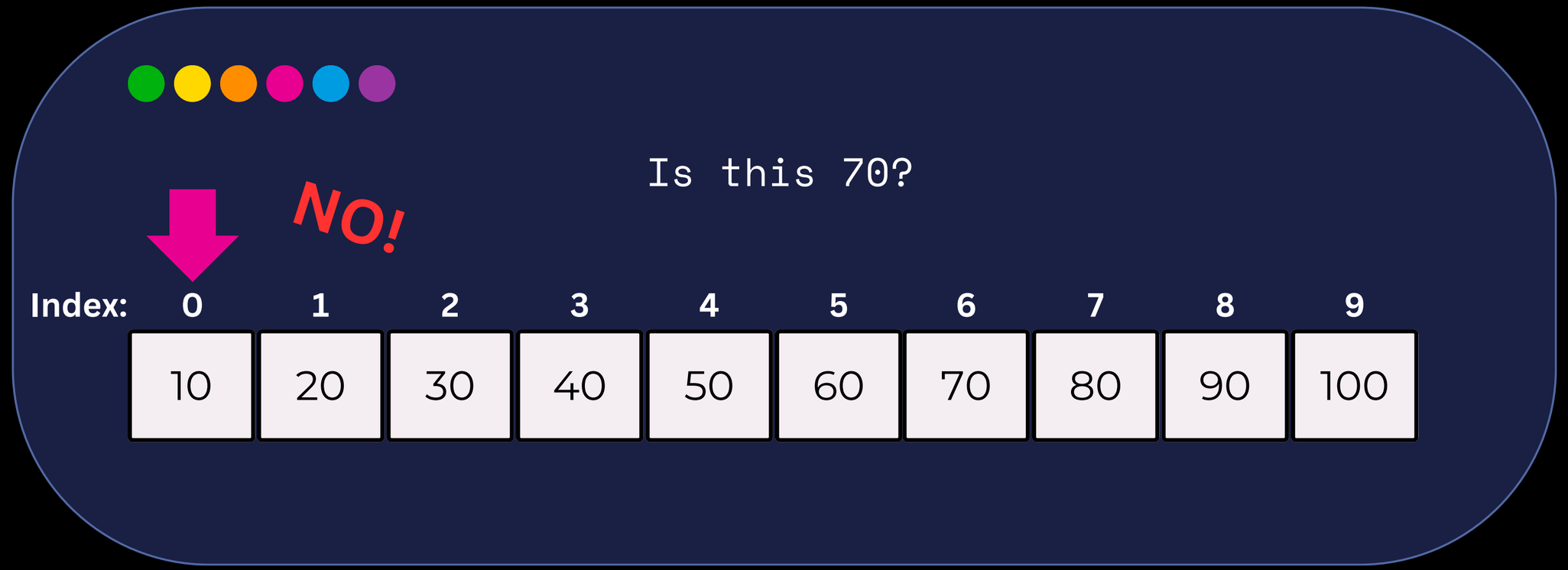
LINEAR SEARCH ALGORITHM

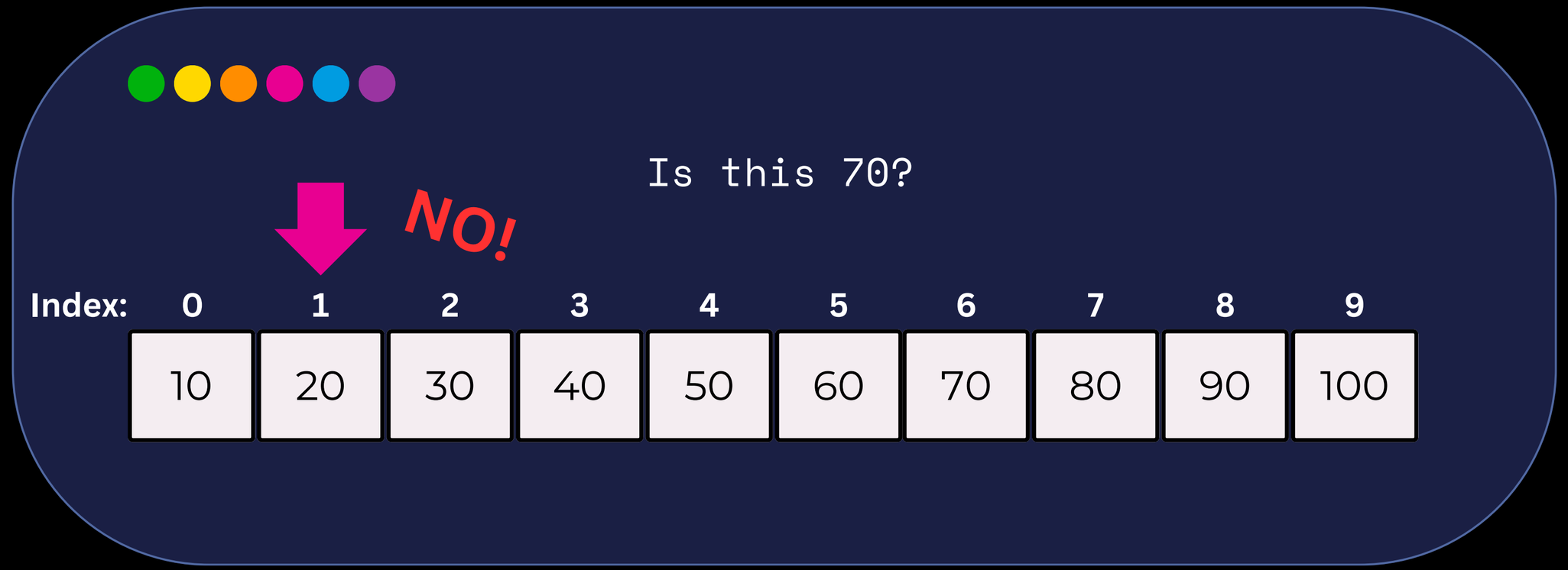
The linear search algorithm iterates over all the elements of an array and checks if the current element is equal to the target element. If it finds any element to be equal, it returns the index

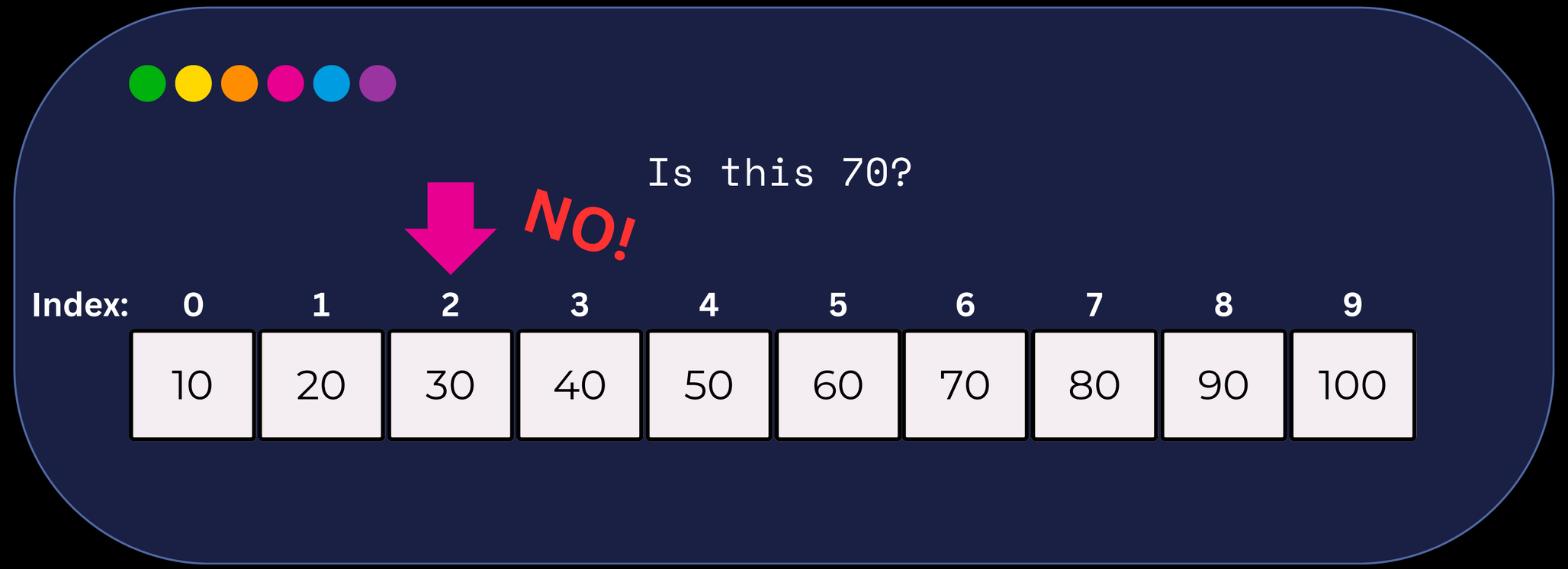
Lets consider an array of integer values:

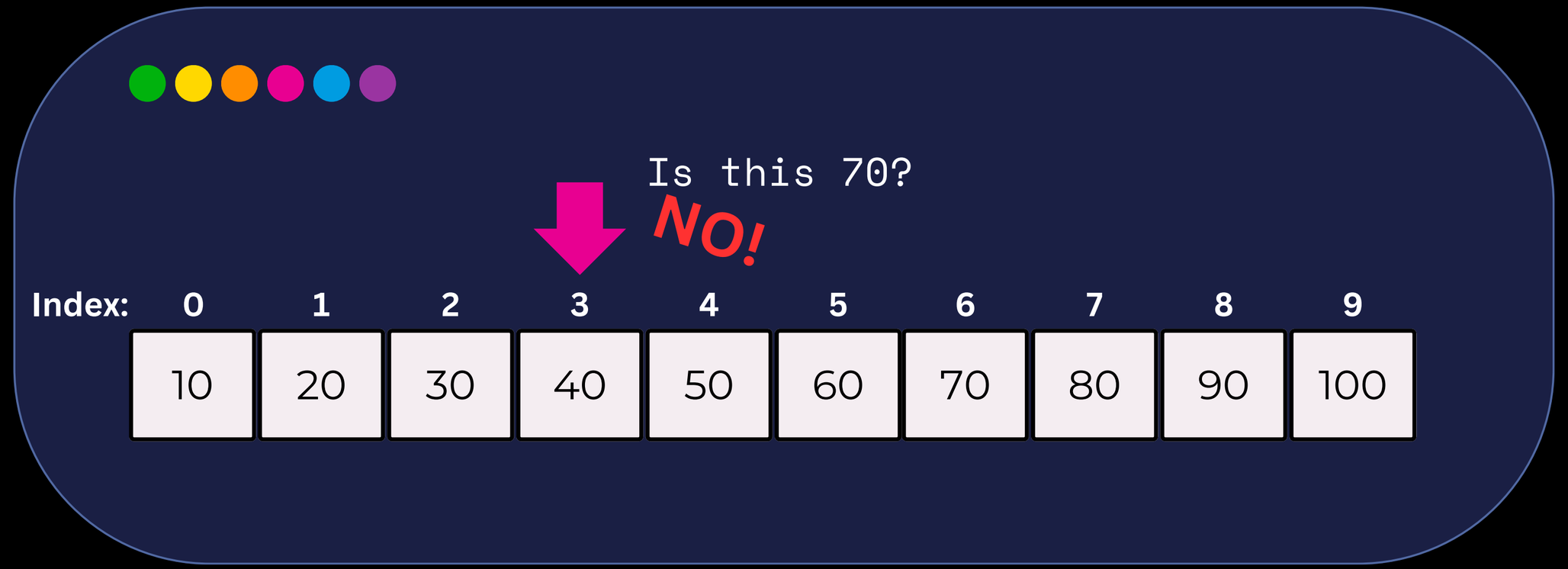
In a Linear Search, we would search the list in order of index, one by one.

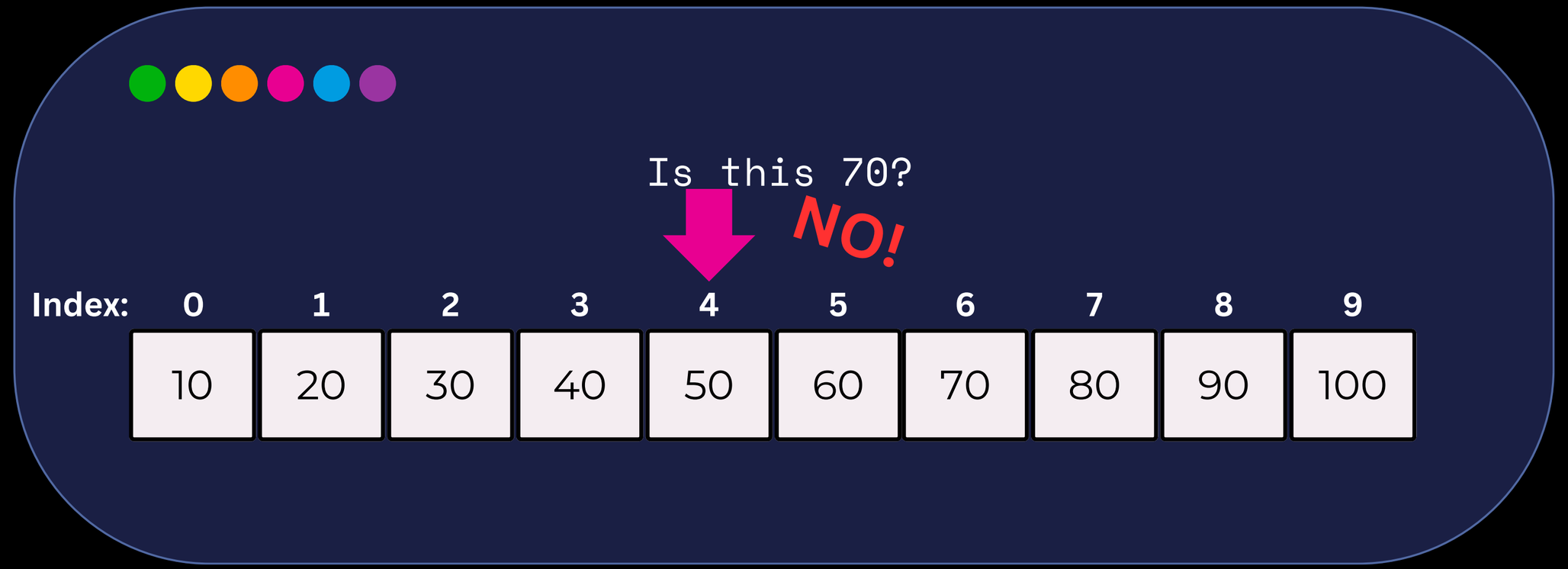
For example! If we were looking for the number 70, in a linear search we would check the first number

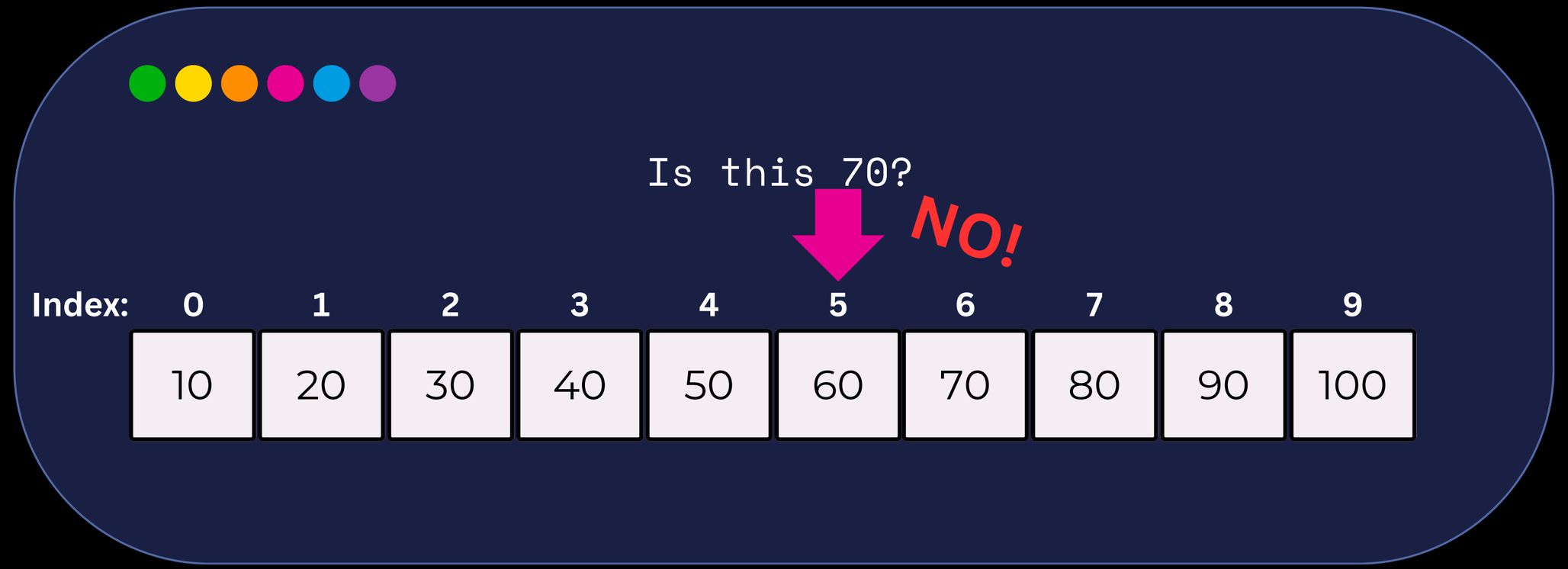

Is this 70?
NO!
Index:
0
1
2
3
4
5
6
7
8
9
10
20
30
40
50
60
70
80
90
100

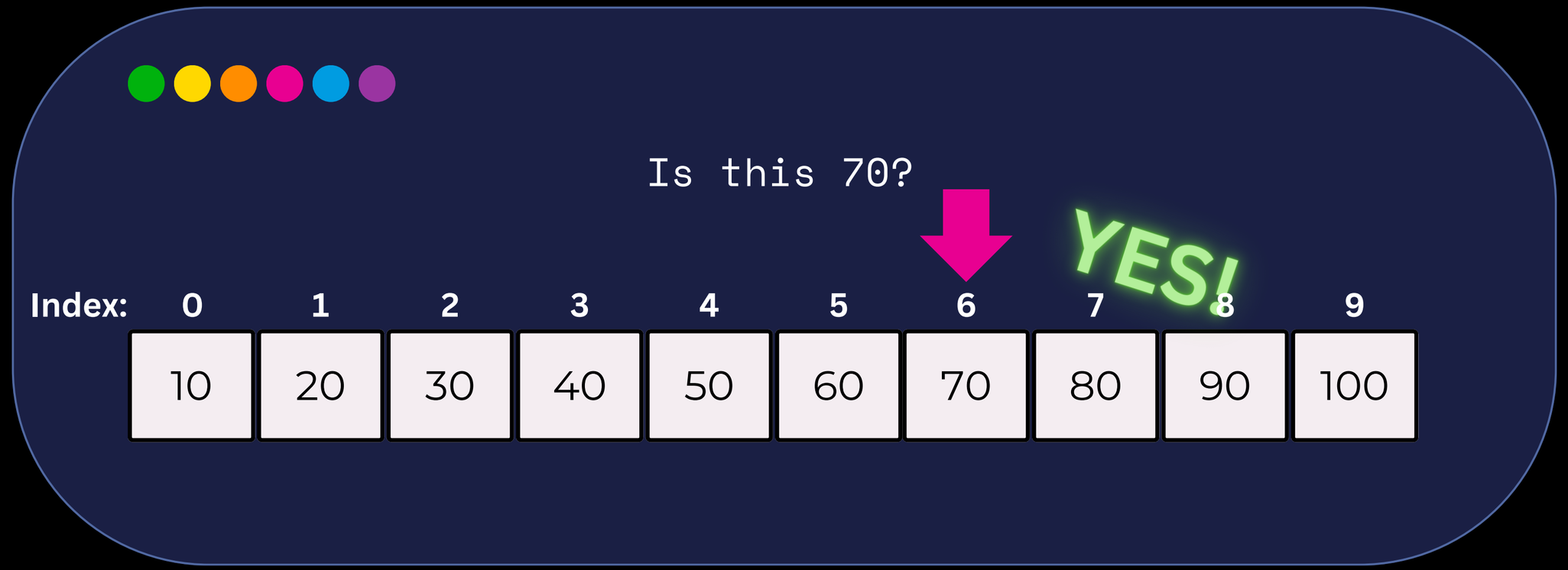

Is this 70?
NO!
Index:
0
1
2
3
4
5
6
7
8
9
10
20
30
40
50
60
70
80
90
100


Is this 70?
NO!
Index:
0
1
2
3
4
5
6
7
8
9
10
20
30
40
50
60
70
80
90
100


Is this 70?
NO!
Index:
0
1
2
3
4
5
6
7
8
9
10
20
30
40
50
60
70
80
90
100


Is this 70?
NO!
Index:
0
1
2
3
4
5
6
7
8
9
10
20
30
40
50
60
70
80
90
100


Is this 70?
NO!
Index:
0
1
2
3
4
5
6
7
8
9
10
20
30
40
50
60
70
80
90
100


Is this 70?
YES!
Index:
0
1
2
3
4
5
6
7
8
9
10
20
30
40
50
60
70
80
90
100

Once 70 is found, the Linear Search returns the index, which here is 6

YES!

THE PSEUDOCODE

```
array = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]

for each number in array
    if number === 70
       return index
    end if
end for
```

EXAMPLES REPLIT! PLEASE GO TO:

HTTPS://REPLIT.COM/@RIKKIEHRHART/GRABABYTE

O NOTATION!

What is O Notation?

- aka “Big O Notation” is a way to describe how efficient an algorithm is as the size of the input grows.
- It tells us how *long* an algorithm might take or how much *work* it might need to do
- Essentially, how many steps or iterations at the *worst*

Common O Notations:

- Linear Time | O(n) - what we’ll cover today
- Logarithmic Time | O(log n) - cover with Binary Search
- Quadratic Time | O(n^2) - cover with Bubble Sort
- Log-Linear Time | O(n log n) - cover with Merge Sort
- Constant Time | O(1) - cover with Hashing

LINEAR TIME - O(N)

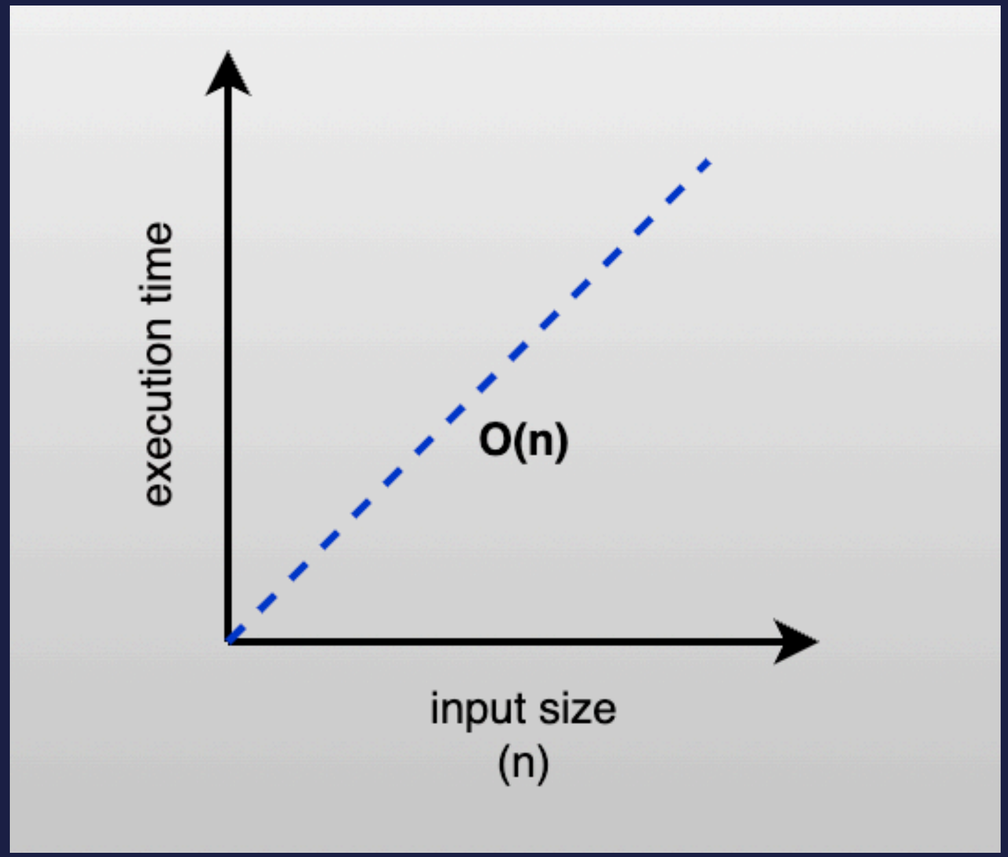
The **O** in **O(n)** stands for “Order Of”, it represents the growth rate or complexity of an algorithm

The **n** in **O(n)** represents the size of the input.

LINEAR TIME - O(N)

- So for Linear Time O Notation, the worst case is the full array
- If there are 10 items in the array, at worst, it would take 10 iterations to find the value
- If there are 100 items in the array, at worst, it would take 100 iterations to find the value
- If there are 1,000 items in the array, at worst, it would take 1,000 iterations to find the value

LINEAR TIME - O(N)



Essentially, as the list gets bigger, the time it takes to complete (at its worst) gets longer.

This is why there are many other algorithms and O Notations - for improved efficiency!

UP NEXT

Feb 5 - Binary Search
Feb 12 - Bubble Sort
Feb 19 - Selection Sort
Feb 26 - Insertion Sort
Mar 5 - Merge Sort
Mar 12 - Quick Sort
SPRING BREAK!
Mar 26 - Breadth-First Search (BFS)
Apr 2 - Depth-First Search (DFS)
Apr 9 Hashing
Apr 16 - Dijkstra's Algorithm
Apr 23 - Dynamic Programming (Knapsack Problem)
Apr 30 - Union-Find
May 7 - Kruskal's Algorithm
May 14 - Prim's Algorithm

Questions? - rikki.ehrhart@g.ausitncc.edu

If you'd like the opportunity to run a Grab a Byte algorithm workshop, please let me know!