

This is CS50





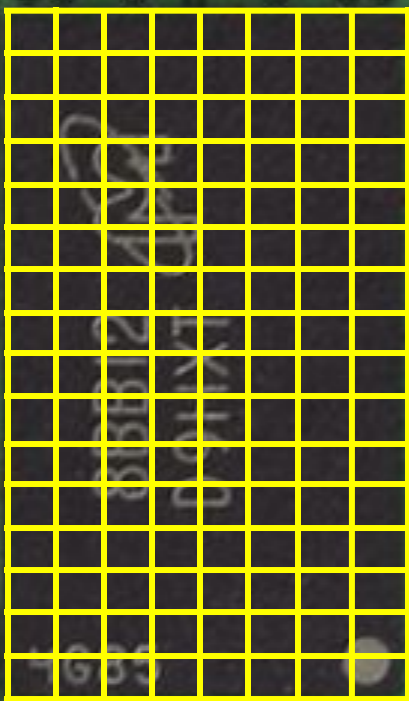
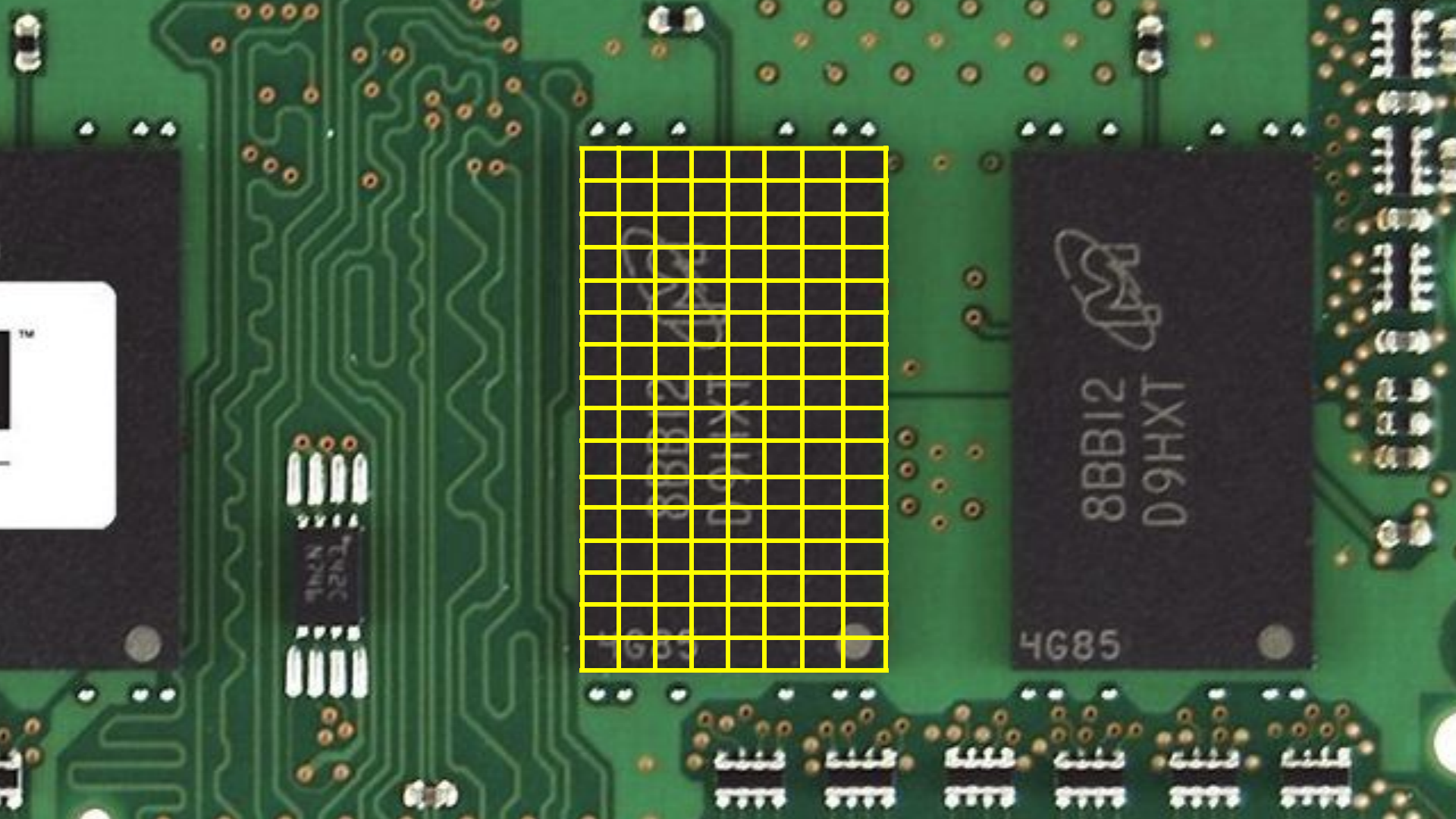
8BB12
D9HXT

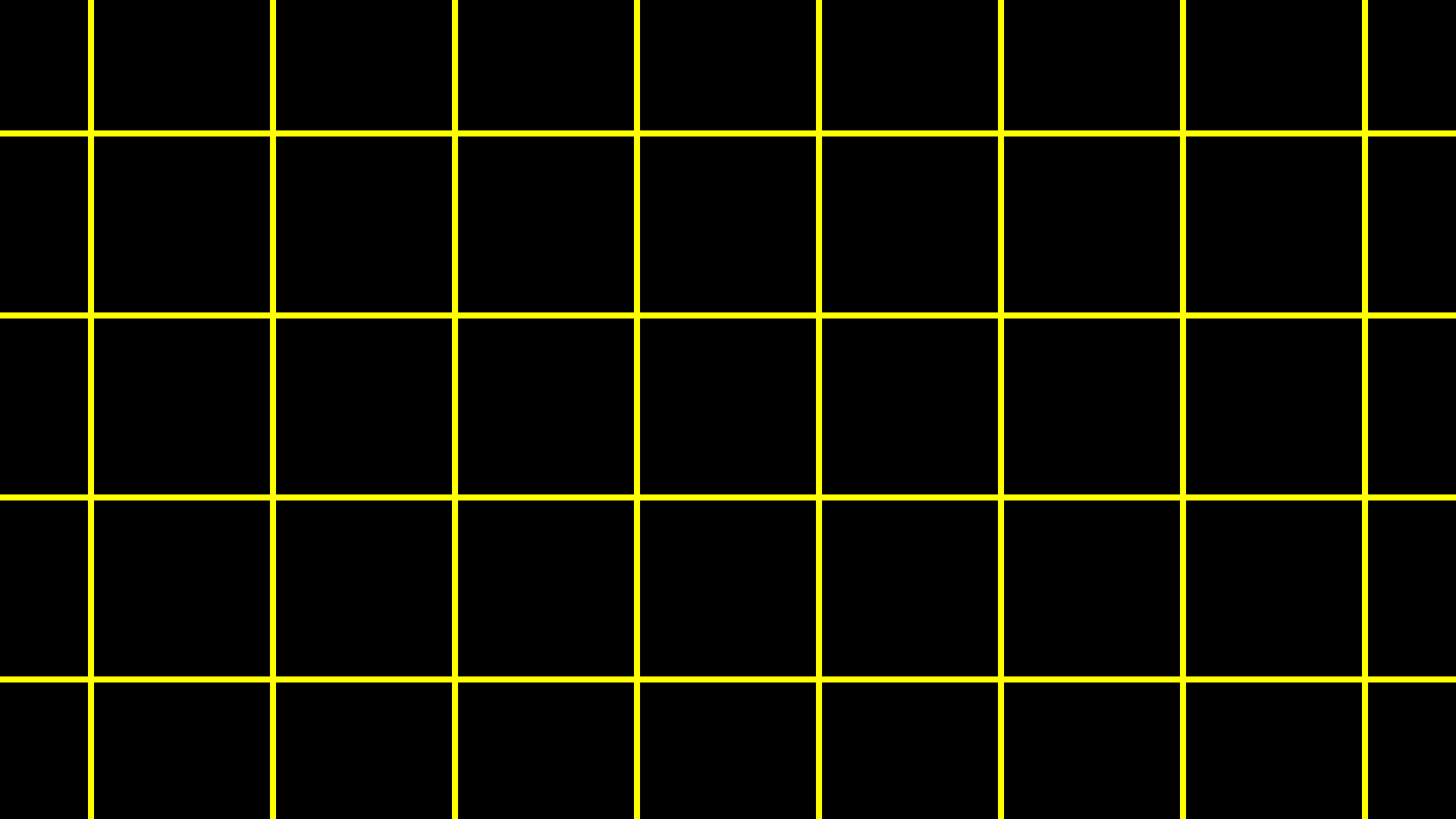
4G85



8BB12
D9HXT

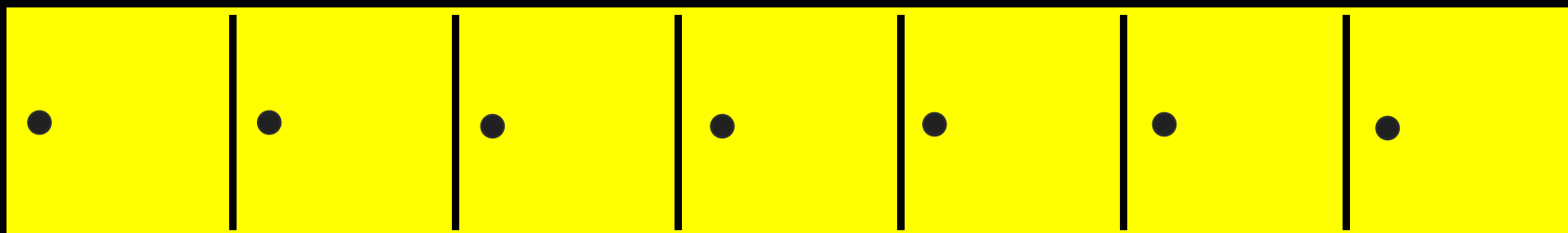
4G85

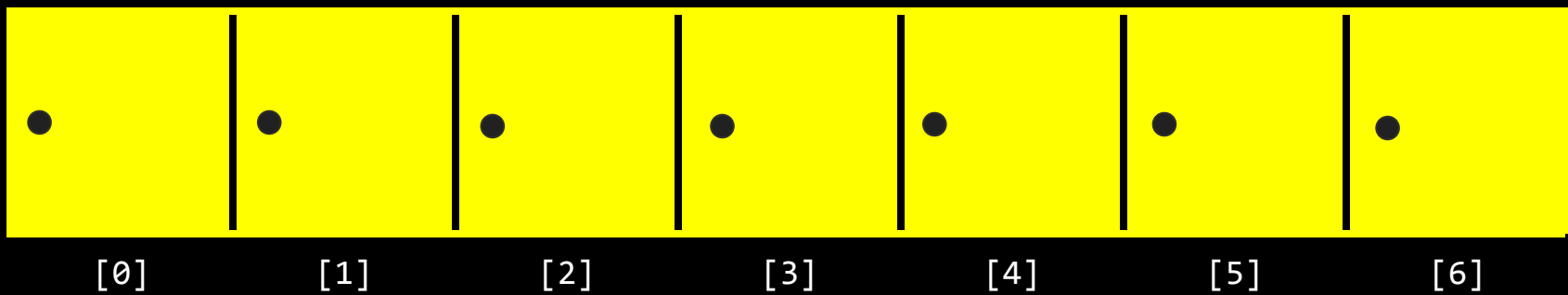




--	--	--	--	--	--	--

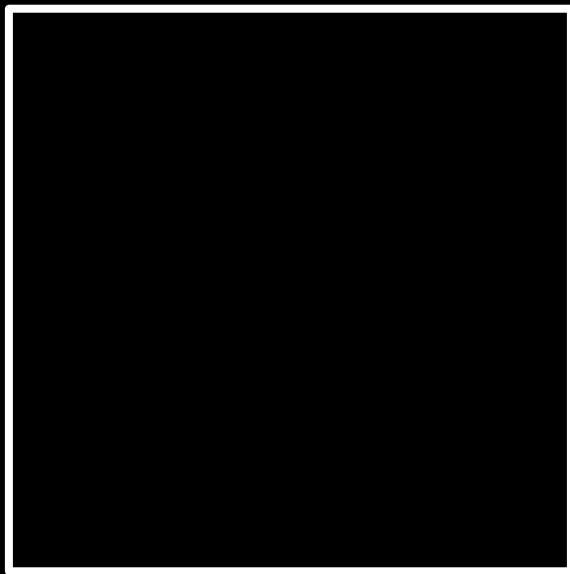
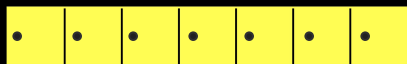
4	6	8	2	7	5	0
---	---	---	---	---	---	---



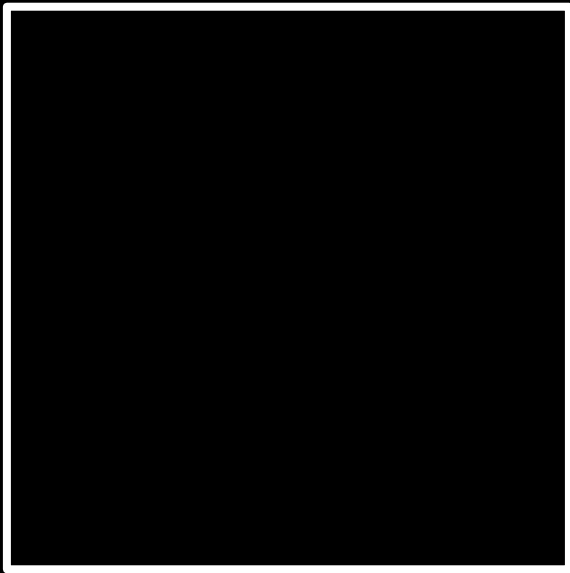
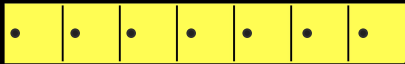


searching





→ output



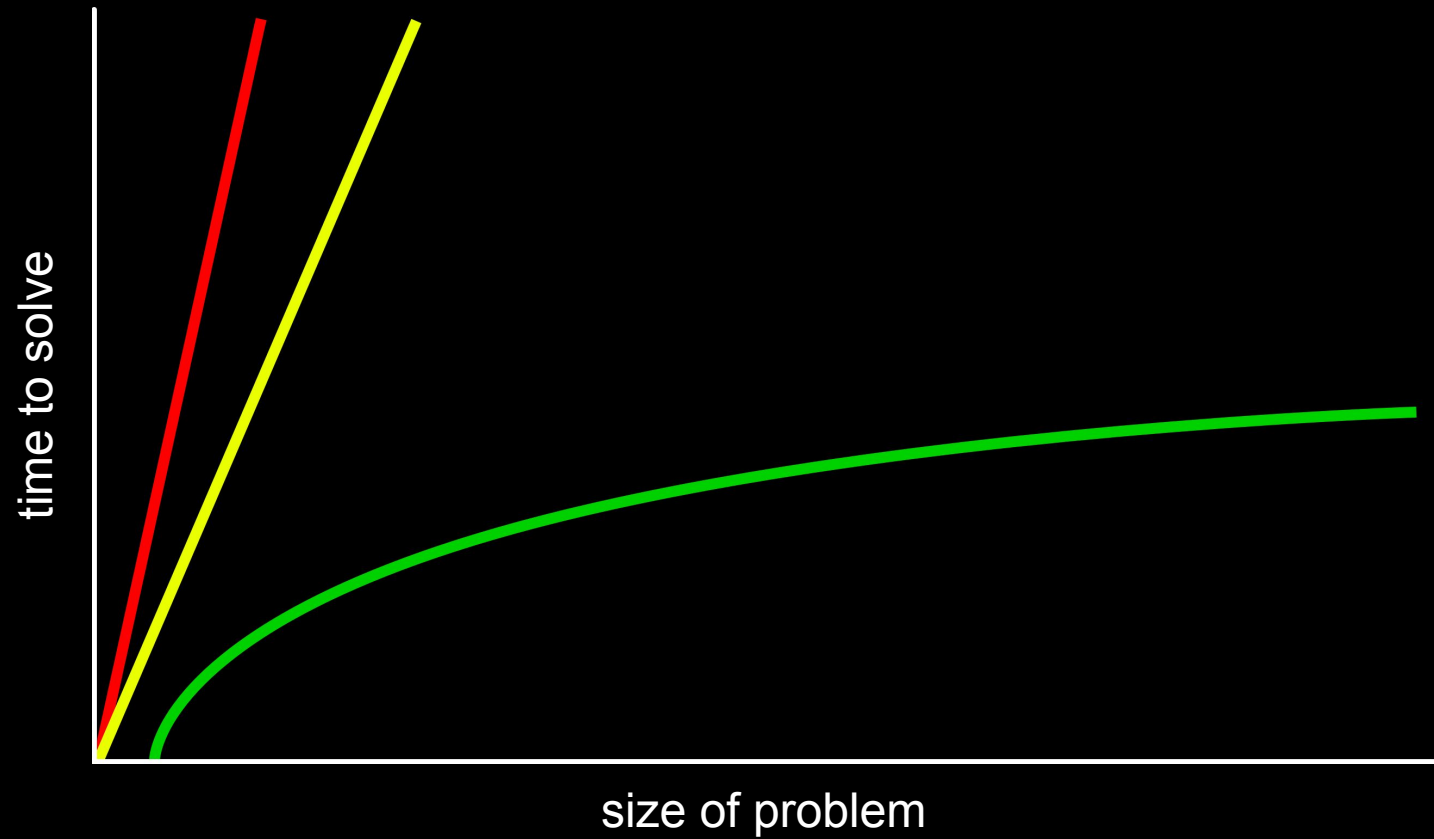
bool

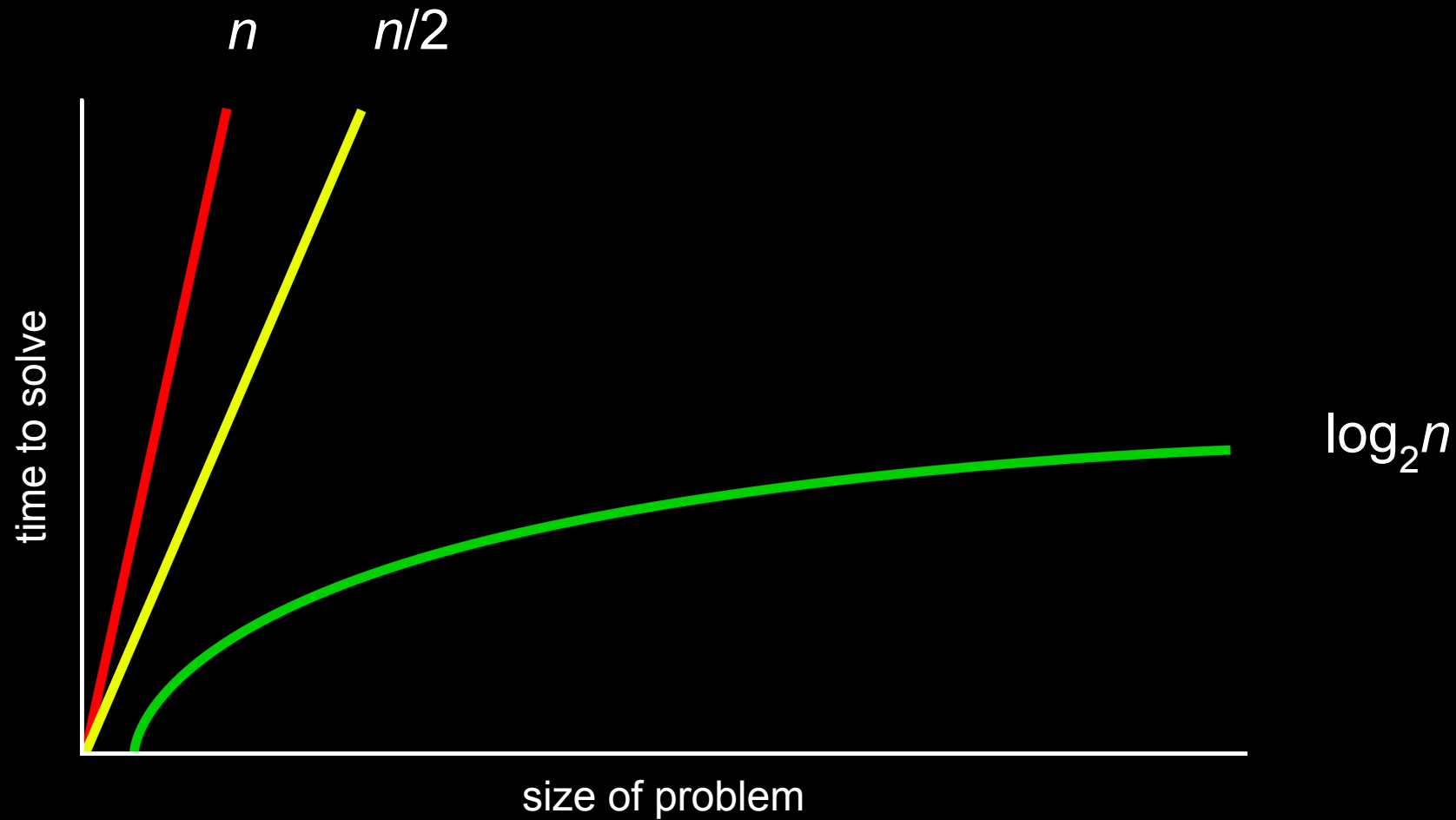


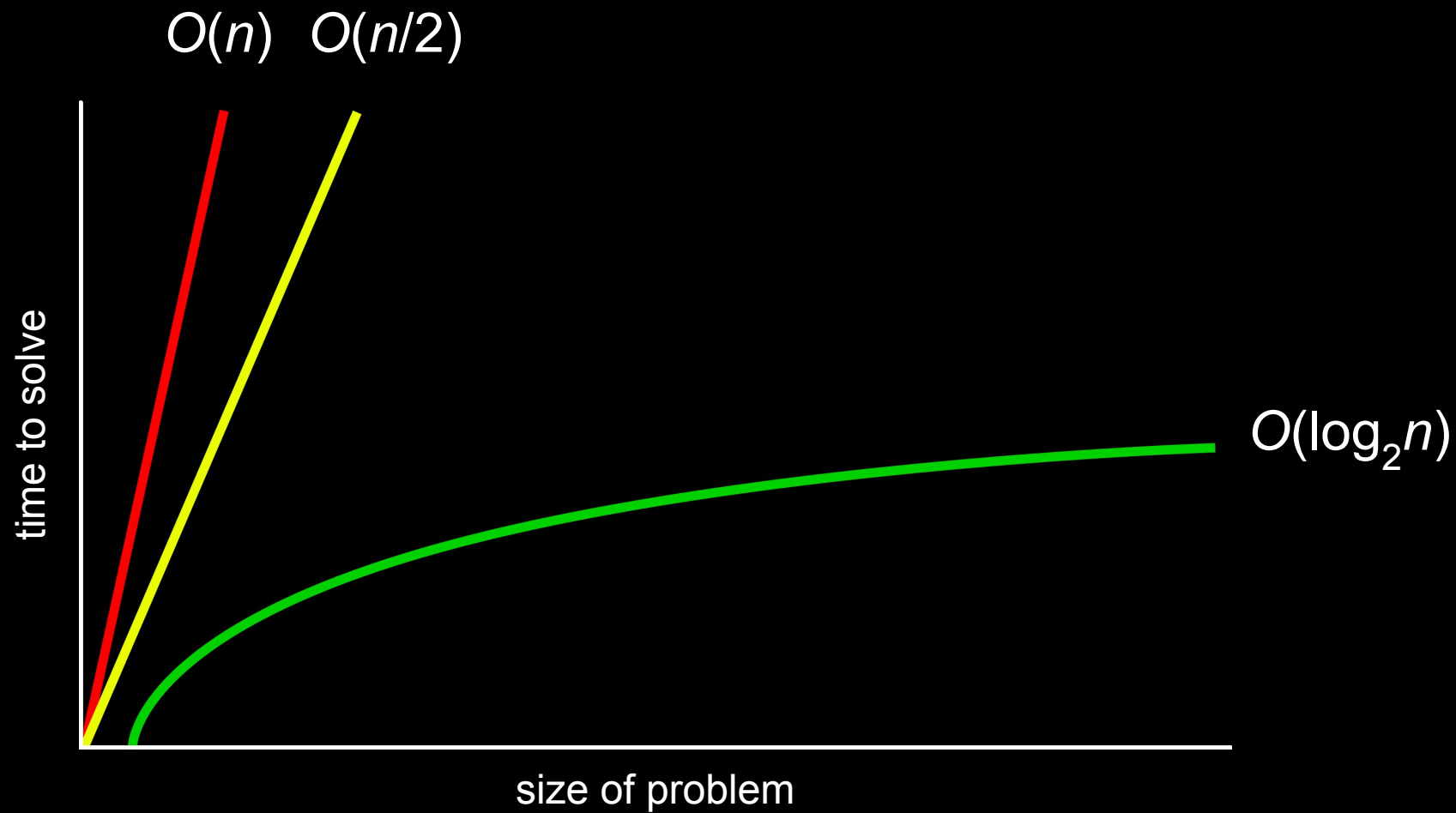
algorithms

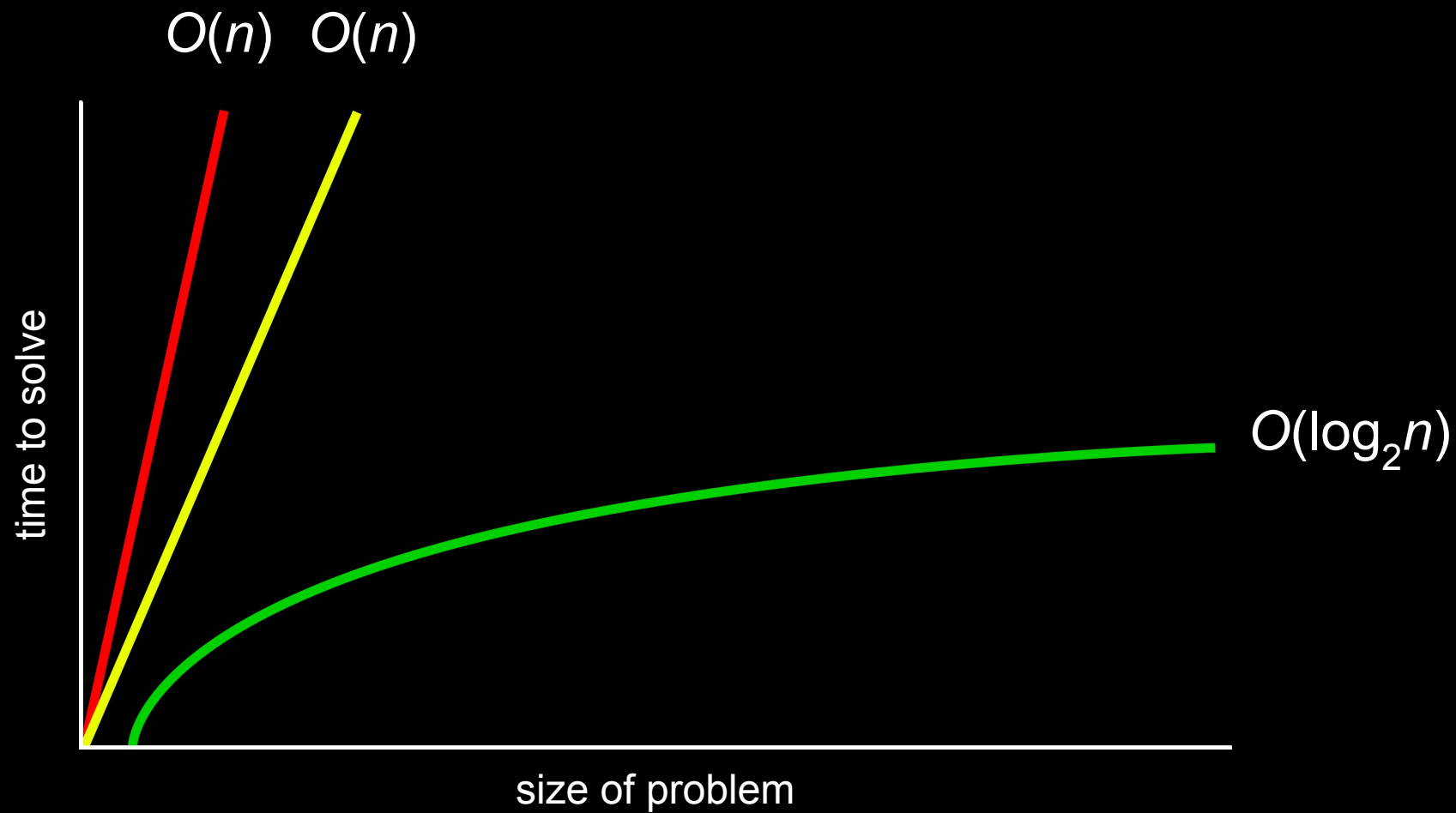
running times

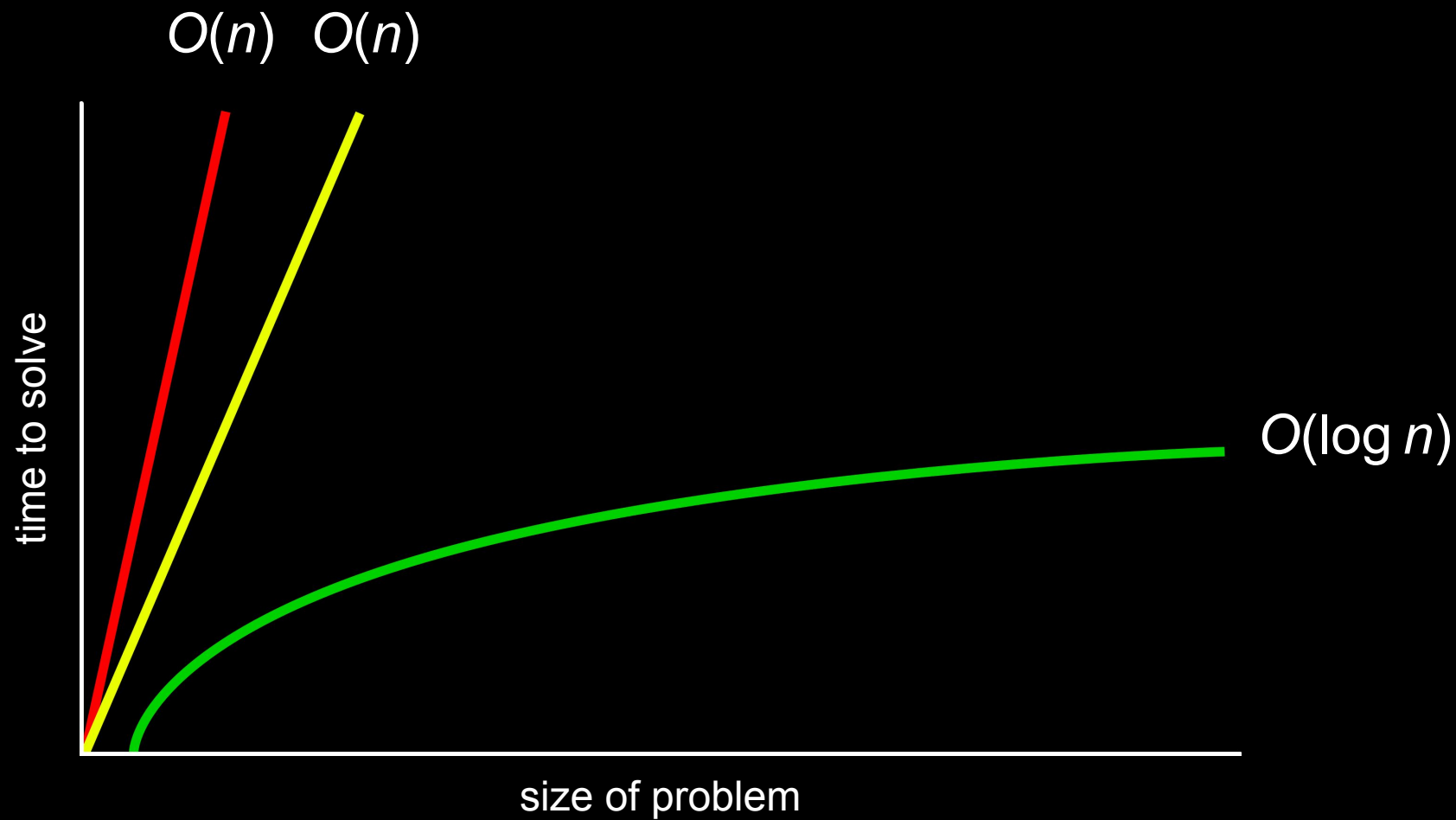
O

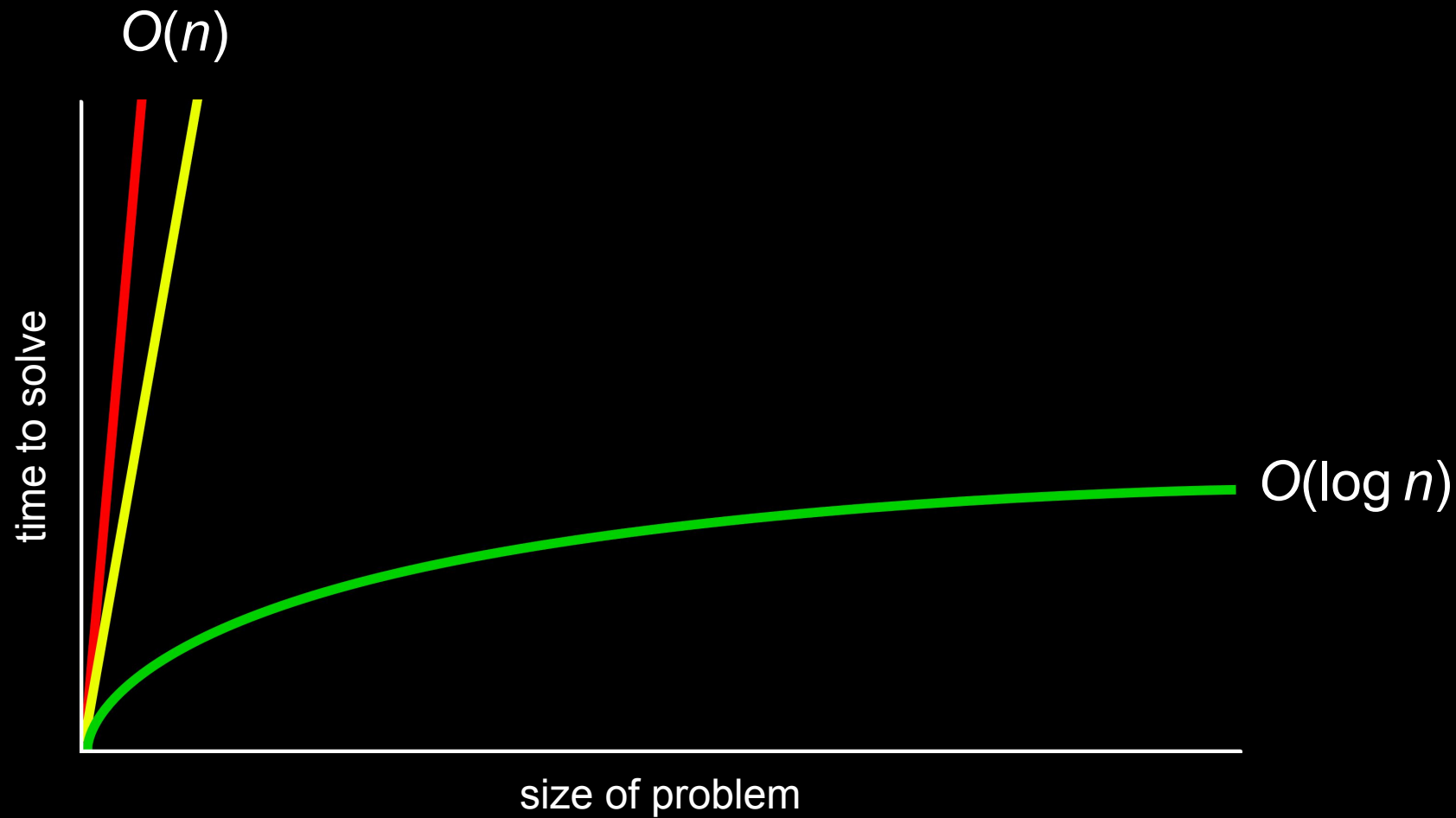












$$O(n^2)$$

$$O(n \log n)$$

$$O(n)$$

$$O(\log n)$$

$$O(1)$$

Ω

$$\Omega(n^2)$$

$$\Omega(n \log n)$$

$$\Omega(n)$$

$$\Omega(\log n)$$

$$\Omega(1)$$



$$\Theta(n^2)$$

$$\Theta(n \log n)$$

$$\Theta(n)$$

$$\Theta(\log n)$$

$$\Theta(1)$$

linear search

```
For each door from left to right
    If number is behind door
        Return true
Return false
```

```
For i from 0 to n-1
    If number behind doors[i]
        Return true
Return false
```

$$O(n^2)$$

$$O(n \log n)$$

$$O(n)$$

$$O(\log n)$$

$$O(1)$$

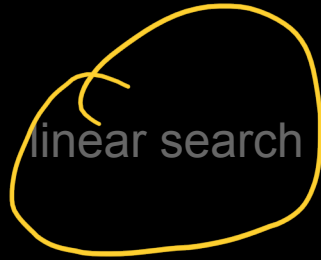
$O(n^2)$

$O(n \log n)$

$O(n)$

$O(\log n)$

$O(1)$



linear search

$$\Omega(n^2)$$

$$\Omega(n \log n)$$

$$\Omega(n)$$

$$\Omega(\log n)$$

$$\Omega(1)$$

$\Omega(n^2)$

$\Omega(n \log n)$

$\Omega(n)$

$\Omega(\log n)$

$\Omega(1)$

linear search



binary search

```
If number behind middle door
    Return true
Else if number < middle door
    Search left half
Else if number > middle door
    Search right half
```

If no doors

If number behind middle door

Return true

Else if number < middle door

Search left half

Else if number > middle door

Search right half

If no doors

Return false

If number behind middle door

Return true

Else if number < middle door

Search left half

Else if number > middle door

Search right half

If no doors

 Return false

If number behind doors[middle]

 Return true

Else if number < doors[middle]

 Search doors[0] through doors[middle - 1]

Else if number > doors[middle]

 Search doors[middle + 1] through doors[n - 1]

$$O(n^2)$$

$$O(n \log n)$$

$$O(n)$$

$$O(\log n)$$

$$O(1)$$

$O(n^2)$

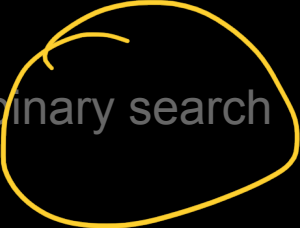
$O(n \log n)$

$O(n)$

$O(\log n)$

$O(1)$

binary search



$$\Omega(n^2)$$

$$\Omega(n \log n)$$

$$\Omega(n)$$

$$\Omega(\log n)$$

$$\Omega(1)$$

$\Omega(n^2)$

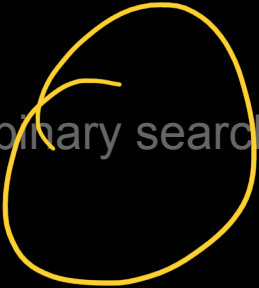
$\Omega(n \log n)$

$\Omega(n)$

$\Omega(\log n)$

$\Omega(1)$

binary search



```
int numbers[]
```

```
string names[]
```

```
string names[]  
string numbers[]
```

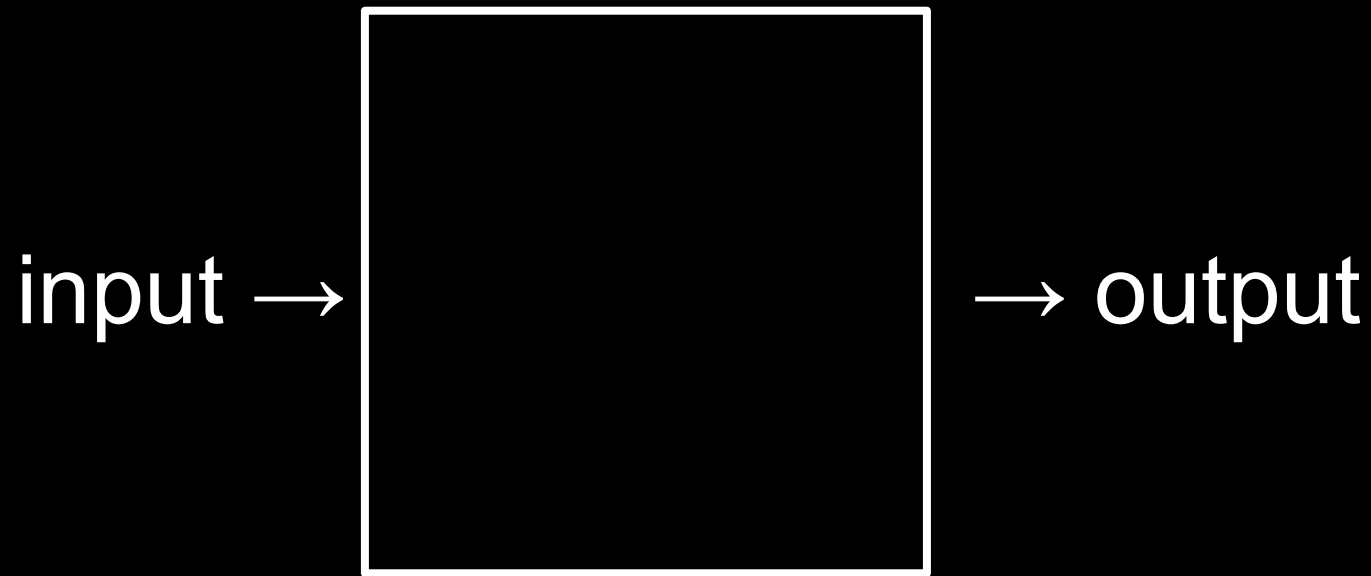
data structures

person people[]

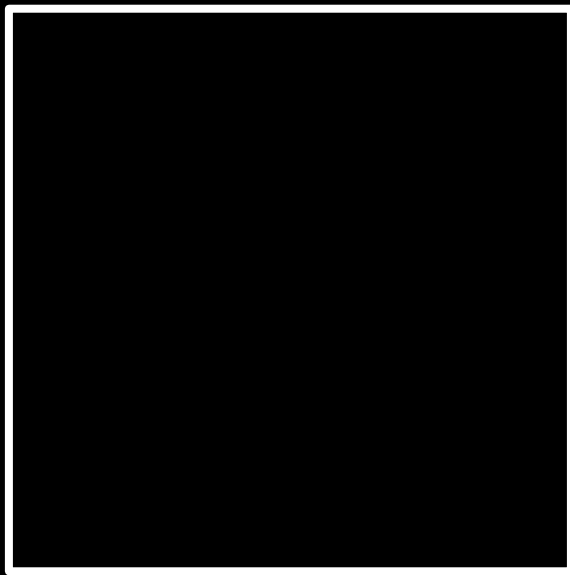

```
string name;  
string number;
```

```
typedef struct
{
    string name;
    string number;
}
person;
```

sorting

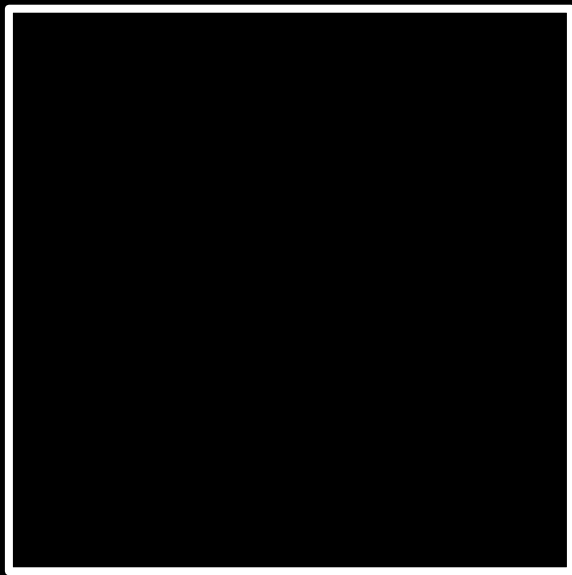


unsorted →



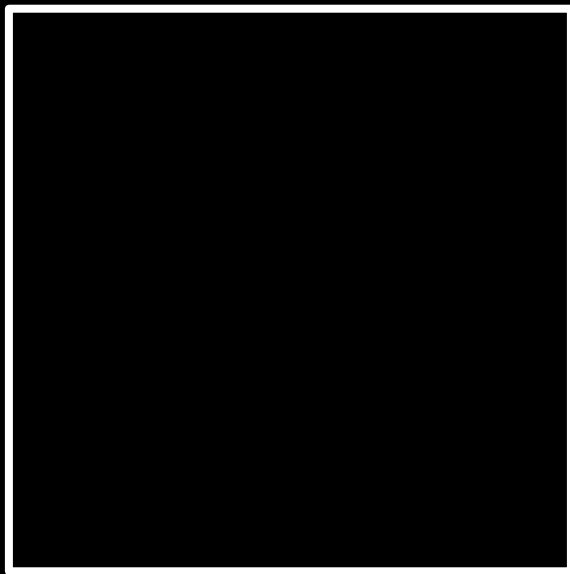
→ output

unsorted →



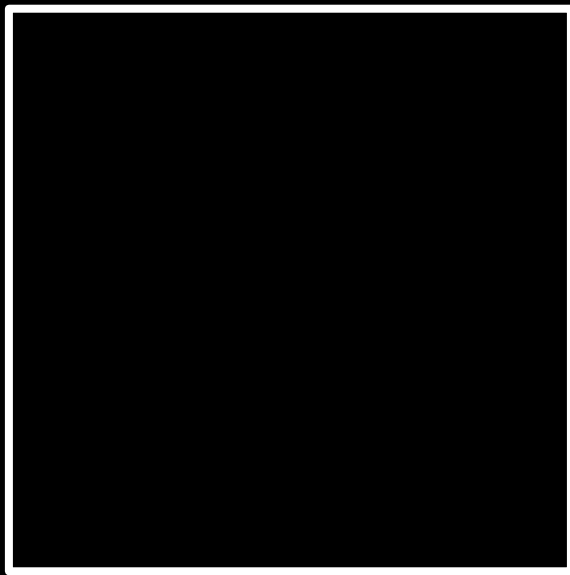
→ sorted

6 3 8 5 2 7 4 1



→ sorted

6 3 8 5 2 7 4 1



1 2 3 4 5 6 7 8

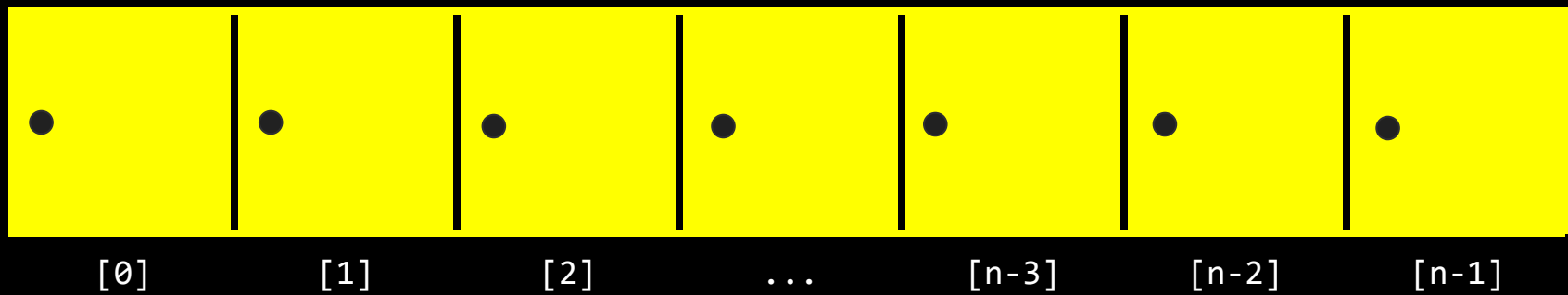
selection sort

5 2 7 4 1 6 3 0

For i from 0 to n-1

Find smallest number between numbers[i] and numbers[n-1]

Swap smallest number with numbers[i]



$$n + (n - 1)$$

$$n + (n - 1) + (n - 2)$$

$$n + (n - 1) + (n - 2) + \dots + 1$$

$$n + (n - 1) + (n - 2) + \dots + 1$$

$$n(n + 1)/2$$

$$n + (n - 1) + (n - 2) + \dots + 1$$

$$n(n + 1)/2$$

$$(n^2 + n)/2$$

$$n + (n - 1) + (n - 2) + \dots + 1$$

$$n(n + 1)/2$$

$$(n^2 + n)/2$$

$$n^2/2 + n/2$$

$$n + (n - 1) + (n - 2) + \dots + 1$$

$$n(n + 1)/2$$

$$(n^2 + n)/2$$

$$n^2/2 + n/2$$

$$O(n^2)$$

$$O(n^2)$$

$$O(n \log n)$$

$$O(n)$$

$$O(\log n)$$

$$O(1)$$

$O(n^2)$

selection sort

$O(n \log n)$

$O(n)$

$O(\log n)$

$O(1)$

For i from 0 to n-1

Find smallest number between numbers[i] and numbers[n-1]

Swap smallest number with numbers[i]

$$\Omega(n^2)$$

$$\Omega(n \log n)$$

$$\Omega(n)$$

$$\Omega(\log n)$$

$$\Omega(1)$$

$\Omega(n^2)$

selection sort

$\Omega(n \log n)$

$\Omega(n)$

$\Omega(\log n)$

$\Omega(1)$

$$\Theta(n^2)$$

$$\Theta(n \log n)$$

$$\Theta(n)$$

$$\Theta(\log n)$$

$$\Theta(1)$$

$\Theta(n^2)$

selection sort

$\Theta(n \log n)$

$\Theta(n)$

$\Theta(\log n)$

$\Theta(1)$

bubble sort

5 2 7 4 1 6 3 0

Repeat $n-1$ times

For i from 0 to $n-2$

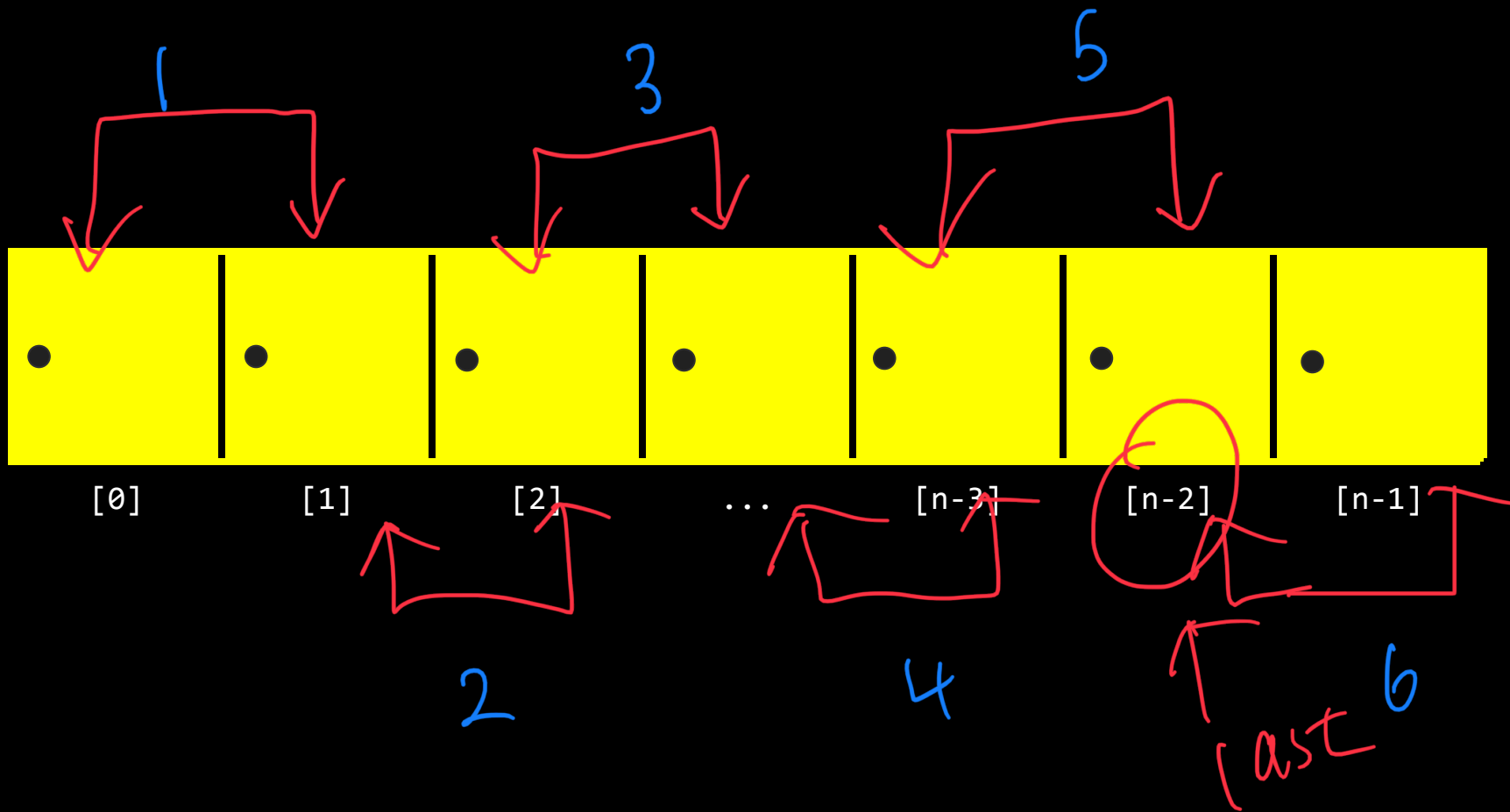
If $\text{numbers}[i]$ and $\text{numbers}[i+1]$ out of order

Swap them

← only move

← 比較 2 數

↑ 2 不超過 Border



$$(n - 1) \times (n - 1)$$

$$(n - 1) \times (n - 1)$$

$$n^2 - 1n - 1n + 1$$

$$(n - 1) \times (n - 1)$$

$$n^2 - 1n - 1n + 1$$

$$n^2 - 2n + 1$$

$$(n - 1) \times (n - 1)$$

$$n^2 - 1n - 1n + 1$$

$$n^2 - 2n + 1$$

$$O(n^2)$$

$$O(n^2)$$

$$O(n \log n)$$

$$O(n)$$

$$O(\log n)$$

$$O(1)$$

$O(n^2)$ bubble sort

$O(n \log n)$

$O(n)$

$O(\log n)$

$O(1)$

Repeat $n-1$ times

For i from 0 to $n-2$

If `numbers[i]` and `numbers[i+1]` out of order

Swap them

If no swaps

Quit

$$\Omega(n^2)$$

$$\Omega(n \log n)$$

$$\Omega(n)$$

$$\Omega(\log n)$$

$$\Omega(1)$$

$\Omega(n^2)$

$\Omega(n \log n)$

$\Omega(n)$

bubble sort

$\Omega(\log n)$

$\Omega(1)$

recursion

If no doors

Return false

If number behind middle door

Return true

Else if number < middle door

Search left half

Else if number > middle door

Search right half

If no doors

Return false

If number behind middle door

Return true

Else if number < middle door

Search left half

Else if number > middle door

Search right half

```
1  Pick up phone book
2  Open to middle of phone book
3  Look at page
4  If person is on page
5      Call person
6  Else if person is earlier in book
7      Open to middle of left half of book
8      Go back to line 3
9  Else if person is later in book
10     Open to middle of right half of book
11     Go back to line 3
12 Else
13     Quit
```

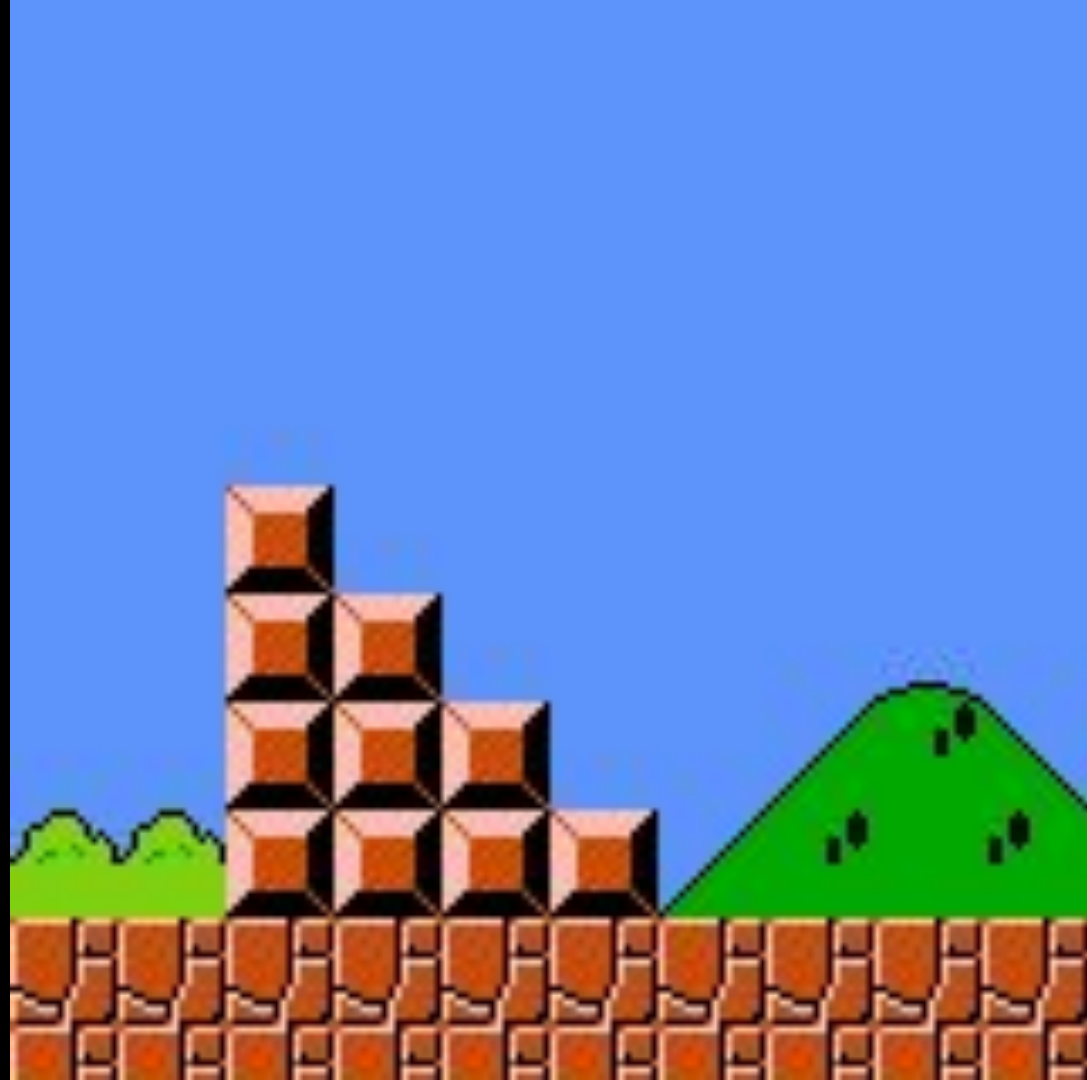
```
1  Pick up phone book
2  Open to middle of phone book
3  Look at page
4  If person is on page
5      Call person
6  Else if person is earlier in book
7      Open to middle of left half of book
8      Go back to line 3
9  Else if person is later in book
10     Open to middle of right half of book
11     Go back to line 3
12 Else
13     Quit
```

```
1  Pick up phone book
2  Open to middle of phone book
3  Look at page
4  If person is on page
5      Call person
6  Else if person is earlier in book
7      Open to middle of left half of book
8      Go back to line 3
9  Else if person is later in book
10     Open to middle of right half of book
11     Go back to line 3
12 Else
13     Quit
```

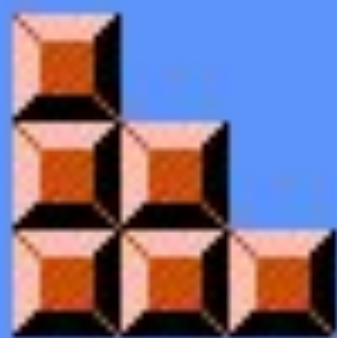


```
1  Pick up phone book
2  Open to middle of phone book
3  Look at page
4  If person is on page
5      Call person
6  Else if person is earlier in book
7      Search left half of book
8
9  Else if person is later in book
10     Search right half of book
11
12 Else
13     Quit
```

```
1  Pick up phone book
2  Open to middle of phone book
3  Look at page
4  If person is on page
5      Call person
6  Else if person is earlier in book
7      Search left half of book
8  Else if person is later in book
9      Search right half of book
10 Else
11     Quit
```

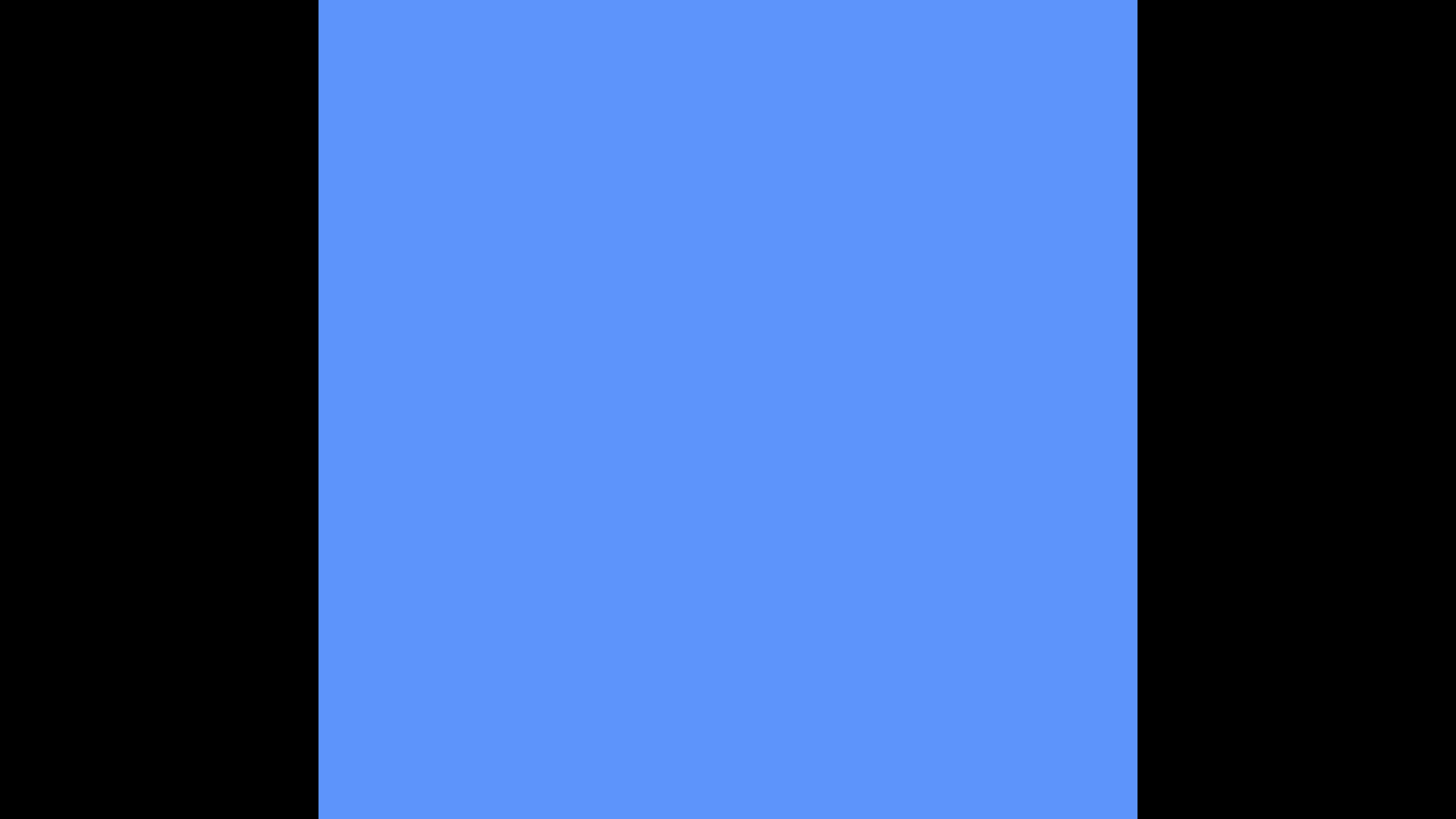












merge sort

Sort left half of numbers
Sort right half of numbers
Merge sorted halves

If only one number

Quit

Else

Sort left half of numbers

Sort right half of numbers

Merge sorted halves

If only one number

Quit

Else

Sort left half of numbers

Sort right half of numbers

Merge sorted halves

2

4

5

7

0

1

3

6

If only one number

Quit

Else

Sort left half of numbers

Sort right half of numbers

Merge sorted halves

5 2 7 4 1 6 3 0

$$O(n^2)$$

$$O(n \log n)$$

$$O(n)$$

$$O(\log n)$$

$$O(1)$$

$O(n^2)$

$O(n \log n)$ merge sort

$O(n)$

$O(\log n)$

$O(1)$

$$\Omega(n^2)$$

$$\Omega(n \log n)$$

$$\Omega(n)$$

$$\Omega(\log n)$$

$$\Omega(1)$$

$$\Omega(n^2)$$

$$\Omega(n \log n) \quad \text{merge sort}$$

$$\Omega(n)$$

$$\Omega(\log n)$$

$$\Omega(1)$$

$$\Theta(n^2)$$

$$\Theta(n \log n)$$

$$\Theta(n)$$

$$\Theta(\log n)$$

$$\Theta(1)$$

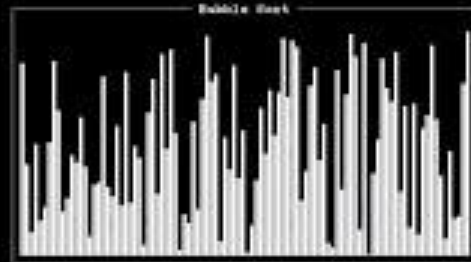
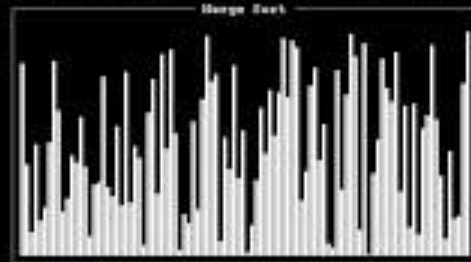
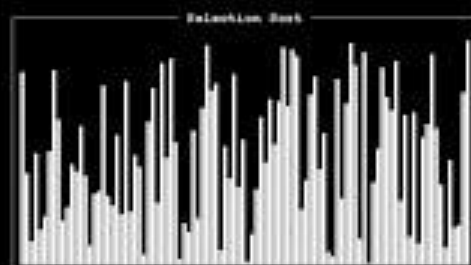
$\Theta(n^2)$

$\Theta(n \log n)$ merge sort

$\Theta(n)$

$\Theta(\log n)$

$\Theta(1)$



This is CS50