# Algoexpert

## What are data structures?

A way to organize and manage data

Data to manipulate, relationship between data

Manipulate by applying certain functions or operations on them

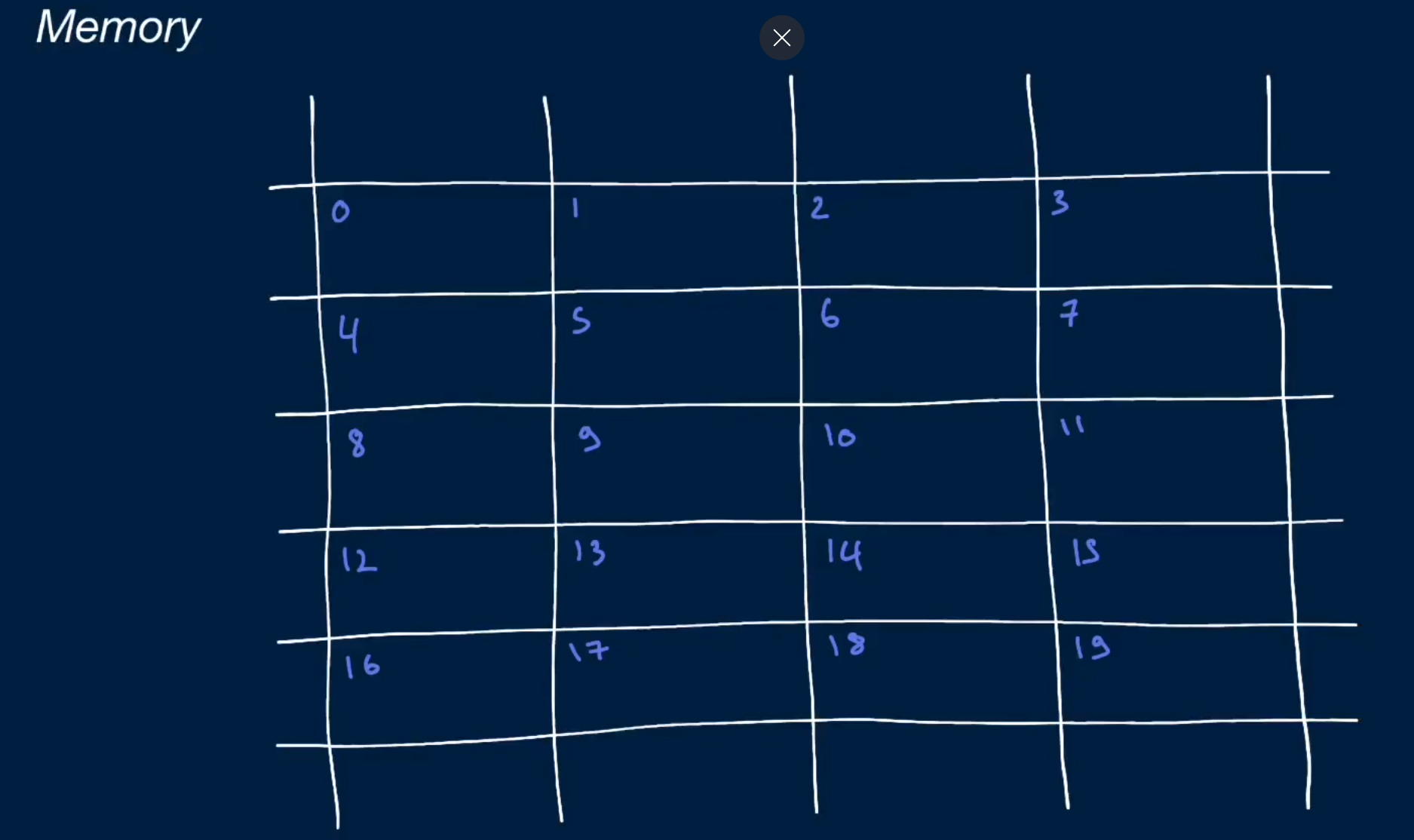
## What are complexity analysis?

To understand data structure

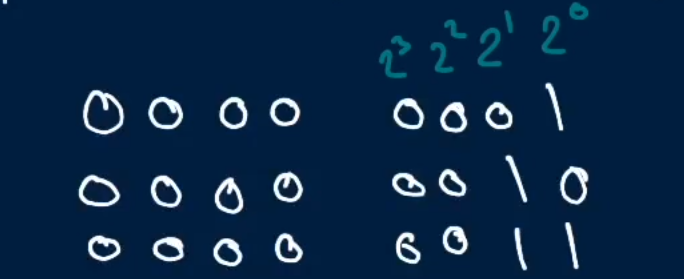
What makes a solution better than the other?

1. Have better complexity?
2. Time and space complexity
3. How fast an algorithm run and how much memory it takes up

## Memory



* The picture is canvas, where we got 20 slots, bounded canvas = finite number of memory slots
* If my data is going to take up more than 1 memory slots, its going to store it in a back2back memory slot
* Memory = bits = 0 and 1, 00101
* One memory slot can hold 8 bits = byte

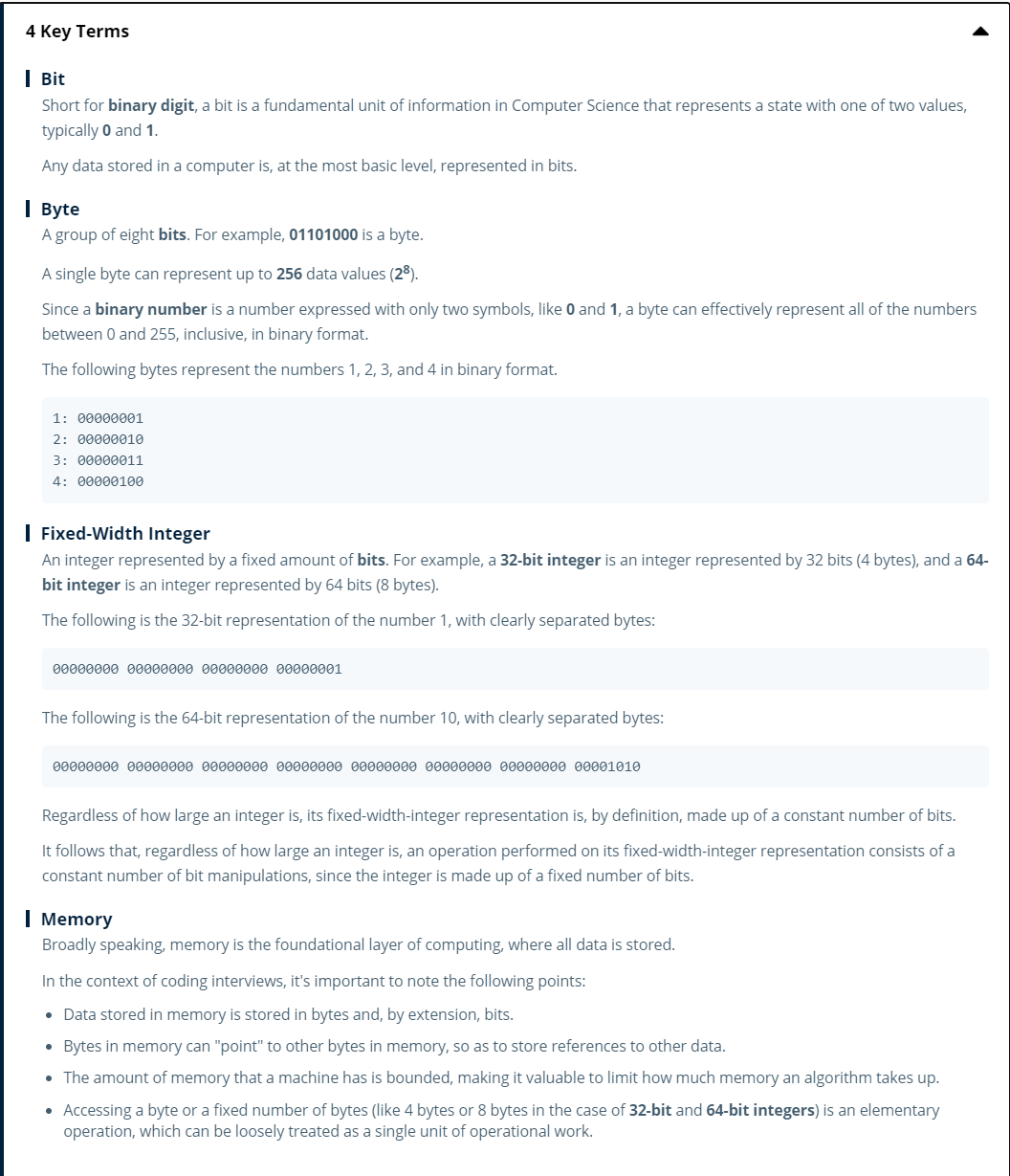


Any piece of data can be transformed into binary/base2 data

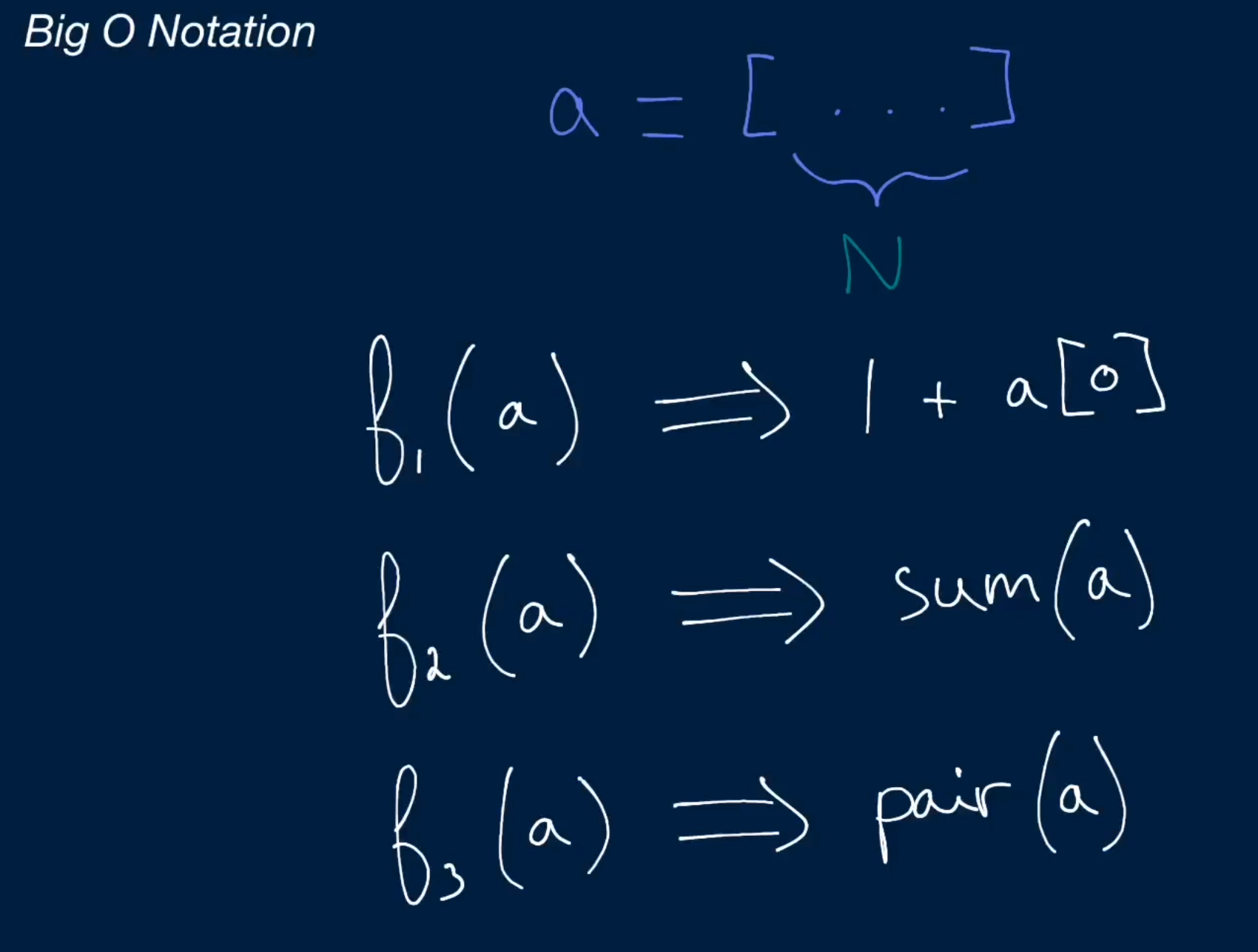
Int = 32 bit int = will always take up 4 memory slots and will always be a constant

Long = 64 bit int = will always take up 8 memory slots and will always be a constant number of memory slots

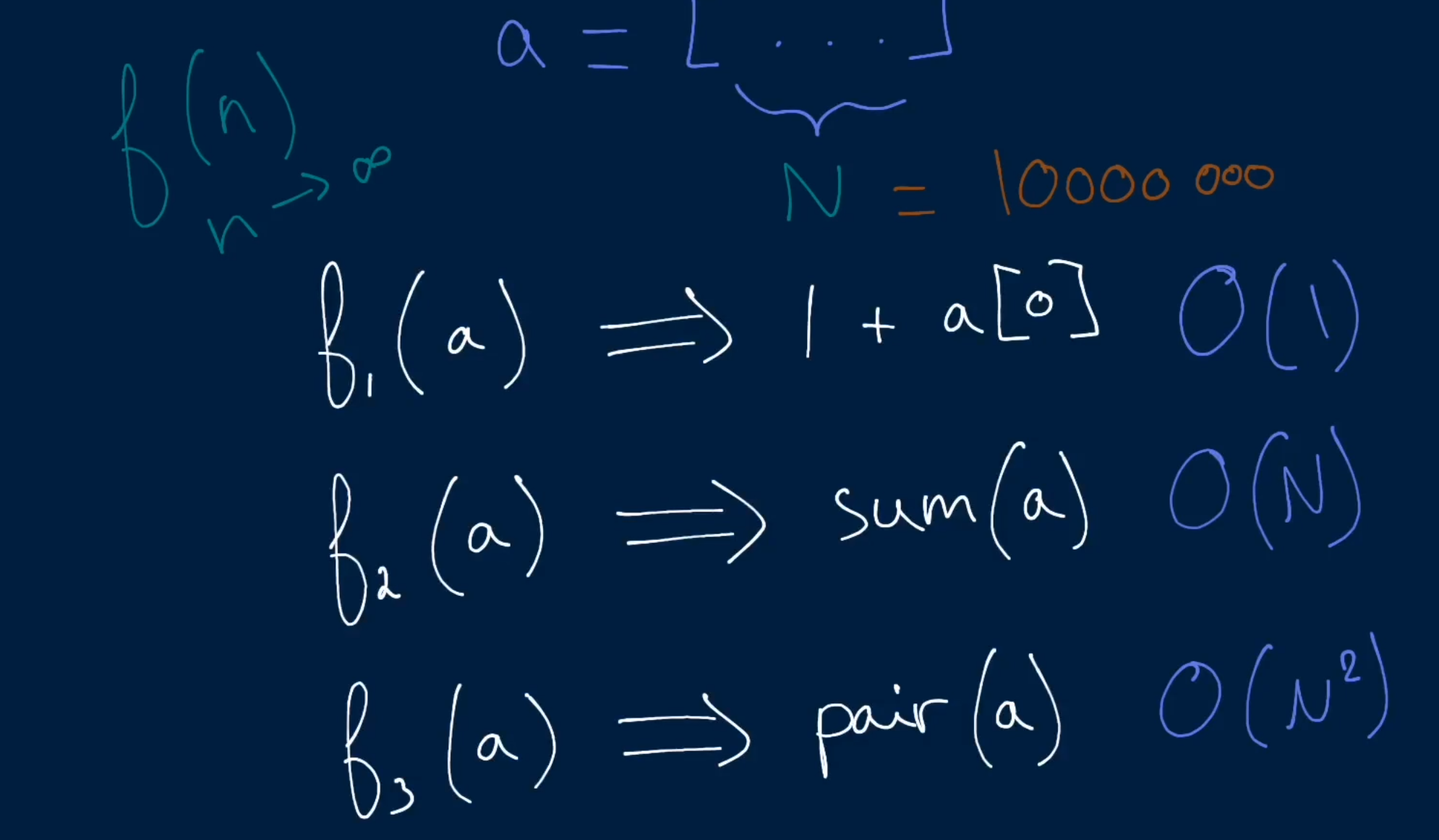
When you’re storing an integer in memory, that interger is always going to be fixed width (64bit., 32 nit. 16 bit, 8 bit etc)

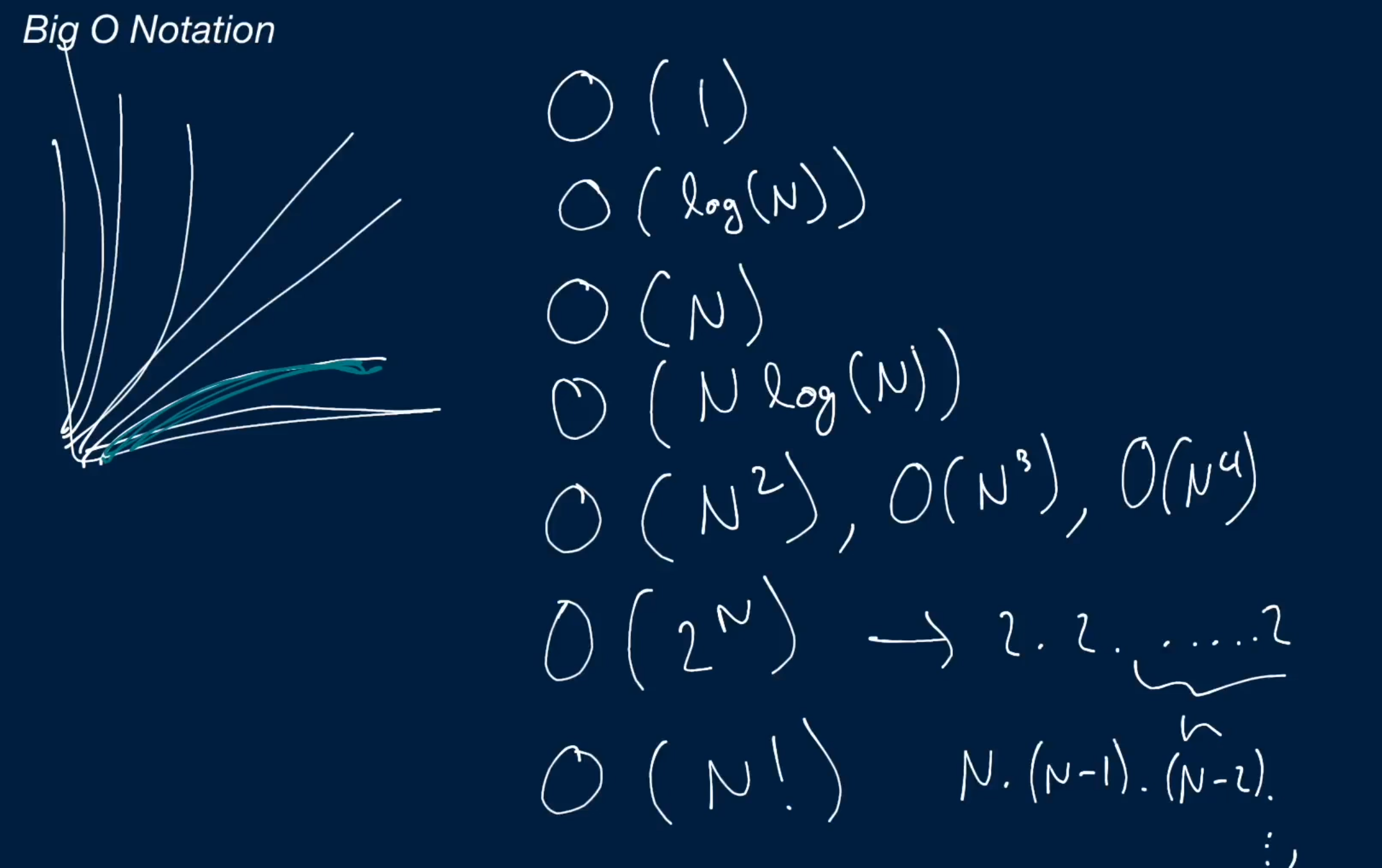


## Big O notation



* How we measure complexity
* How we describe complexity
* Determine complexity of the algorithm





Usually in coding interviews these refer to worst case scenarios

1. Sorting algorithm
2. Quick sort algorithm

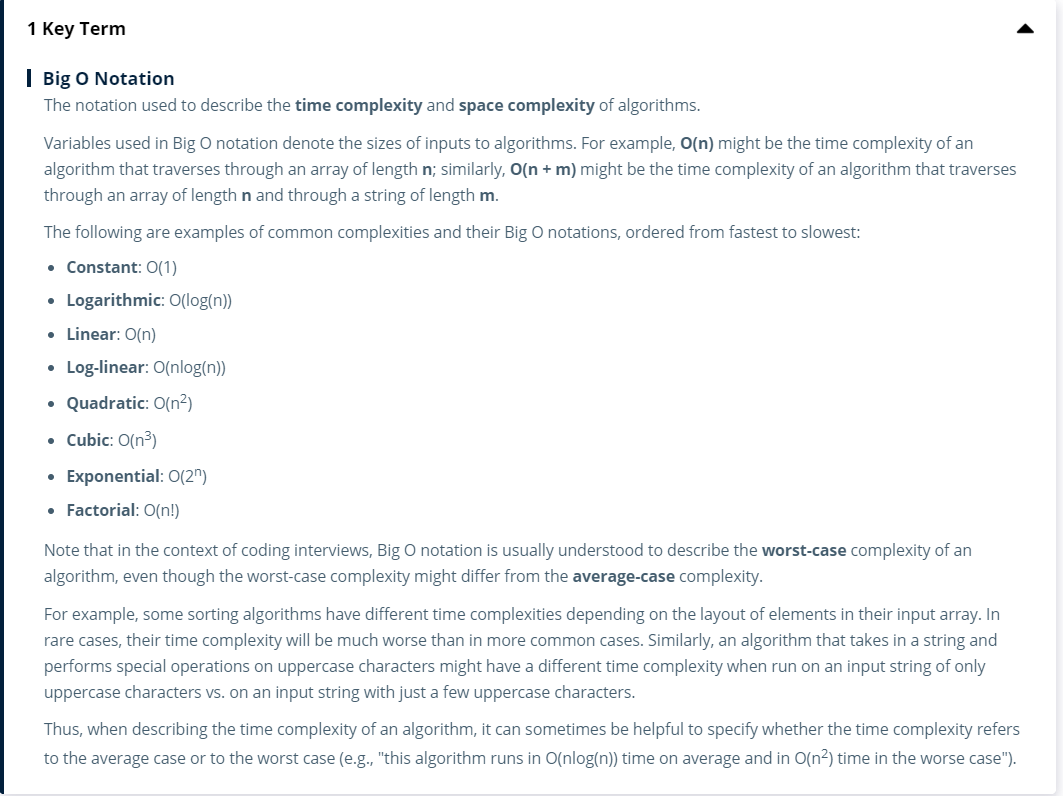
Examples

Algorithm that does elementary operations (sums up numbers, declare variables, declare an array of length)

* O(1)

Algorithm that only traverse through an array

* O (N)



## Logarithm

Graphical user interface, application, Word

Description automatically generated

How to apply to complexity analysis and what it really means

1. Math behind log

A picture containing text, blackboard

Description automatically generated

In comp sci, we’re always only going to be using base 2

A picture containing text, blackboard

Description automatically generated

As the input doubles, the number of things we’re doing in the algorithm only increases by one

2? = N 🡨🡪 log(N) = ?

220 = 1 million

230 = 1 billion

For log, N would be the input. Log(N) favours large input

Examples in coding interviews/ algorithm analysis

Log(N) complexity

A blue screen with numbers

Description automatically generated with low confidence

Array of length 8

A picture containing text, blackboard, night sky

Description automatically generated

Shape

Description automatically generated with low confidence

1. Intuition behind log
   1. How it relates to complexity analysis

## Arrays

Graphical user interface, text, application

Description automatically generated

Array are called list in Python

Calendar

Description automatically generated

Static and dynamic array

A picture containing text, blackboard

Description automatically generated

Amortized analysis

* Inserting an element to the end of the array

Popping = removing the last element of the array

Can also pop element at the beginning of the array

Linked list is for popping the first element

Queue is a library for popping the first element

## Linked list

Diagram

Description automatically generated

Linked list vs array in terms of performance ?

Need to see code examples

## Hash tables

Graphical user interface, text, application

Description automatically generated

A key value stores

A data structure where I can store every key and values

You can access a value given a key

Is it basically an object

Hash table are built on top of an array and it is also a linked list

Modulo operator %

Collision in hash tables

Popular hash table implementations 🡪 fancy and powerful that minimize number of collisions

Hashtable resizing

# Systems design