This is CS50

help50

style50

check50

printf

debug50

ddb

help50

style50

check50

printf

debug50

ddb

Share

you

I'm hoping you can help me solve a problem

ddb

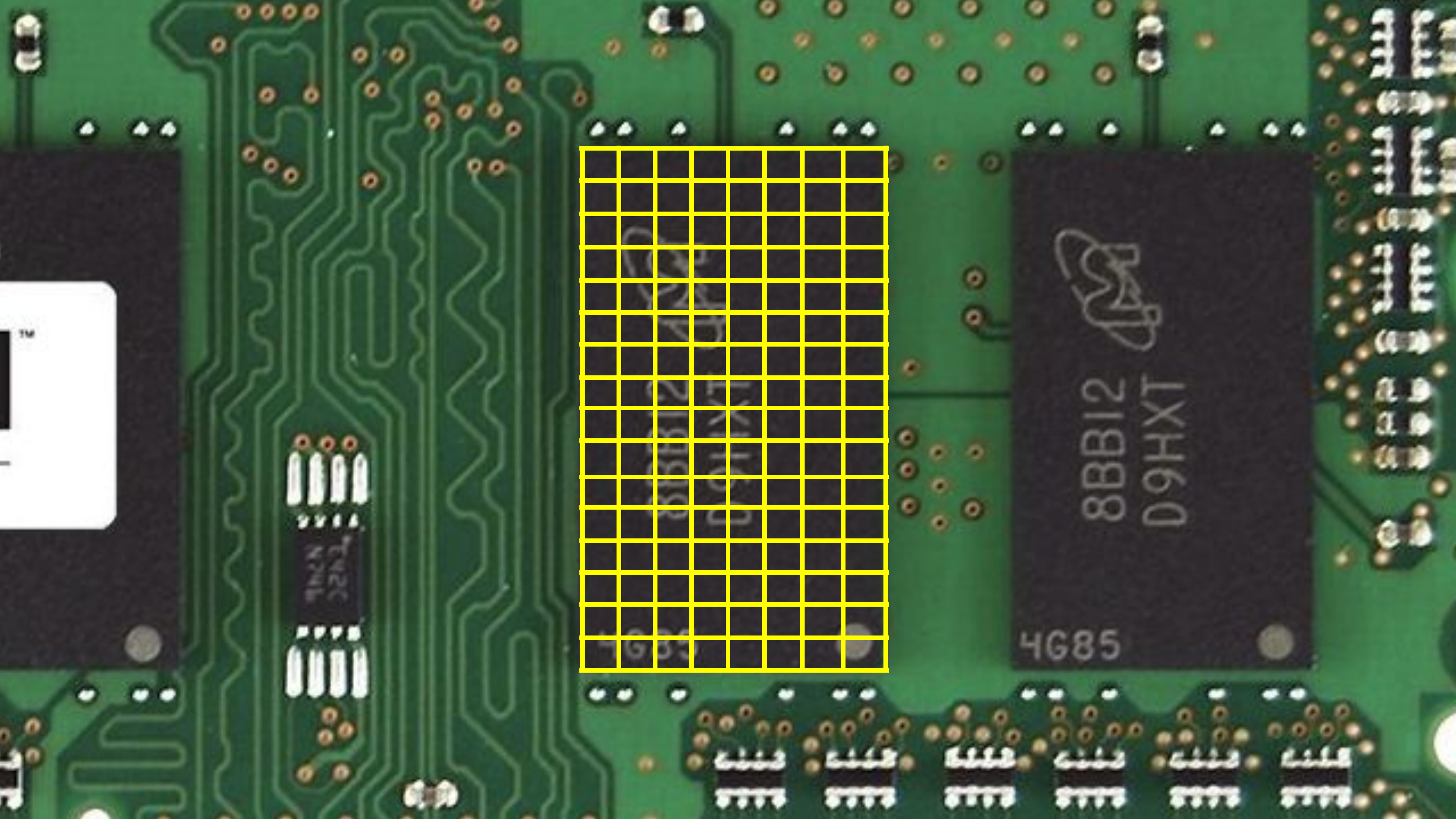
quack

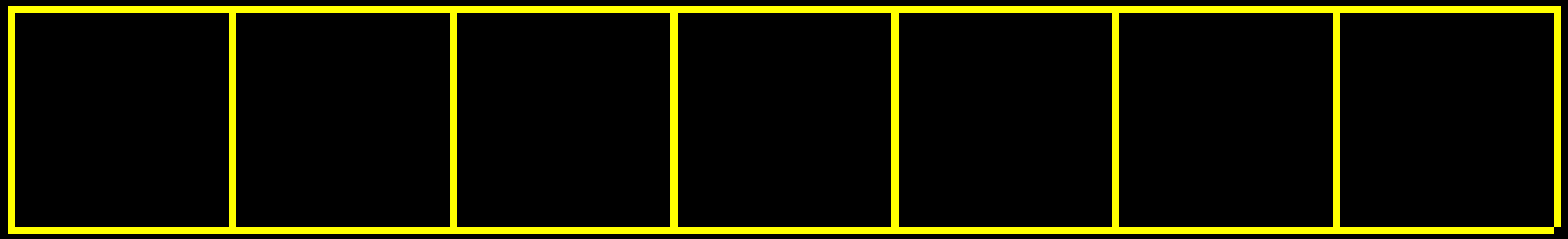
Collaborate

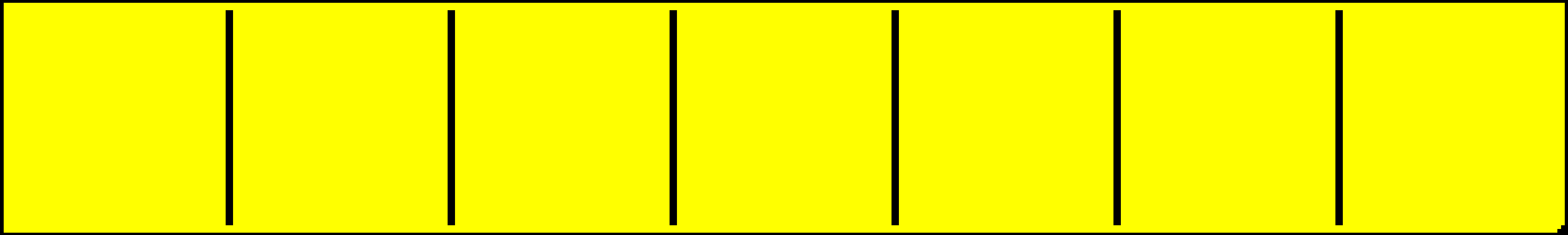
Outline

Debugger

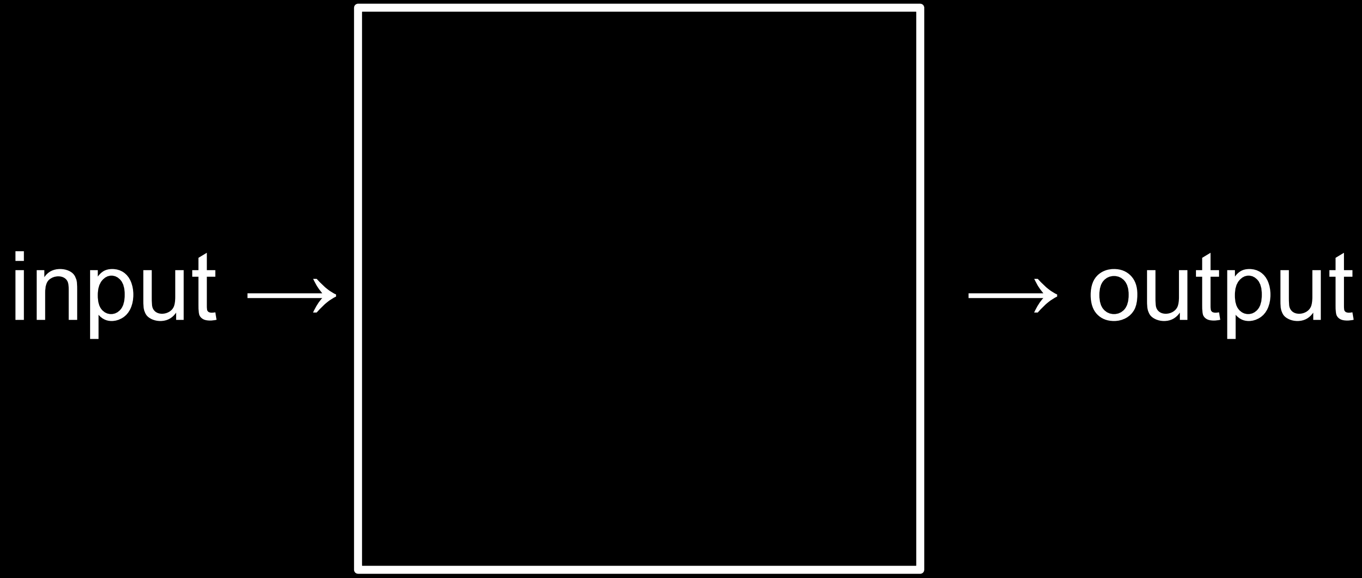

8BB12
D9HXT
4G85
8BB12
D9HXT
4G85

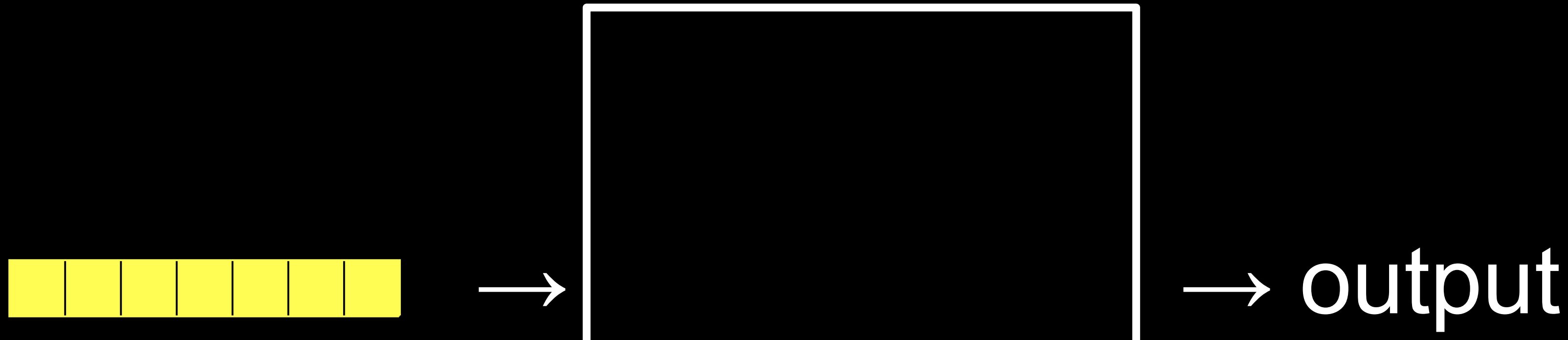

8BB12
D9HXT
4G85

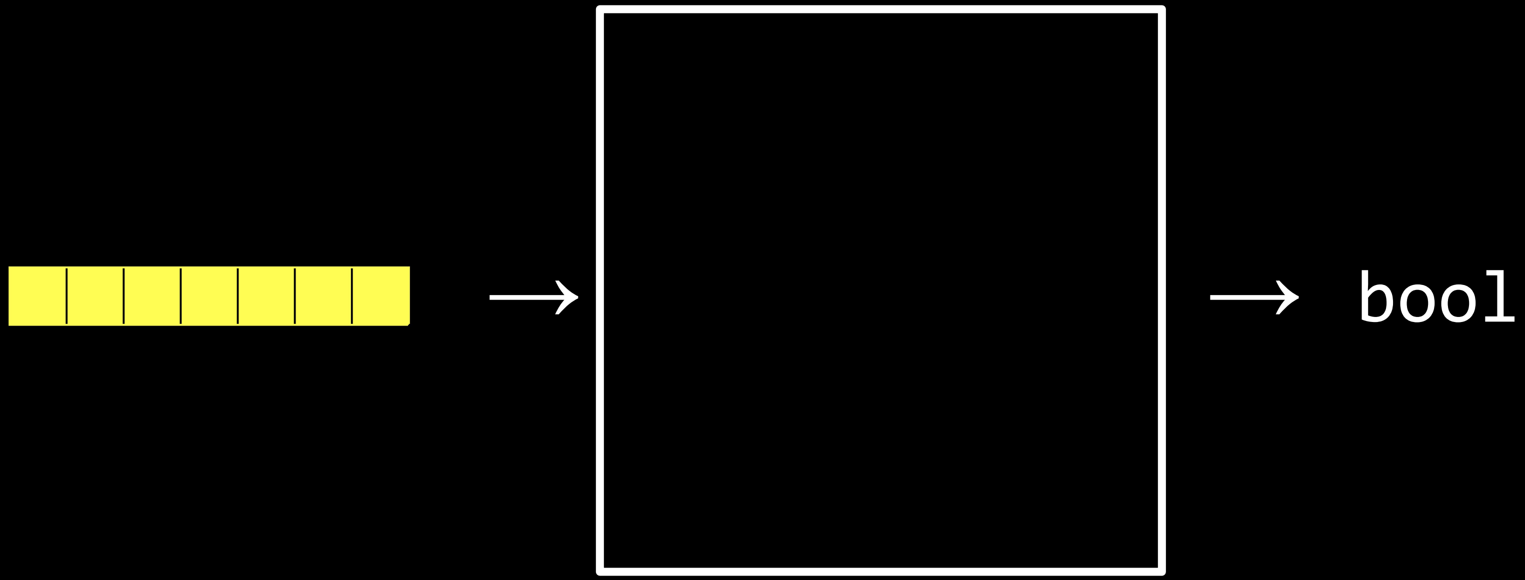




searching


input →
→ output

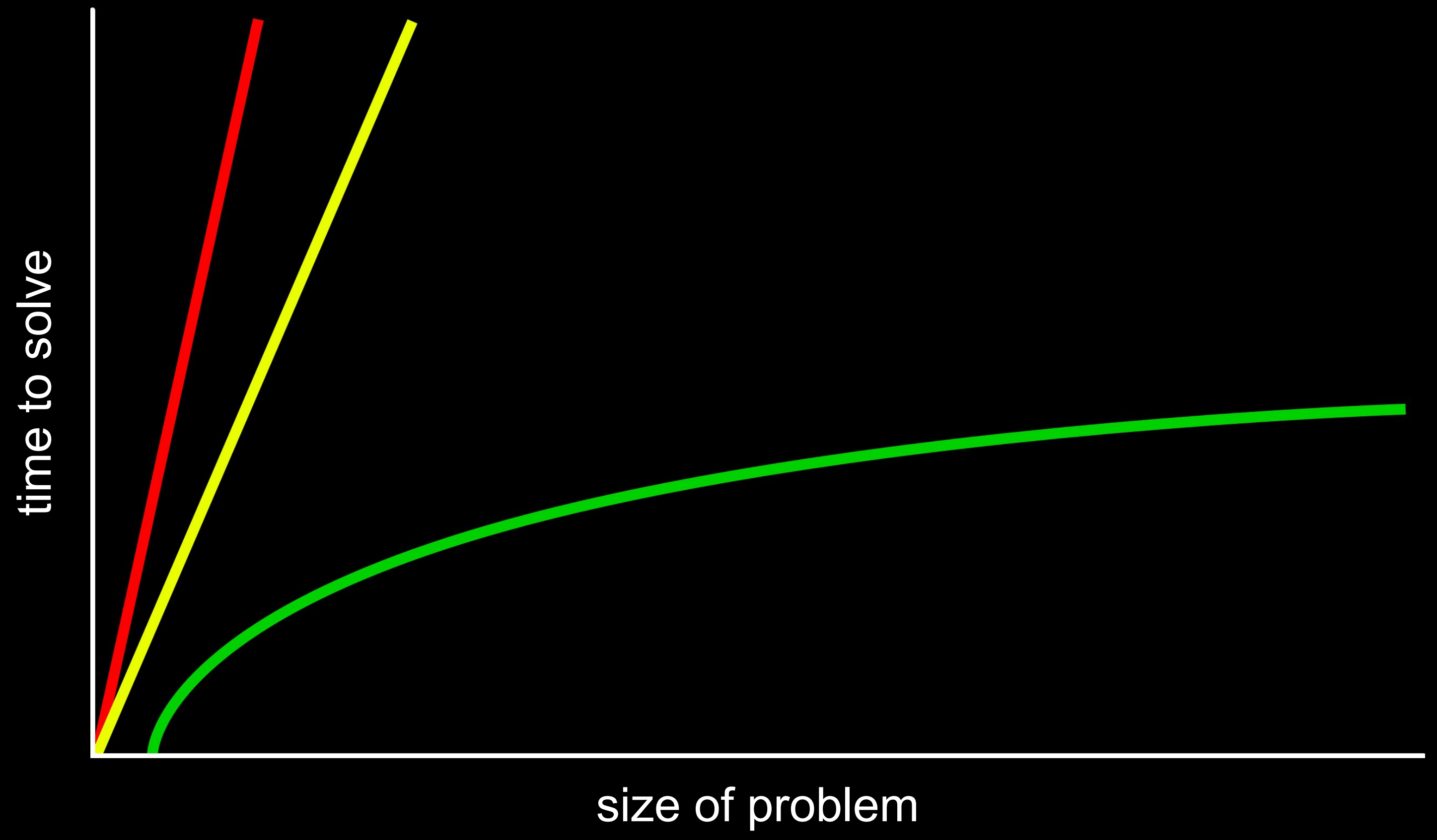

output

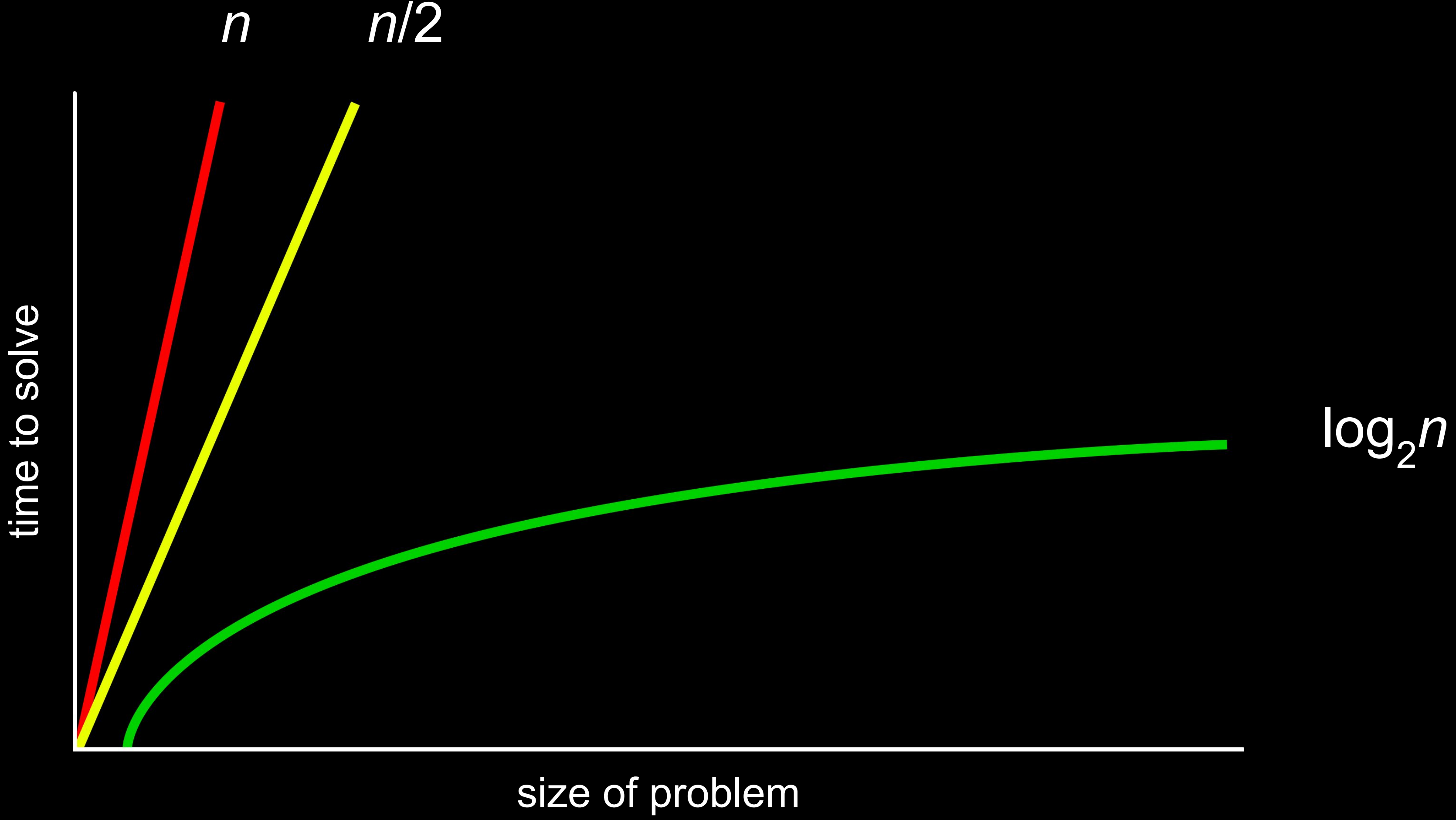

bool

algorithms

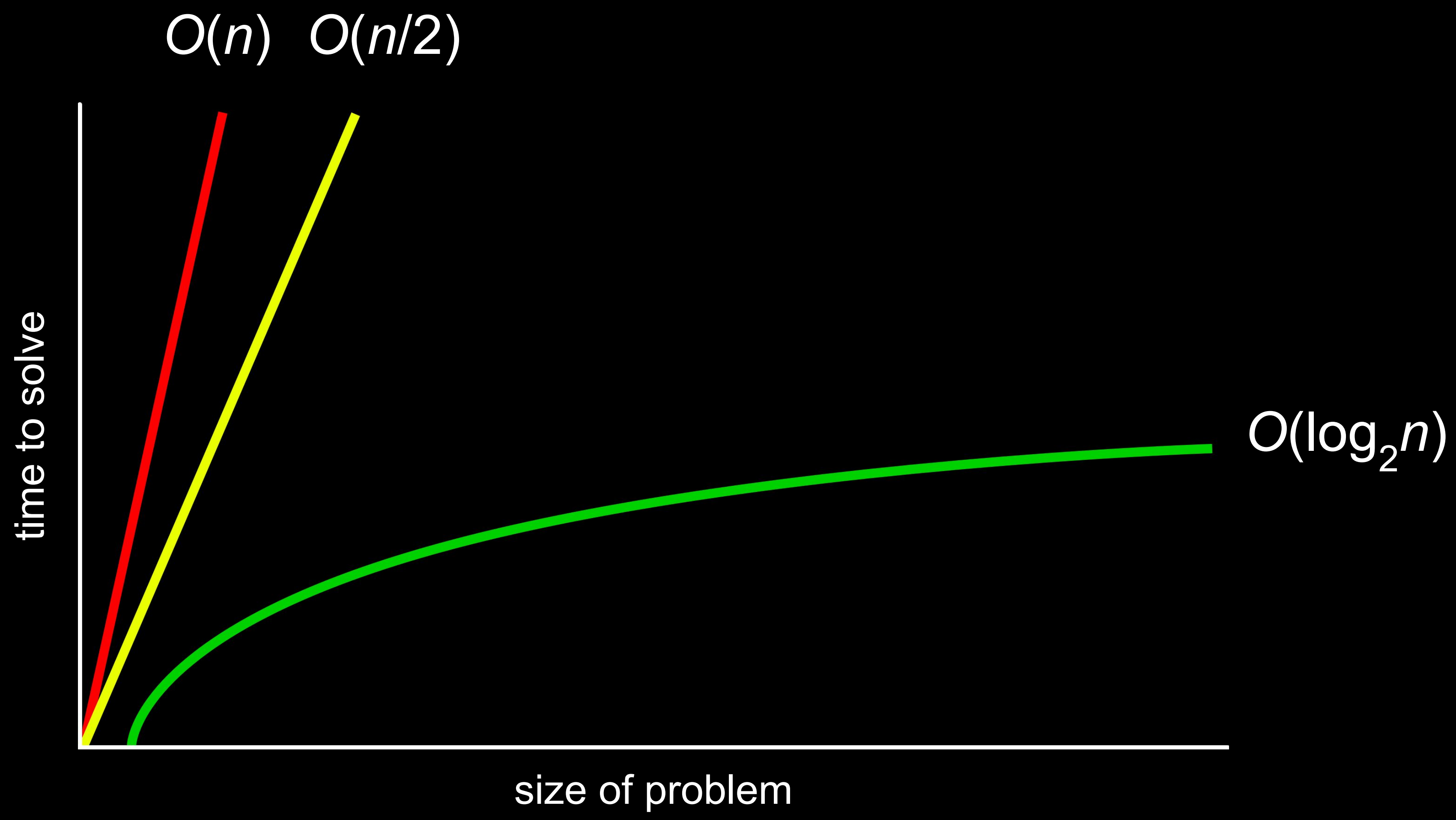
running times

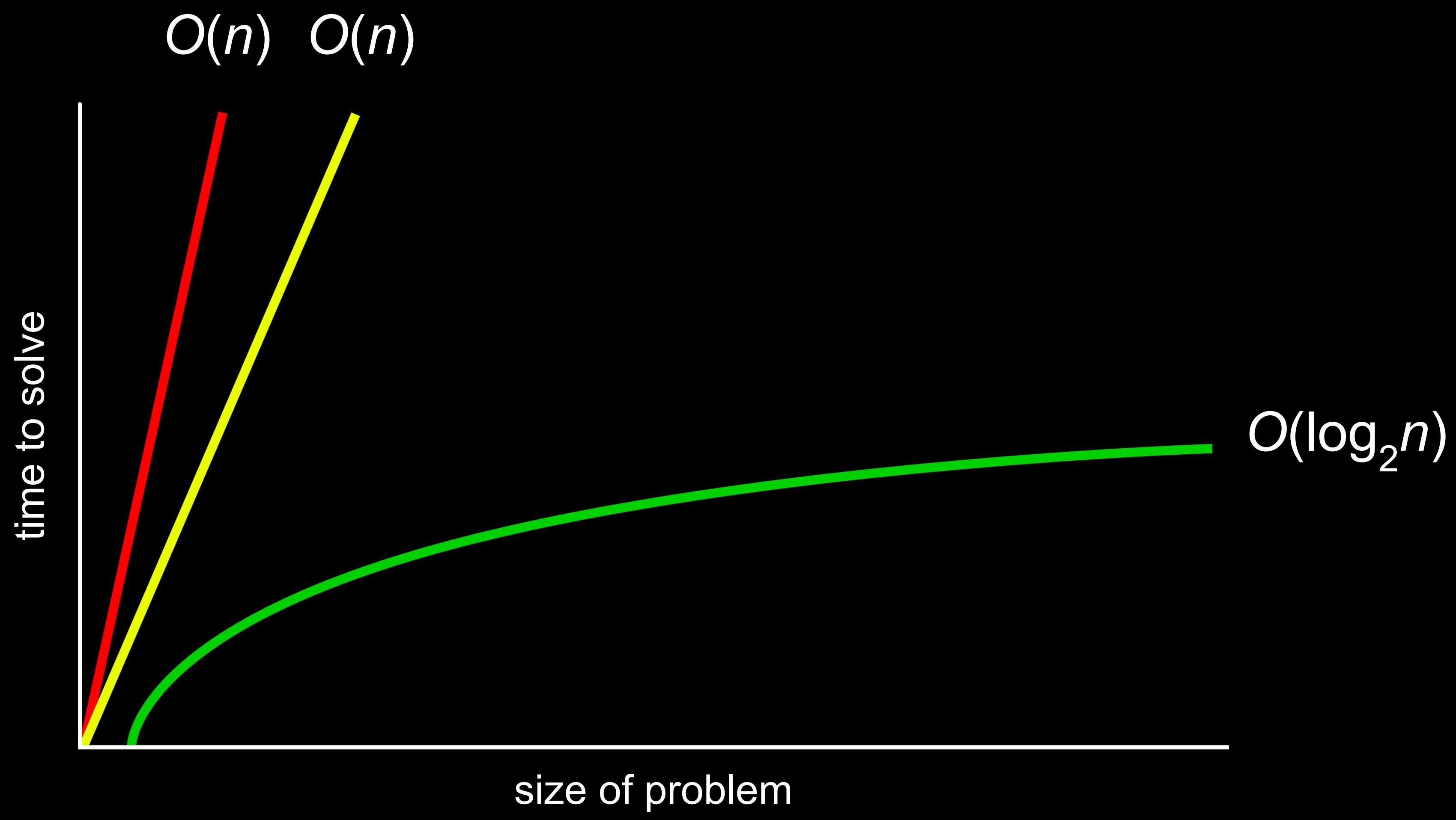
o


time to solve
size of problem

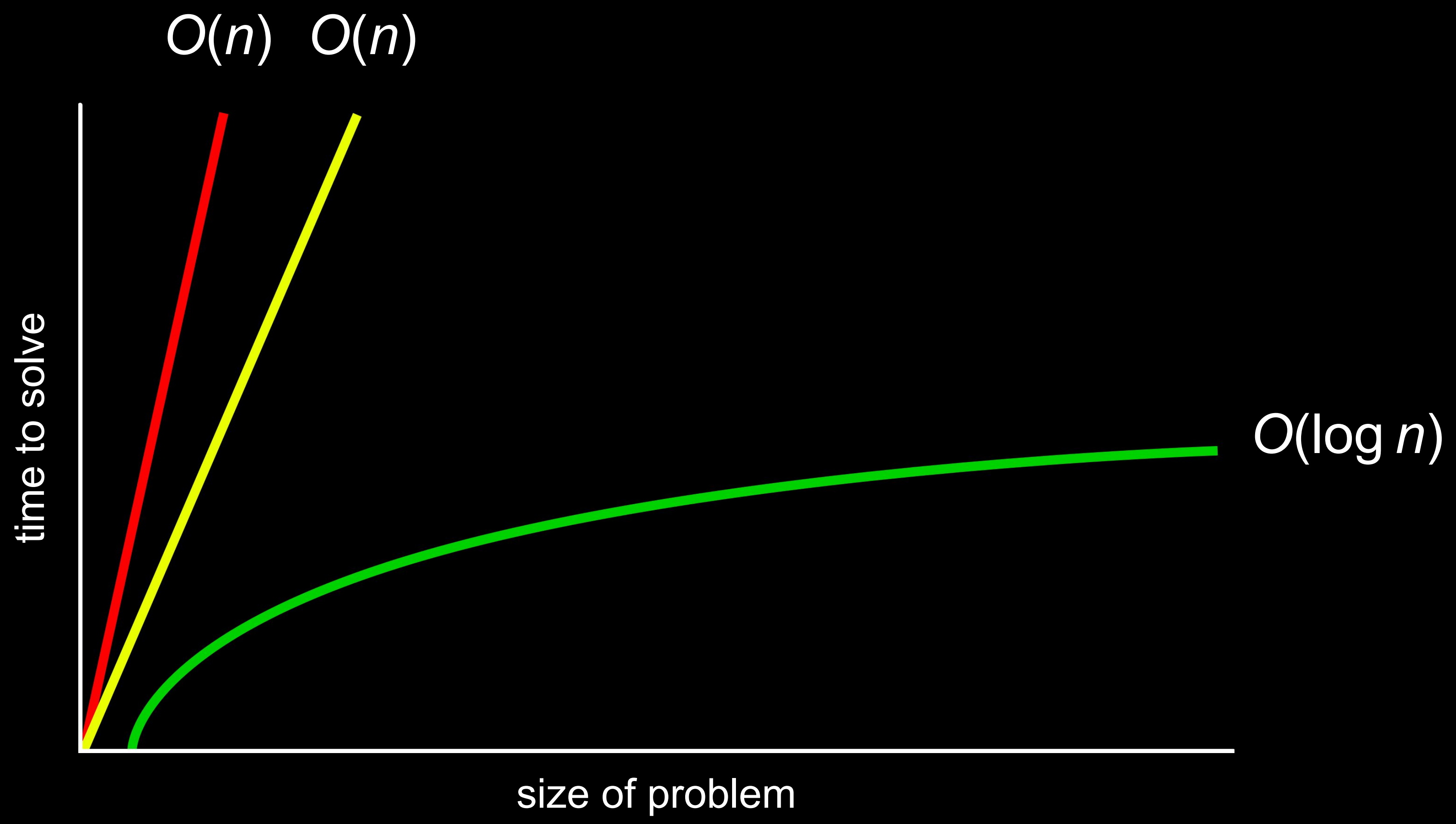


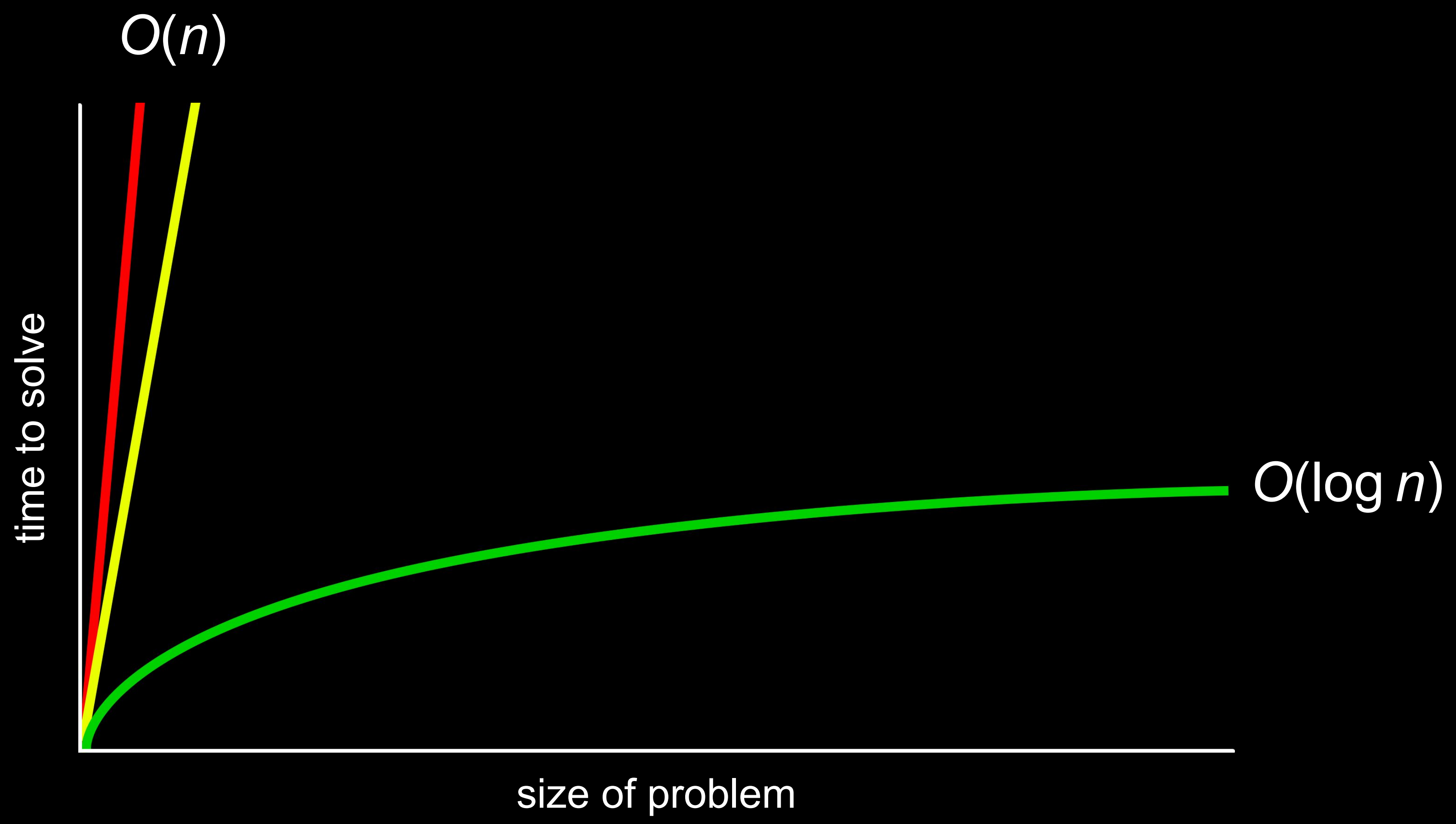
n
n/2
log2n
time to solve
size of problem


$O(n)$
$O(n/2)$
$O(\log_2 n)$
time to solve
size of problem



$O(n)$
$O(n)$
$O(\log_2 n)$
time to solve
size of problem


$O(n)$
$O(n)$
$O(\log n)$
time to solve
size of problem



O(n)
O(log n)
time to solve
size of problem

$O(n^2)$

$O(n \log n)$

$O(n)$

$O(\log n)$

$O(1)$

Ω

$\Omega(n^2)$

$\Omega(n \log n)$

$\Omega(n)$

$\Omega(\log n)$

$\Omega(1)$

linear search

```
For i from 0 to n-1
    If number behind i'th door
        Return true
Return false
```

$O(n^2)$

$O(n \log n)$

$O(n)$

$O(\log n)$

$O(1)$

$O(n^2)$

$O(n \log n)$

$O(n)$ linear search

$O(\log n)$

$O(1)$

$\Omega(n^2)$

$\Omega(n \log n)$

$\Omega(n)$

$\Omega(\log n)$

$\Omega(1)$

$\Omega(n^2)$

$\Omega(n \log n)$

$\Omega(n)$

$\Omega(\log n)$

$\Omega(1)$ 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
```

$O(n^2)$

$O(n \log n)$

$O(n)$ linear search

$O(\log n)$

$O(1)$

$O(n^2)$	
$O(n \log n)$	
$O(n)$	linear search
$O(\log n)$	binary search
$O(1)$	

$\Omega(n^2)$

$\Omega(n \log n)$

$\Omega(n)$

$\Omega(\log n)$

$\Omega(1)$ linear search

$\Omega(n^2)$

$\Omega(n \log n)$

$\Omega(n)$

$\Omega(\log n)$

$\Omega(1)$ linear search, binary search

```
int numbers[]
```

```
string names[]
```

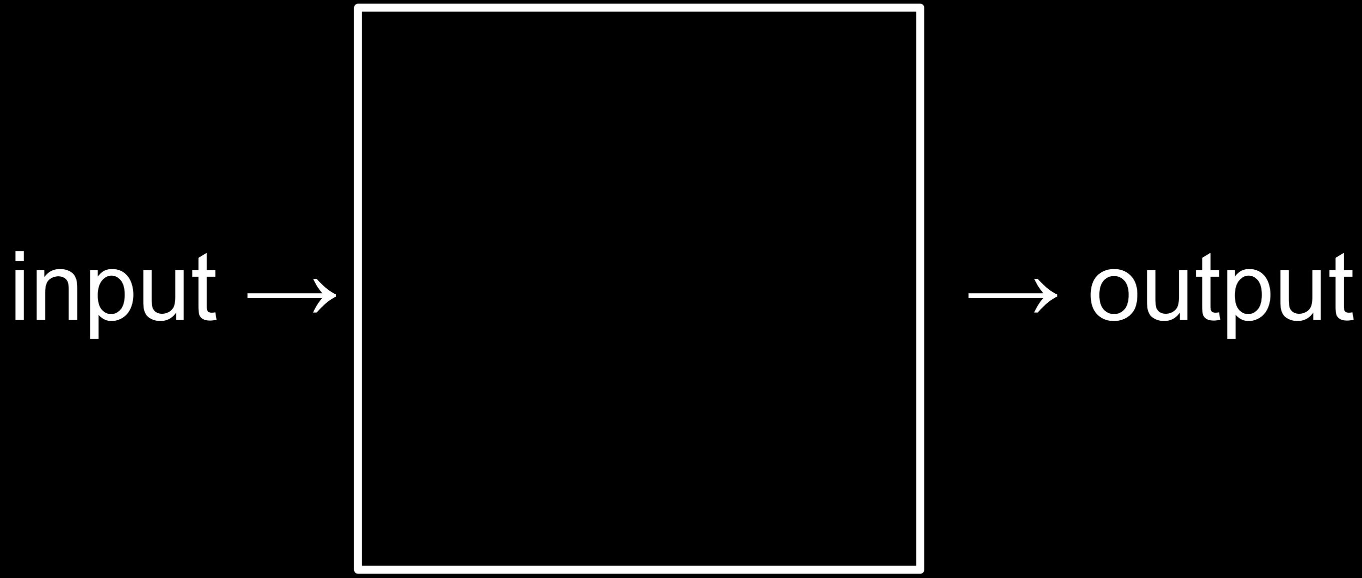
data structures

```
person people[]
```

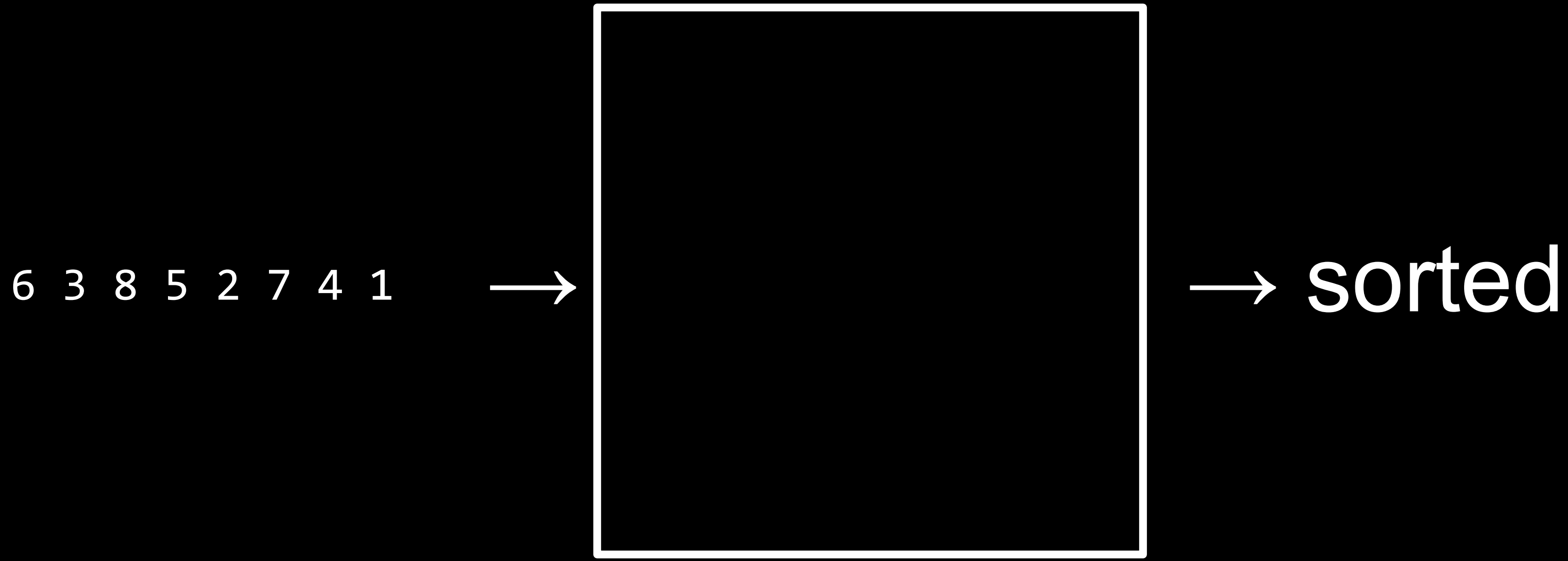
```
string name;
string number;
```

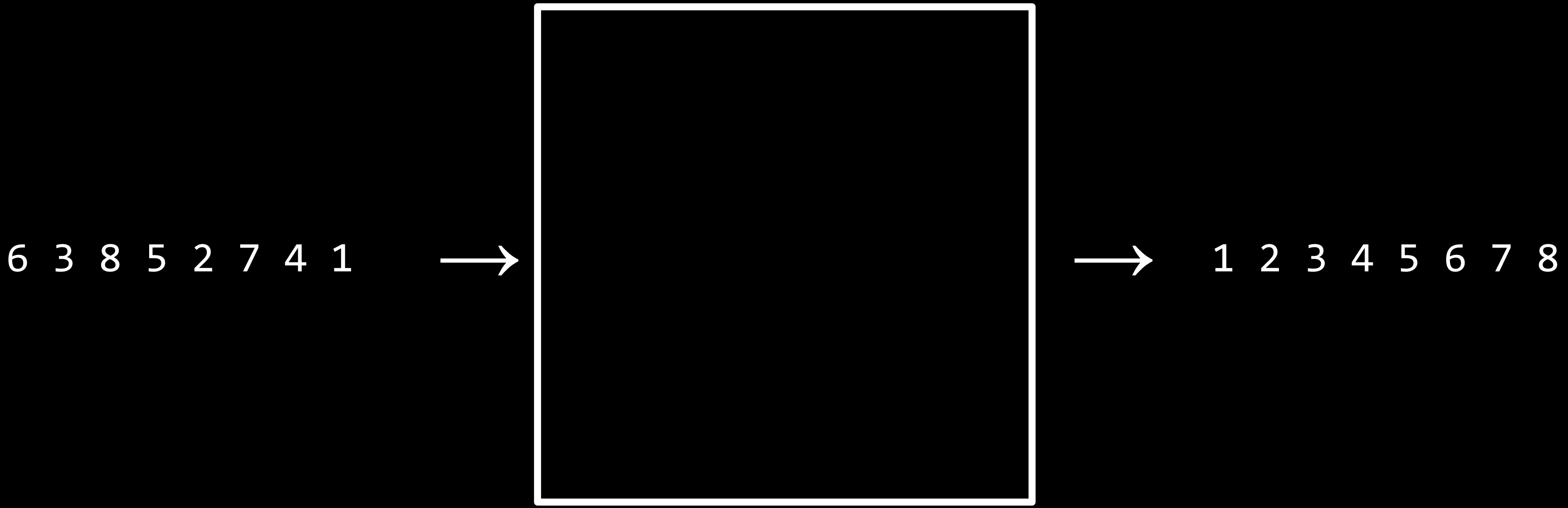
```
typedef struct
{
    string name;
    string number;
}
person;
```

sorting


input →
→ output

unsorted → $\boxed{}$ → output unsorted → $\boxed{}$ → 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

6 3 8 5 2 7 4 1

```
For i from 0 to n-1
    Find smallest item between i'th item and last item
    Swap smallest item with i'th item
```

n

$n + (n - 1)$

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

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

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

$n(n + 1)/2$

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

$n(n + 1)/2$

$(n^2 + \mathrm{n})/2$

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

$n(n + 1)/2$

$(n^2 + \mathrm{n})/2$

$n^2/2 + n/2$

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

$n(n + 1)/2$

$(n^2 + \mathrm{n})/2$

$n^2/2 + n/2$

$O(n^2)$

$O(n^2)$

$O(n \log n)$

$O(n)$ linear search

$O(\log n)$ binary search

$O(1)$

$O(n^2)$	selection sort
$O(n \log n)$	
$O(n)$	linear search
$O(\log n)$	binary search
$O(1)$	

```
For i from 0 to n-1
    Find smallest item between i'th item and last item
    Swap smallest item with i'th item
```

$\Omega(n^2)$

$\Omega(n \log n)$

$\Omega(n)$

$\Omega(\log n)$

$\Omega(1)$ linear search, binary search

$\Omega(n^2)$	selection sort
$\Omega(n \log n)$	
$\Omega(n)$	
$\Omega(\log n)$	
$\Omega(1)$	linear search, binary search

bubble sort

6 3 8 5 2 7 4 1

```
Repeat until sorted
    For i from 0 to n-2
        If i'th and i+1'th elements out of order
            Swap them
```

```
Repeat n-1 times
    For i from 0 to n-2
        If i'th and i+1'th elements out of order
            Swap them
```

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

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

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

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

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

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

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

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

$n^2 - 2n + 1$

$O(n^2)$

$O(n^2)$	selection sort
$O(n \log n)$	
$O(n)$	linear search
$O(\log n)$	binary search
$O(1)$	

$O(n^2)$	selection sort, bubble sort
$O(n \log n)$	
$O(n)$	linear search
$O(\log n)$	binary search
$O(1)$	

```
Repeat n-1 times
    For i from 0 to n-2
        If i'th and i+1'th elements out of order
            Swap them
    If no swaps
        Quit
```

$\Omega(n^2)$ selection sort

$\Omega(n \log n)$

$\Omega(n)$

$\Omega(\log n)$

$\Omega(1)$ linear search, binary search

$\Omega(n^2)$	selection sort
$\Omega(n \log n)$	
$\Omega(n)$	bubble sort
$\Omega(\log n)$	
$\Omega(1)$	linear search, binary search

recursion

```
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
```

```
Pick up phone book
Open to middle of phone book
Look at page
If person is on page
    Call person
Else if person is earlier in book
    Search left half of book

Else if person is later in book
    Search right half of book

Else
    Quit
```

```
Pick up phone book
Open to middle of phone book
Look at page
If person is on page
    Call person
Else if person is earlier in book
    Search left half of book
Else if person is later in book
    Search right half of book
Else
    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
```

3 5 6 8 1 2 4 7

```
If only one number
    Quit
Else
    Sort left half of numbers
    Sort right half of numbers
    Merge sorted halves
```

$O(n^2)$	selection sort, bubble sort
$O(n \log n)$	
$O(n)$	linear search
$O(\log n)$	binary search
$O(1)$	

$O(n^2)$	selection sort, bubble sort
$O(n \log n)$	merge sort
$O(n)$	linear search
$O(\log n)$	binary search
$O(1)$	

$\Omega(n^2)$	selection sort
$\Omega(n \log n)$	
$\Omega(n)$	bubble sort
$\Omega(\log n)$	
$\Omega(1)$	linear search, binary search

$\Omega(n^2)$	selection sort
$\Omega(n \log n)$	merge sort
$\Omega(n)$	bubble sort
$\Omega(\log n)$	
$\Omega(1)$	linear search, binary search

θ

$\Theta(n^2)$

$\Theta(n \log n)$

$\Theta(n)$

$\Theta(\log n)$

$\Theta(1)$

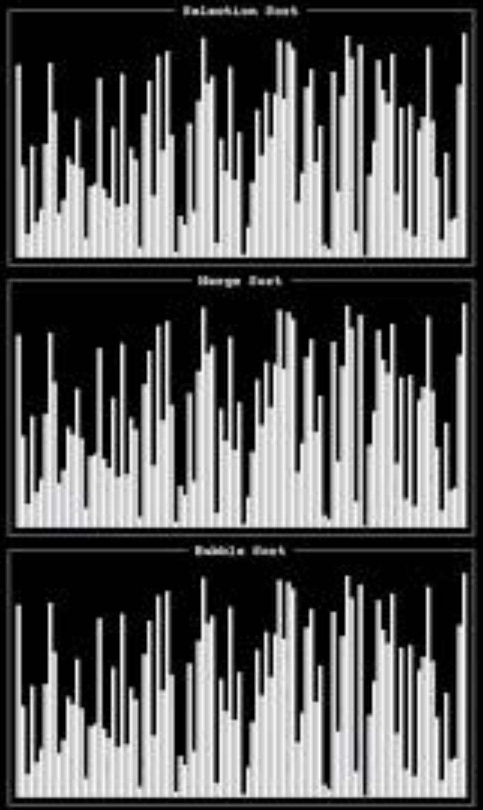
$\Theta(n^2)$ selection sort

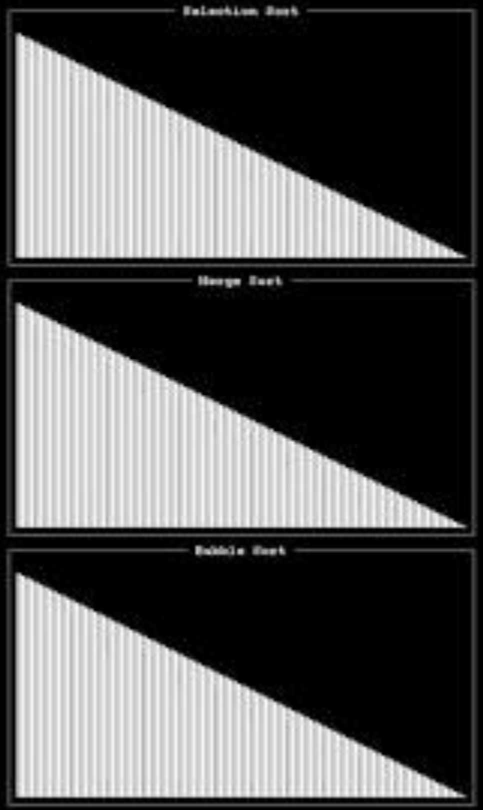
$\Theta(n \log n)$ merge sort

$\Theta(n)$

$\Theta(\log n)$

$\Theta(1)$


Selection Sort
Merge Sort
Bubble Sort


Selection Sort
Merge Sort
Bubble Sort

This is CS50