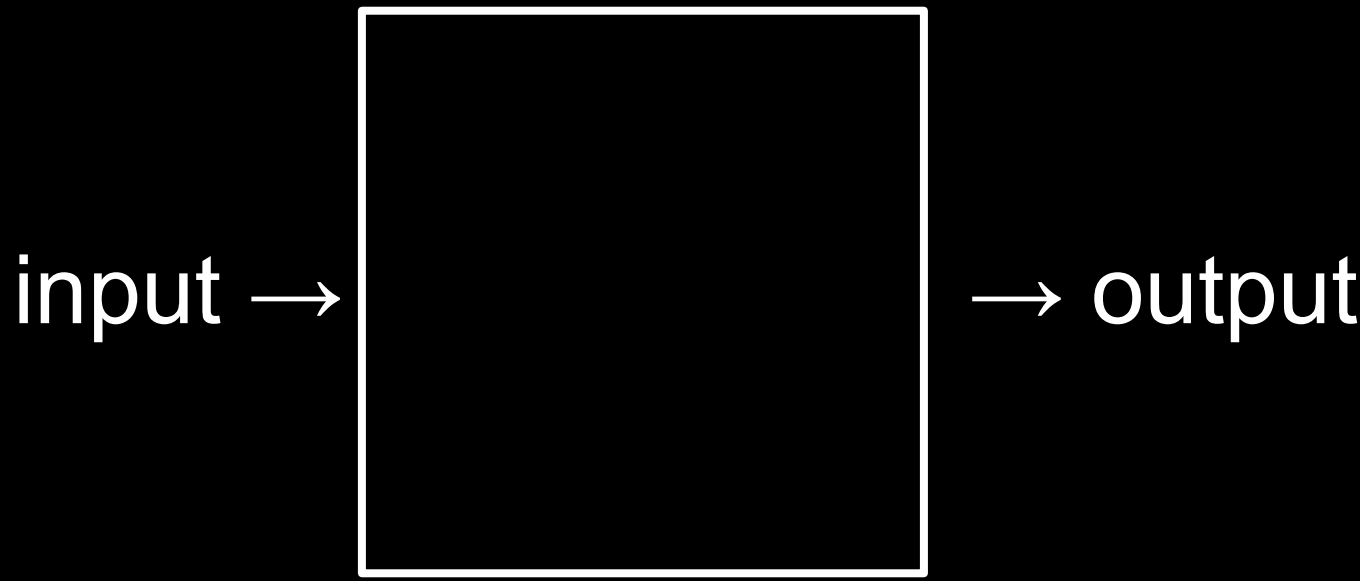
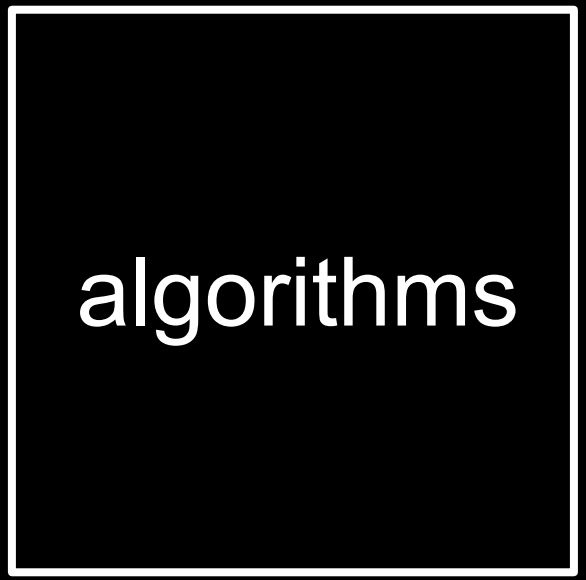
CS50 for MBAs

Algorithms

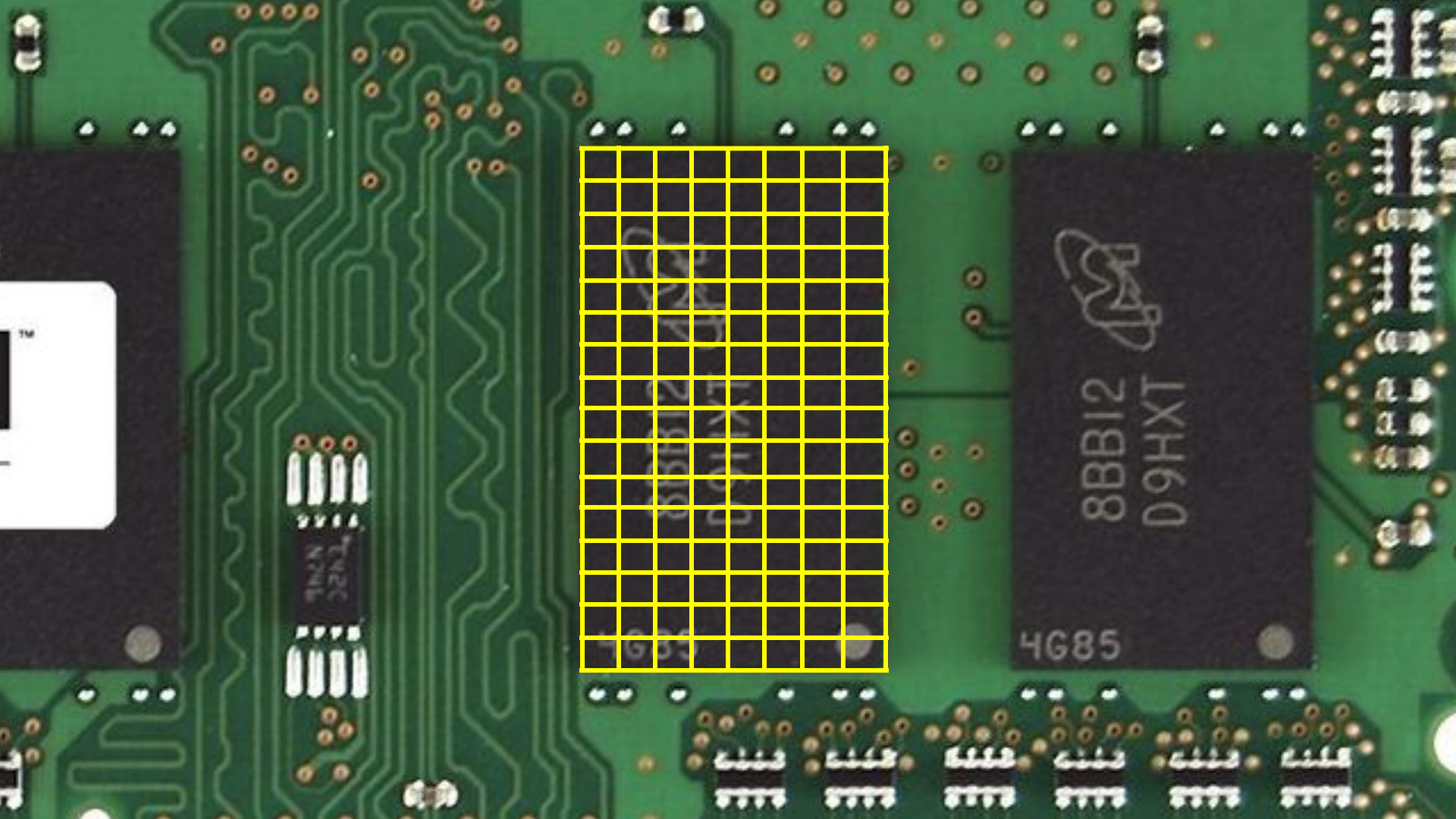
shorts

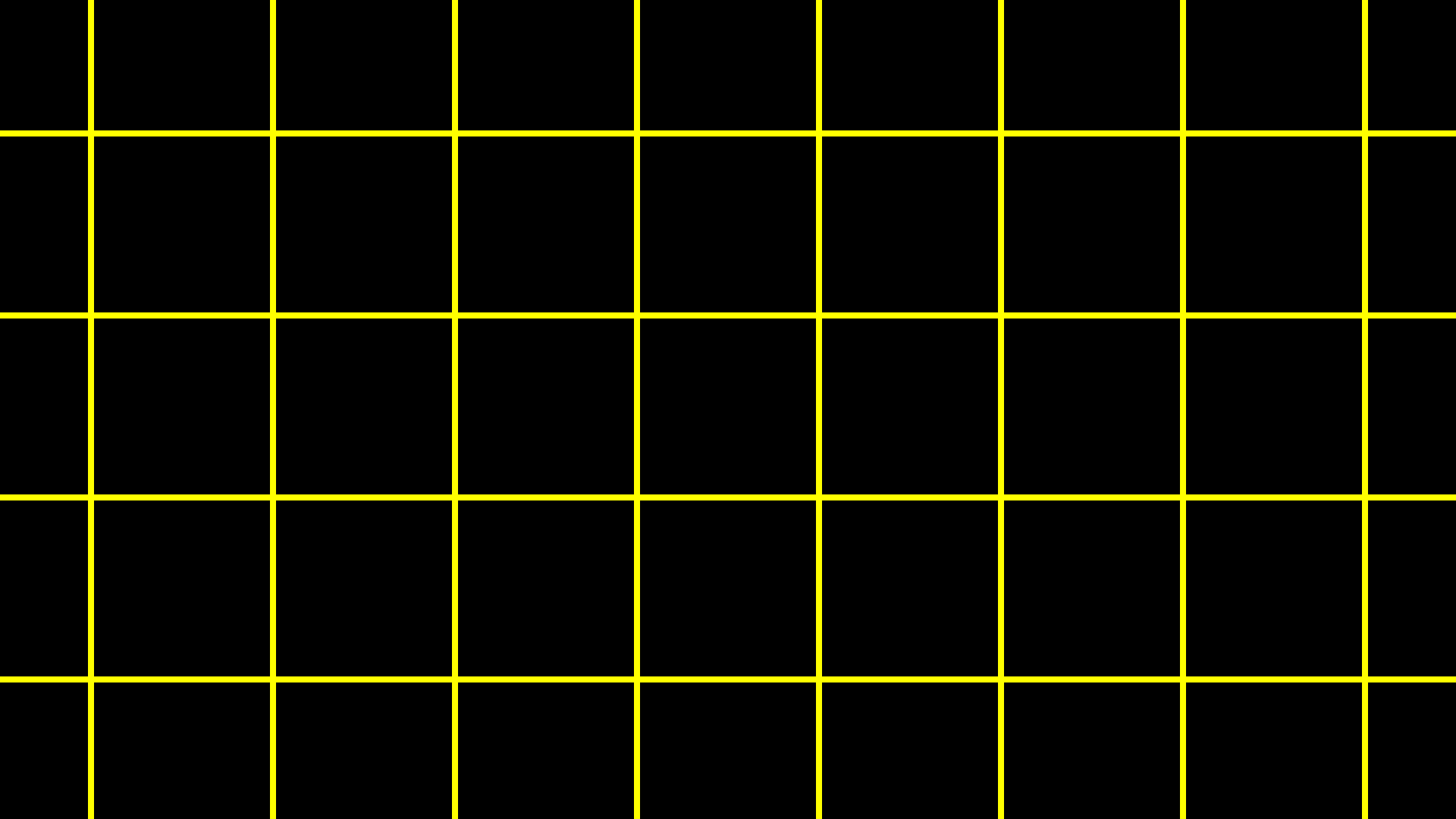

input →
→ output

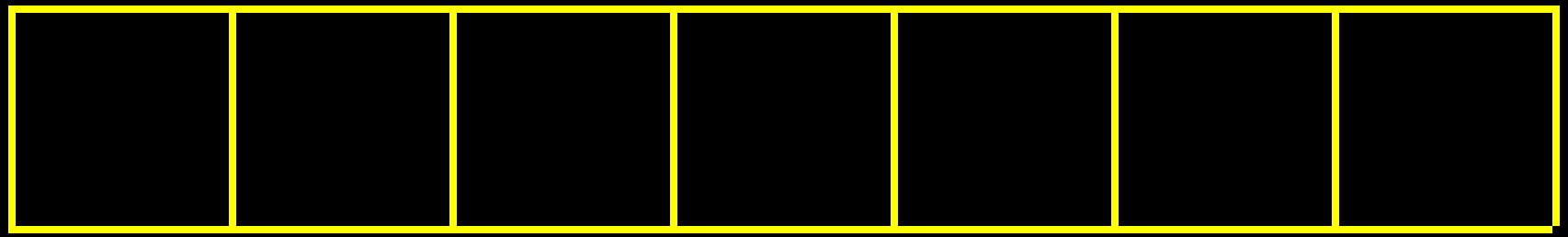

algorithms

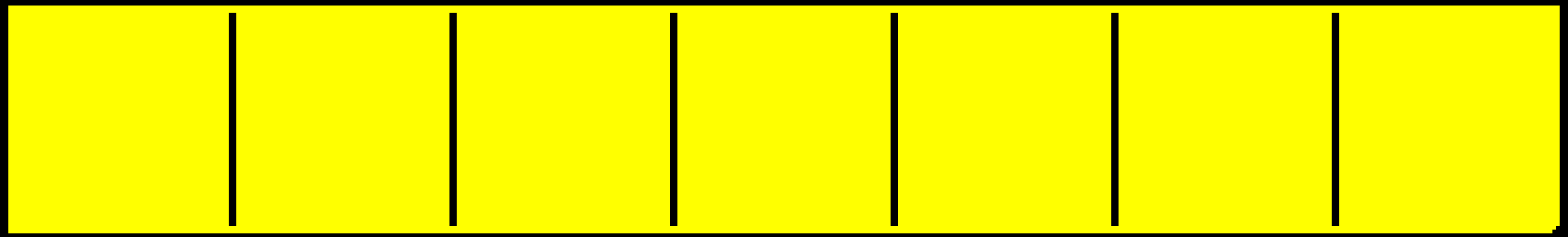

crucial
by Micron
8BB12
D9HXT
8BB12
D9HXT
47J6
4G85
4G85
4G85


8BBI2
D9HXT
4G85
8BBI2
D9HXT
4G85


8BBI2
D9HXT
4G85







linear search

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

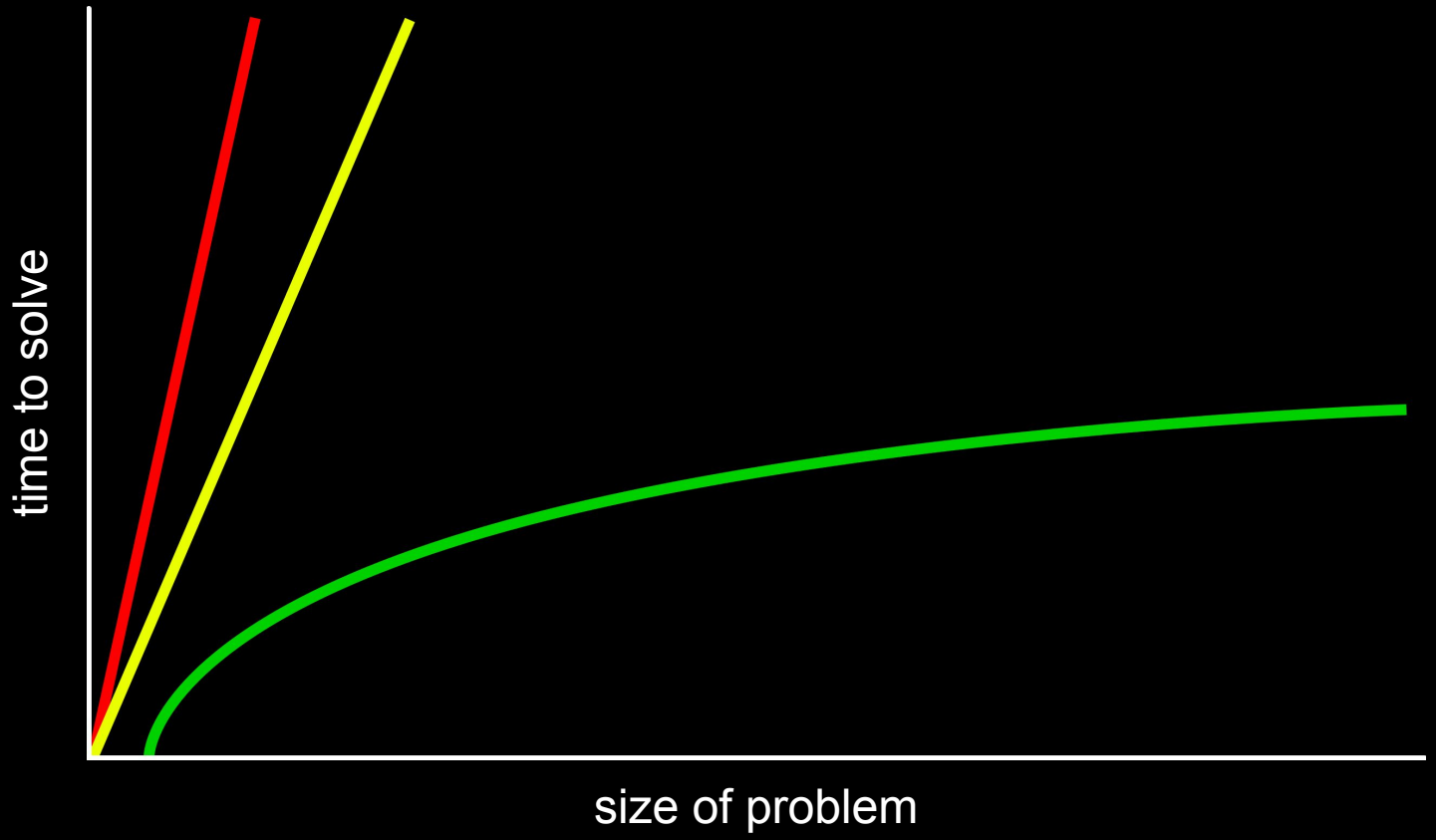
```
For each door from left to right
    If number is behind door
        Answer is true
    Else
        Answer is false
```

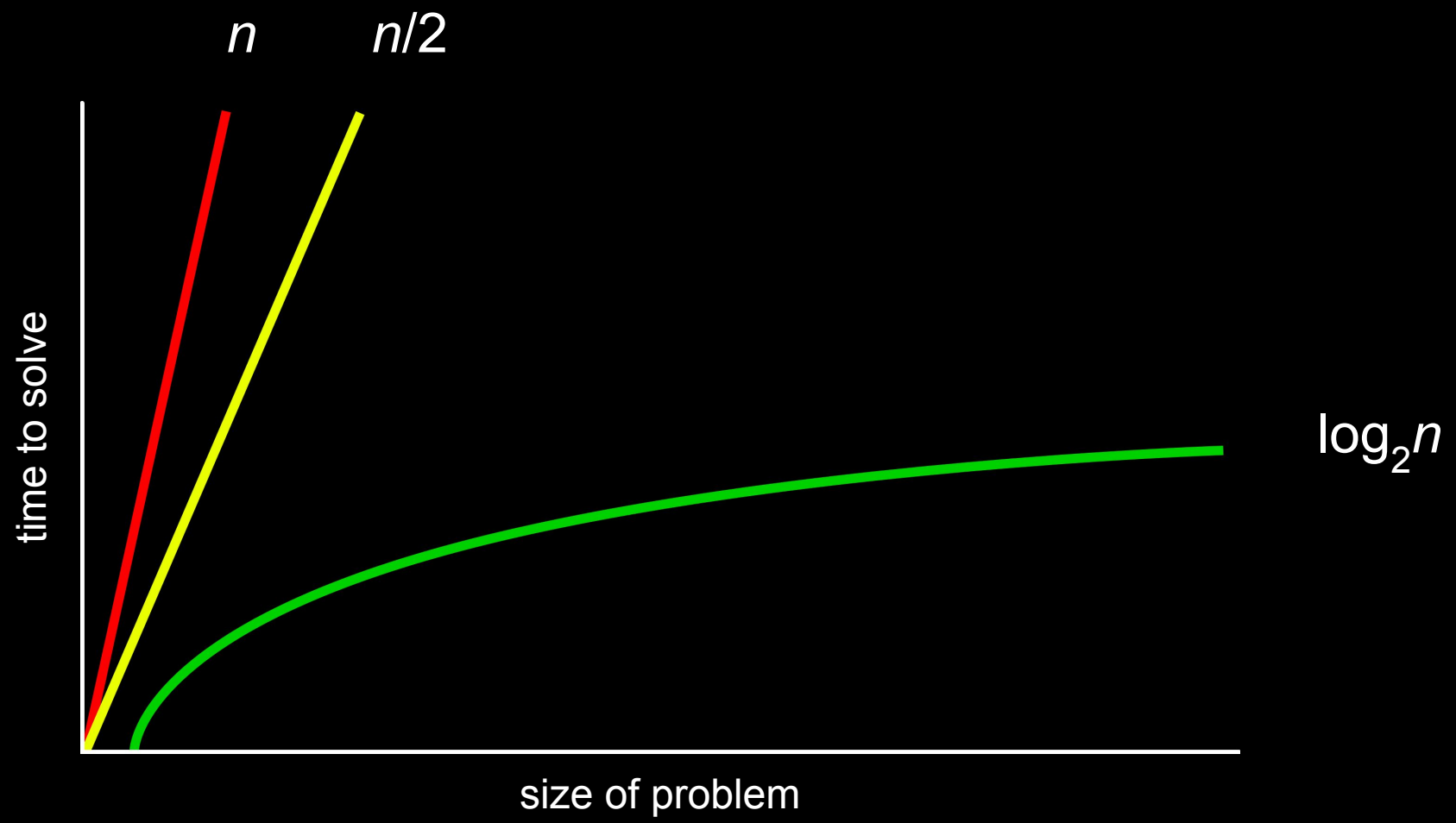
binary search

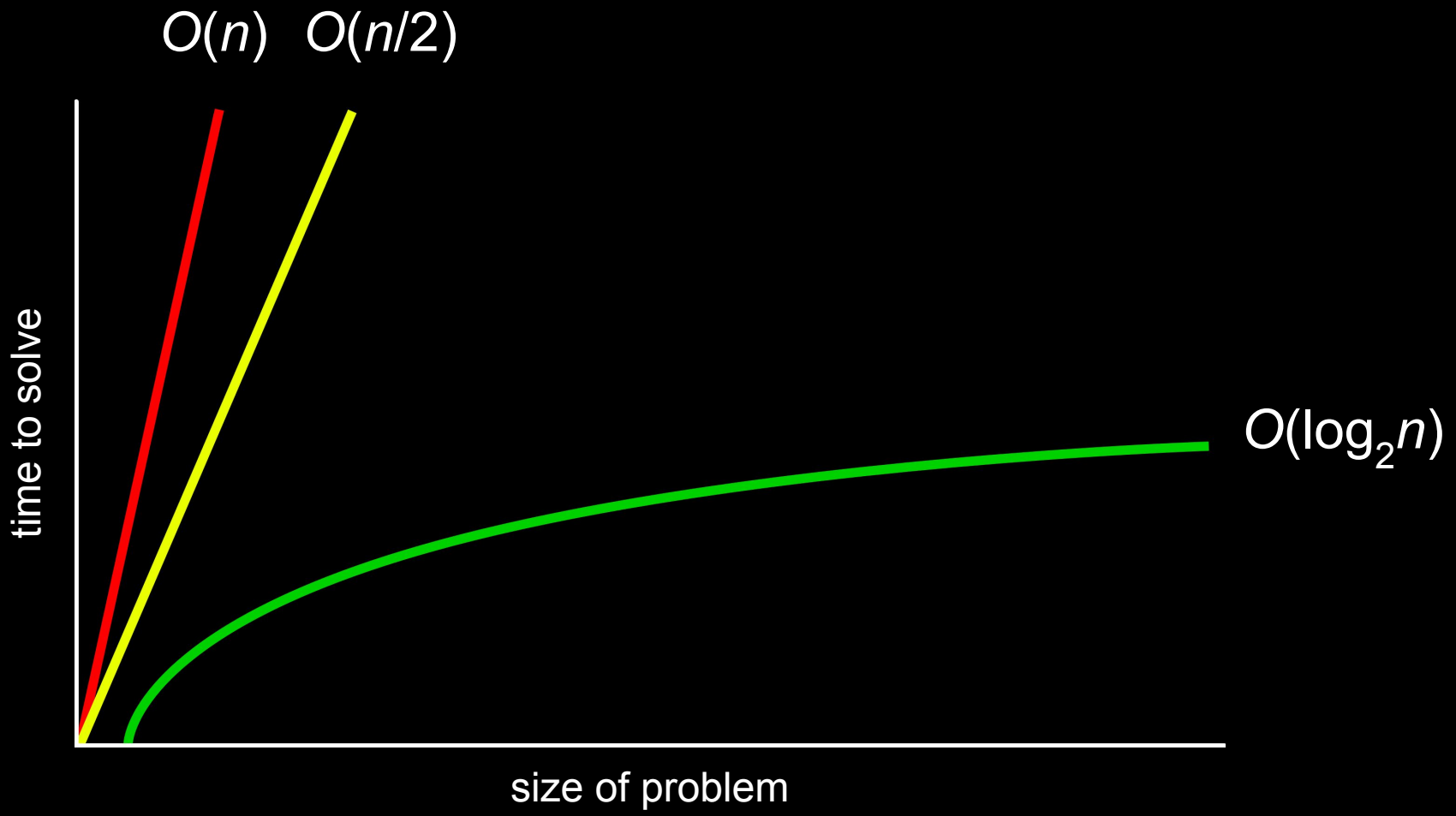
```
If number behind middle door
    Return true
Else if number < middle door
    Search left half
Else if number > middle door
    Search right half
```

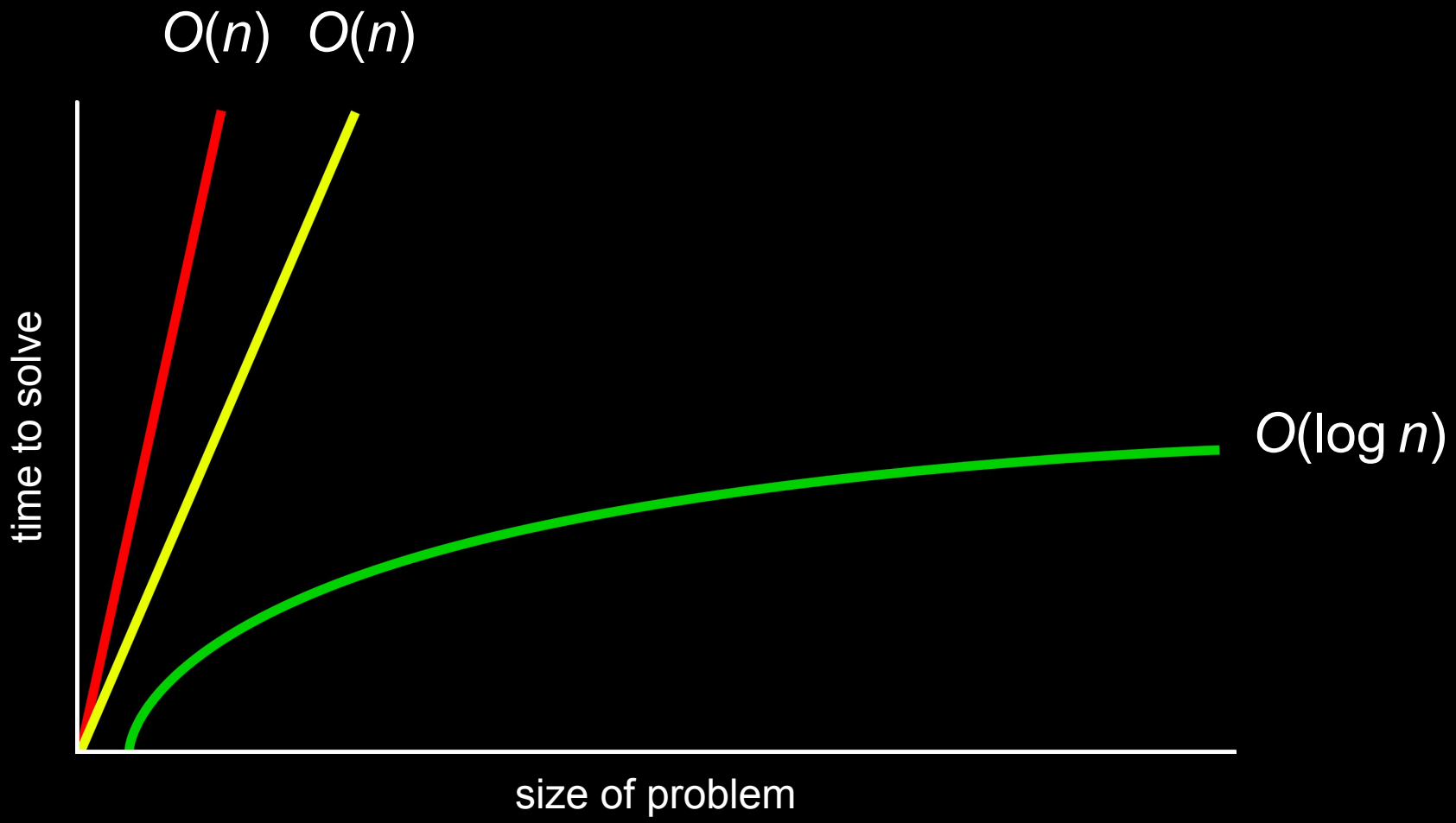
```
If no doors

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
    Return result of searching left half
Else if number > middle door
    Return result of searching right half
```


time to solve
size of problem


n
n/2
time to solve
log2n
size of problem


O(n)
O(n/2)
O(log2n)
time to solve
size of problem


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

$O(n^2)$

$O(n \log n)$

$O(n)$

$O(\log n)$

$O(1)$

$O(n^2)$

$O(n \log n)$

$O(n)$ linear search

$O(\log n)$ binary search

$O(1)$

$\Omega(n^2)$

$\Omega(n \log n)$

$\Omega(n)$

$\Omega(\log n)$

$\Omega(1)$

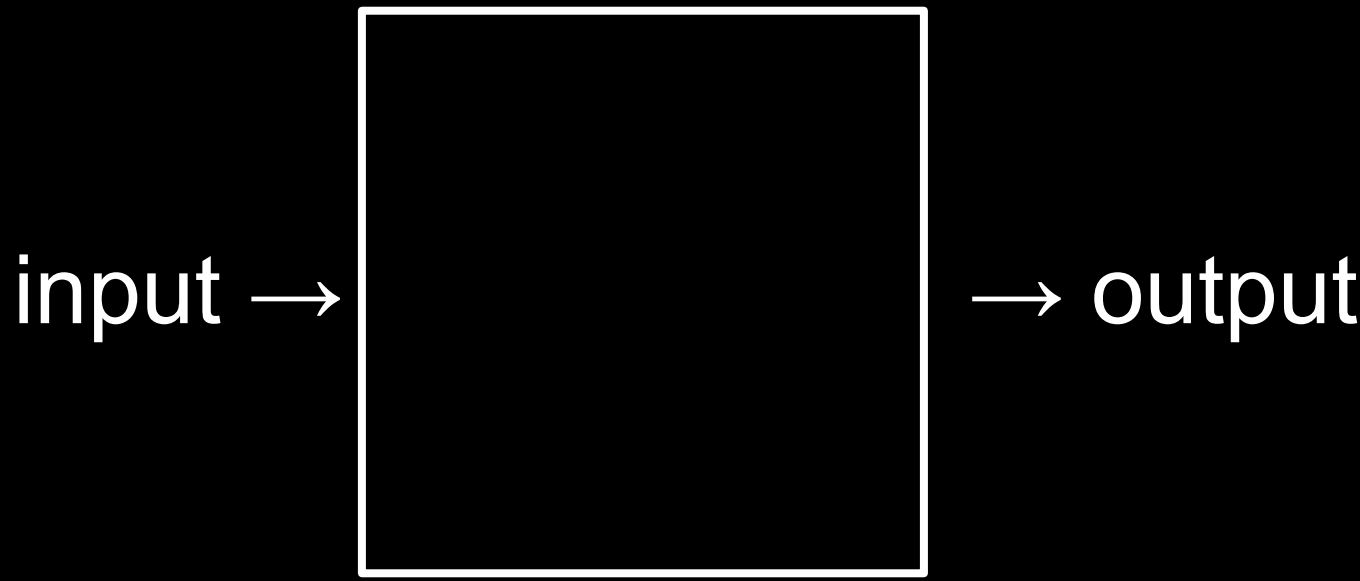
$\Omega(n^2)$

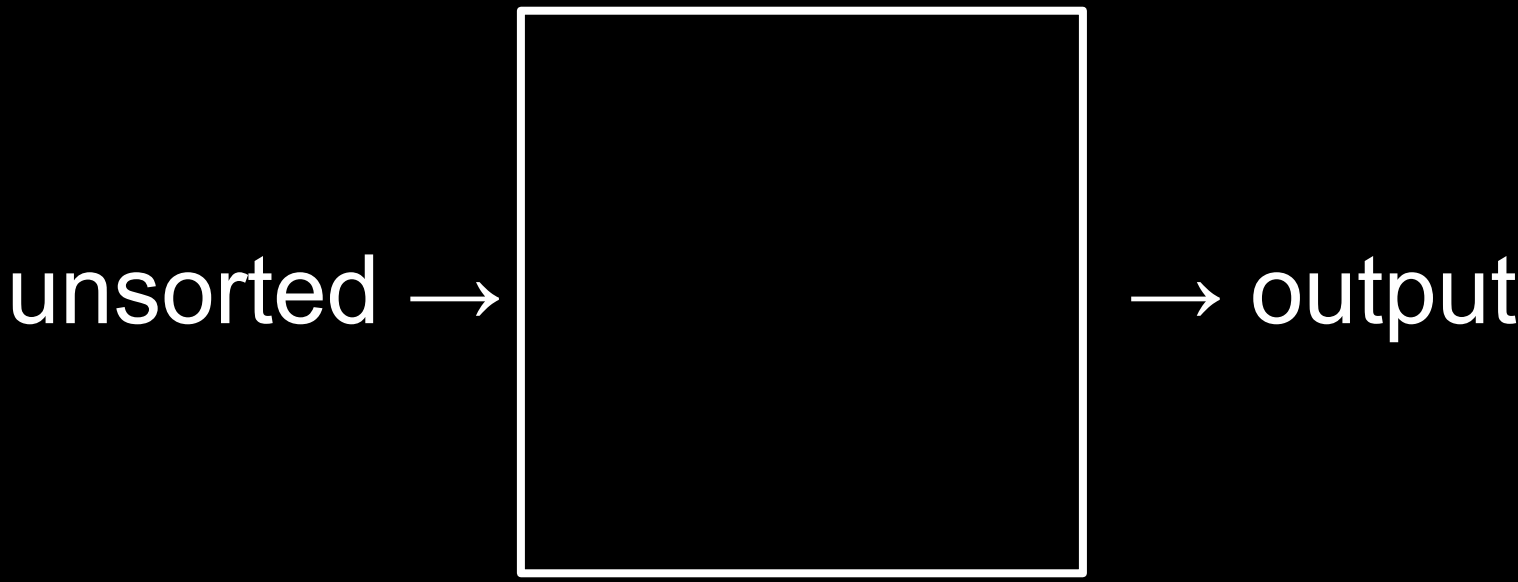
$\Omega(n \log n)$

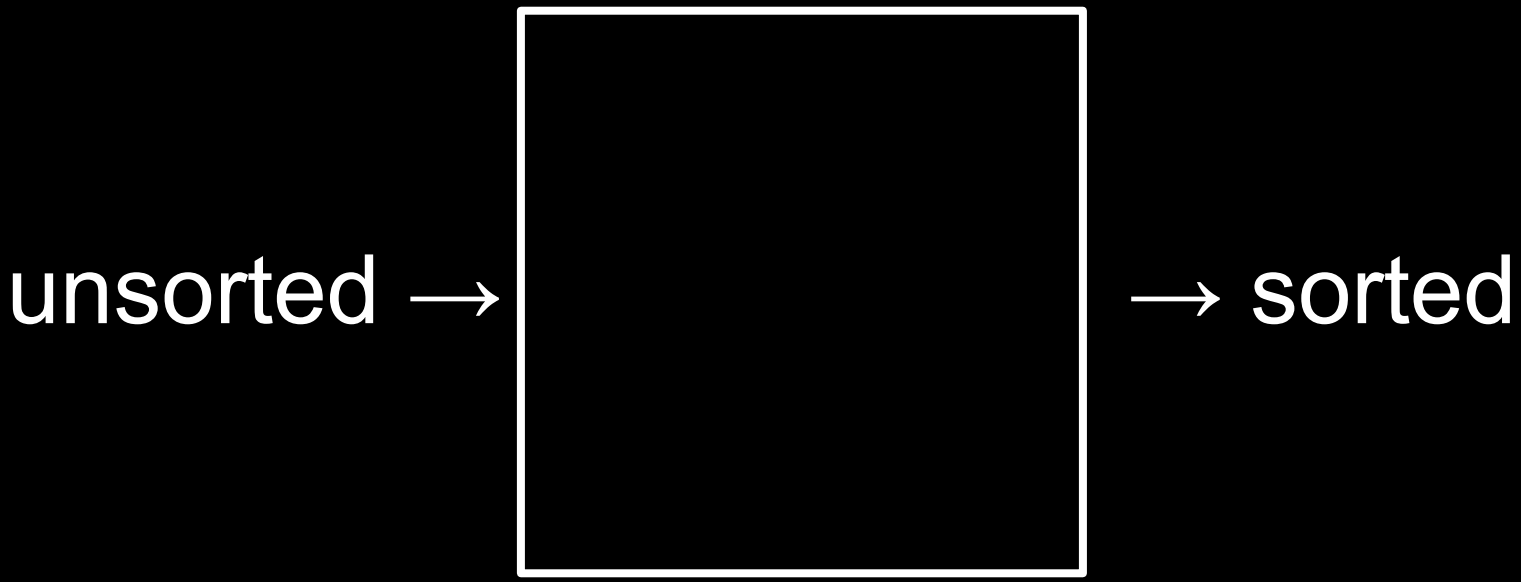
$\Omega(n)$

$\Omega(\log n)$

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


input →
→ output


unsorted →
→ output


unsorted →
→ sorted

6 3 8 5 2 7 4 1

selection sort

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

$(n - 1)$

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

$(n - 1) + (n - 2) + (n - 3)$

$(n - 1) + (n - 2) + (n - 3) + \ldots + 1$

$(n - 1) + (n - 2) + (n - 3) + \ldots + 1$

$n(n - 1)/2$

$(n - 1) + (n - 2) + (n - 3) + \ldots + 1$

$n(n - 1)/2$

$(n^2 - n)/2$

$(n - 1) + (n - 2) + (n - 3) + \ldots + 1$

$n(n - 1)/2$

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

$n^2/2 - n/2$

$(n - 1) + (n - 2) + (n - 3) + \ldots + 1$

$n(n - 1)/2$

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

$n^2/2 - n/2$

$O(n^2)$

$O(n^2)$ selection sort

$O(n \log n)$

$O(n)$

$O(\log n)$

$O(1)$

$\Omega(n^2)$ selection sort

$\Omega(n \log n)$

$\Omega(n)$

$\Omega(\log n)$

$\Omega(1)$

6 3 8 5 2 7 4 1

bubble sort

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

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

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

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

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

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

$n^2 - 2n + 1$

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

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

$n^2 - 2n + 1$

$O(n^2)$

$O(n^2)$ bubble sort

$O(n \log n)$

$O(n)$

$O(\log n)$

$O(1)$

$\Omega(n^2)$ bubble sort

$\Omega(n \log n)$

$\Omega(n)$

$\Omega(\log n)$

$\Omega(1)$

bubble sort

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

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

$O(n^2)$ bubble sort

$O(n \log n)$

$O(n)$

$O(\log n)$

$O(1)$

$\Omega(n^2)$

$\Omega(n \log n)$

$\Omega(n)$ bubble sort

$\Omega(\log n)$

$\Omega(1)$


Google

recursion

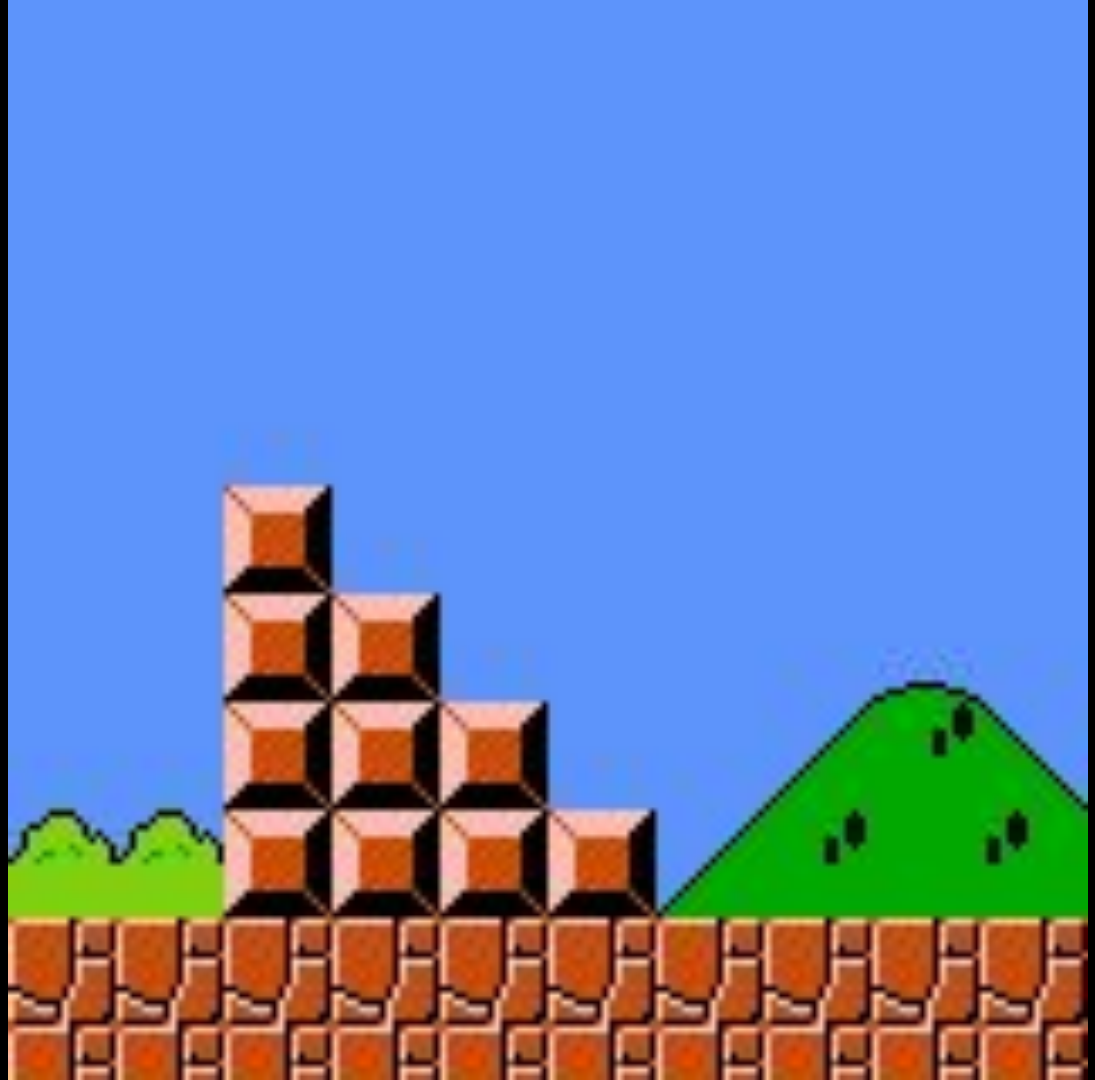
```
1   Pick up phone book
2   Open to middle of phone book
3   Look at page
4   If person is on page
5       Call person
6   Else if person is earlier in book
7       Open to middle of left half of book
8       Go back to line 3
9   Else if person is later in book
10      Open to middle of right half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

```
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

```
If only one item
    Return
Else
    Sort left half of items
    Sort right half of items
    Merge sorted halves
```

```
If only one item
    Return
Else
    Sort left half of items
    Sort right half of items
    Merge sorted halves
```

7 4 5 2 6 3 8 1

7 4 5 2 6 3 8 1

7 4 5 2 6 3 8 1

7 4 5 2 6 3 8 1

7 4 5 2 6 3 8 1


7 4 5 2 6 3 8 1

7

5 2 6 3 8 1

4

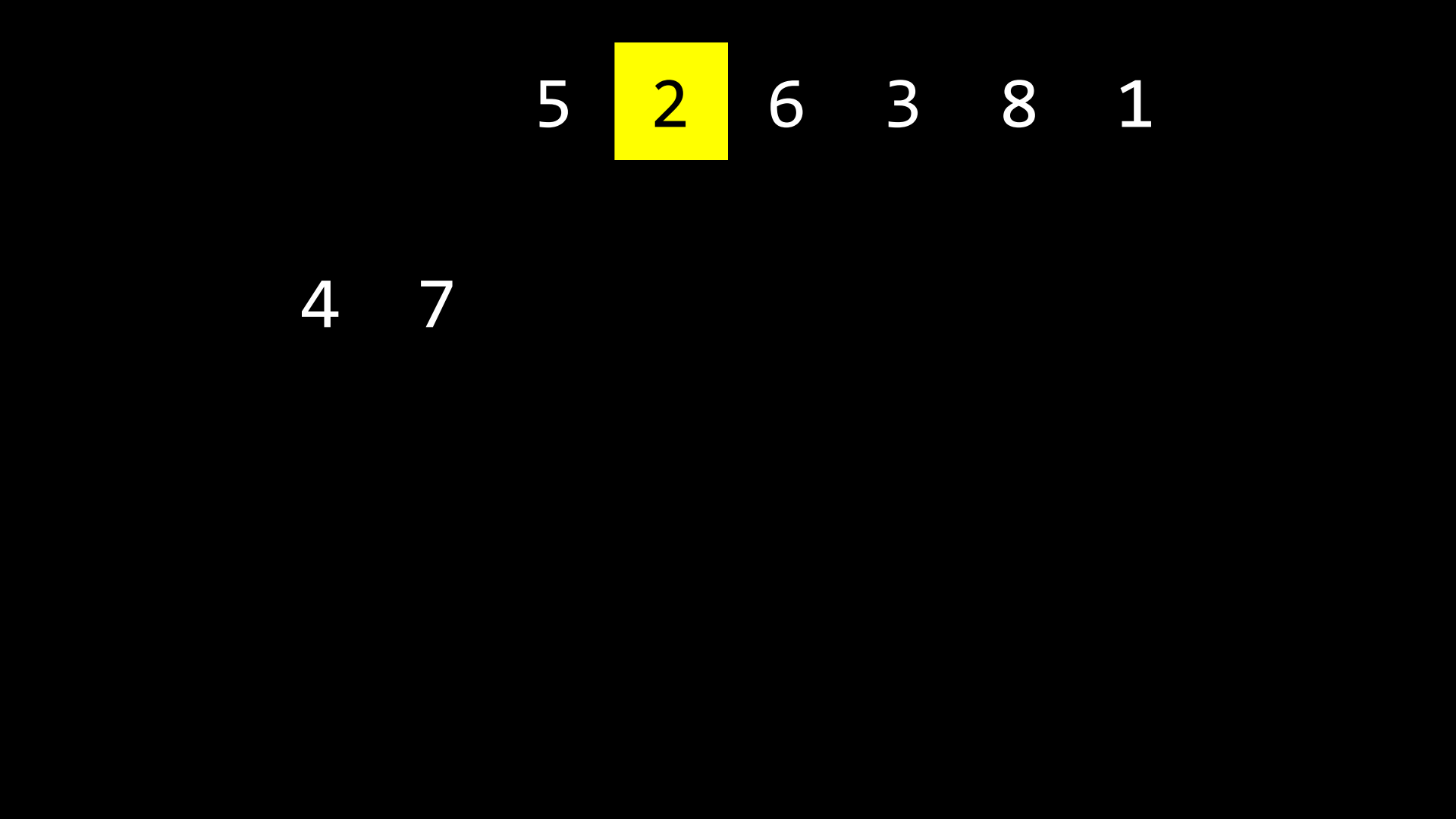
5 2 6 3 8 1

4 7

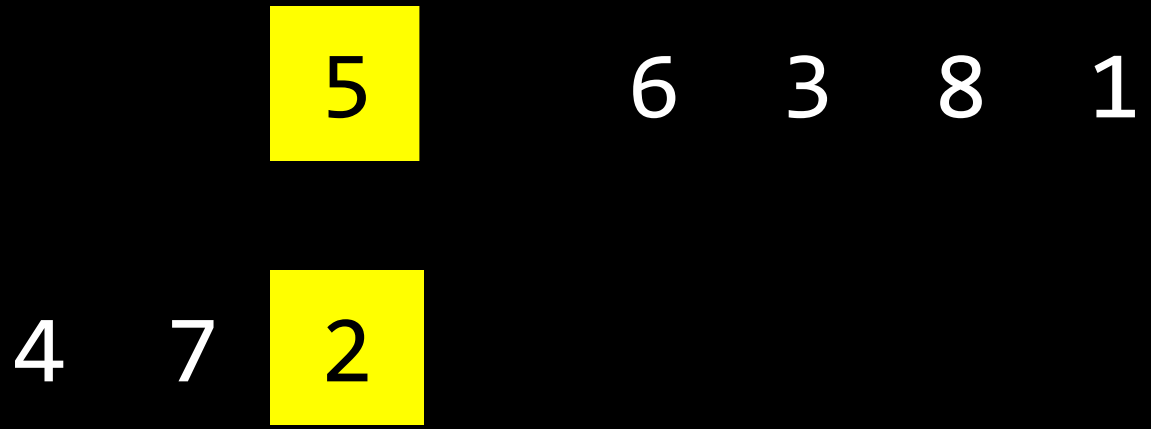
5 2 6 3 8 1

4 7

4 7


5 2 6 3 8 1
4 7

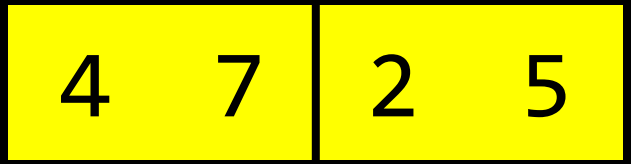
4 7


5
6 3 8 1
4 7 2

6 3 8 1

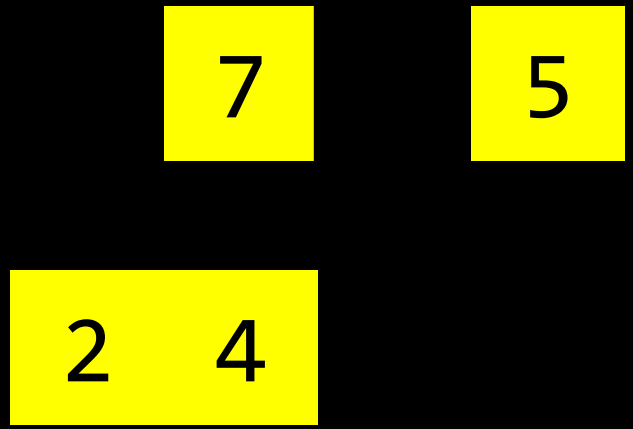
4 7 2 5

6 3 8 1

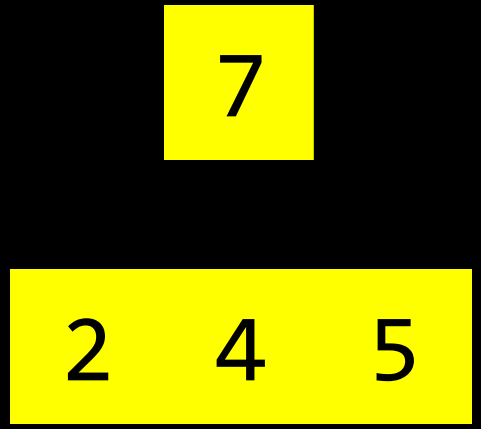


6 3 8 1

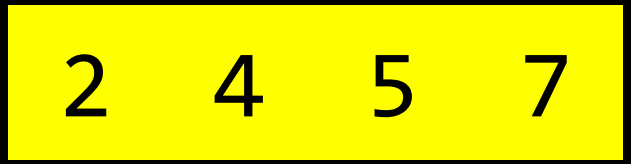
6 3 8 1



6 3 8 1



6 3 8 1



6 3 8 1

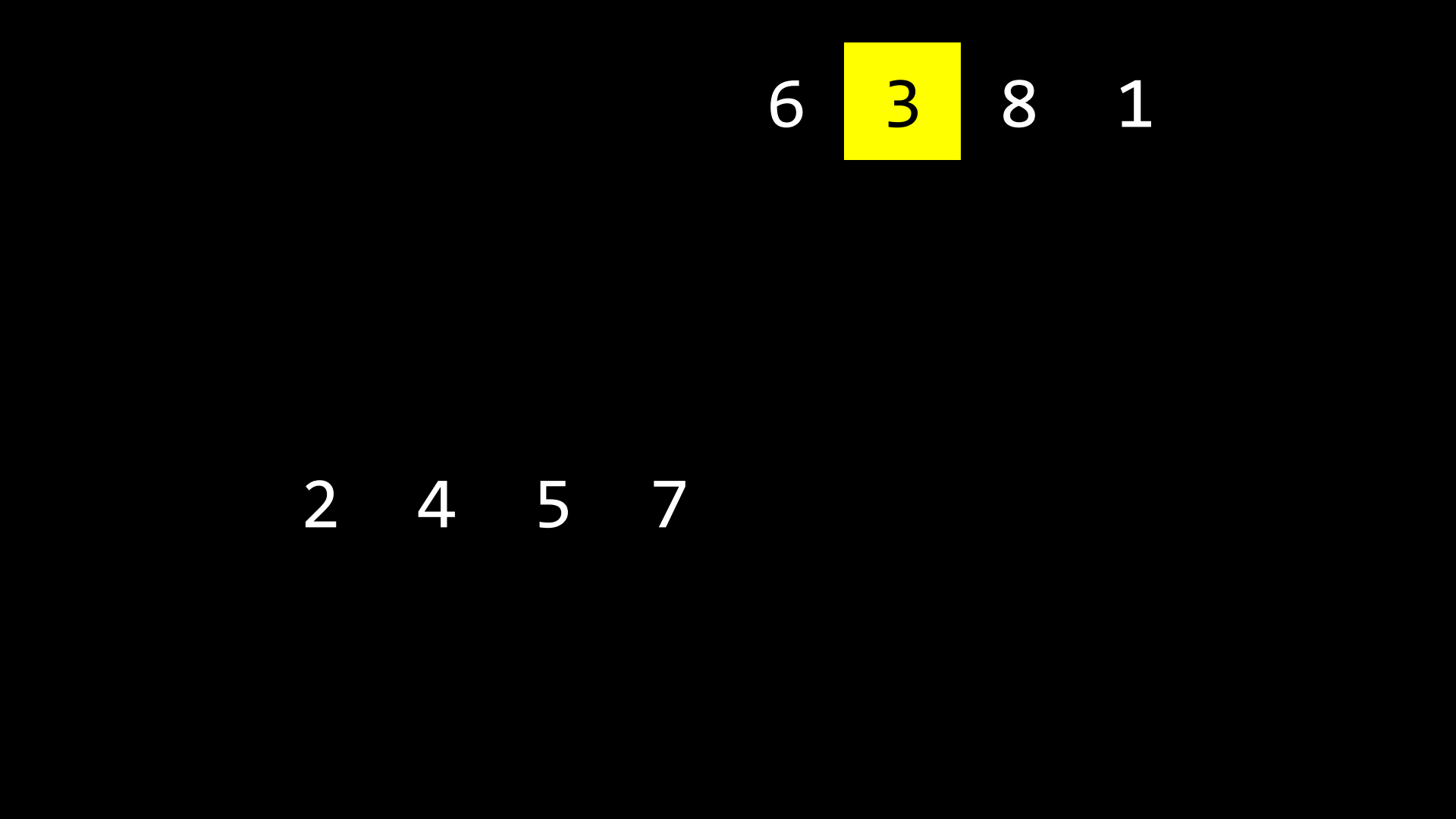
2 4 5 7

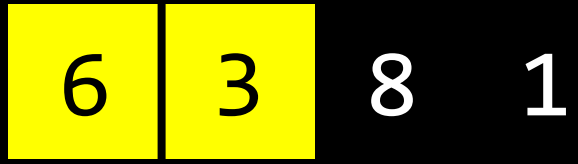
6 3 8 1

2 4 5 7

6 3 8 1

2 4 5 7


6 3 8 1
2 4 5 7



2 4 5 7

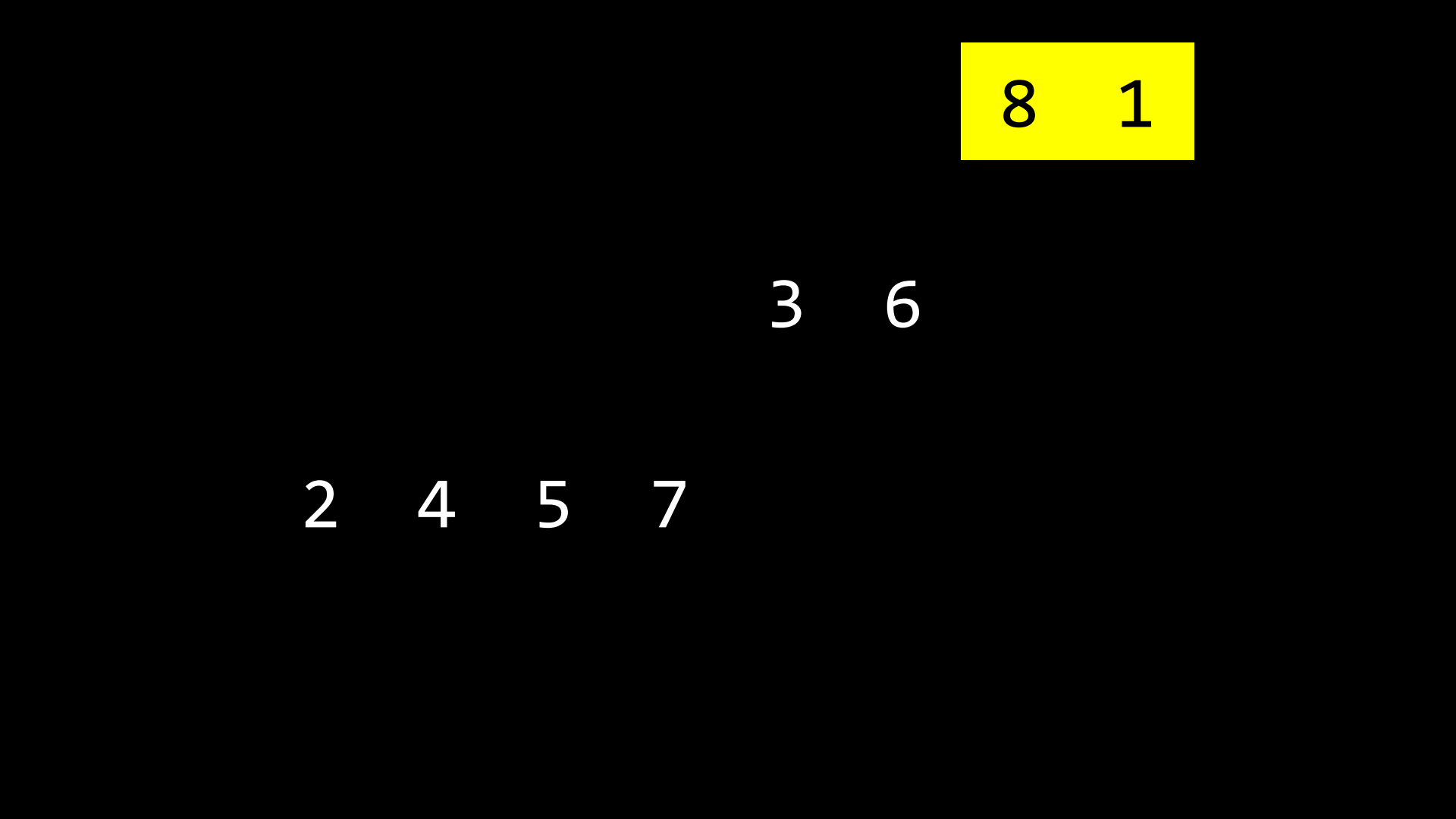
8 1

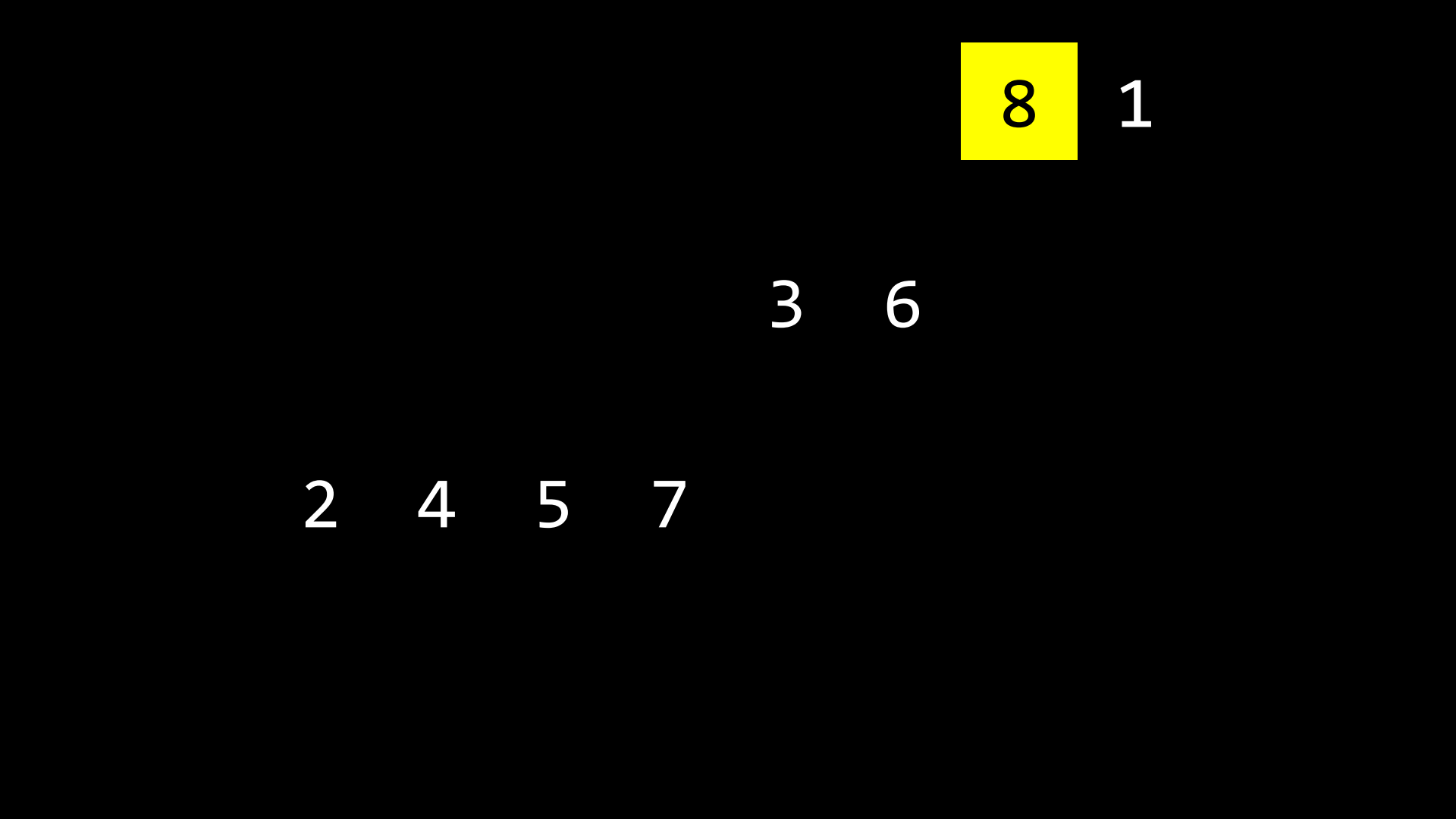
2 4 5 7

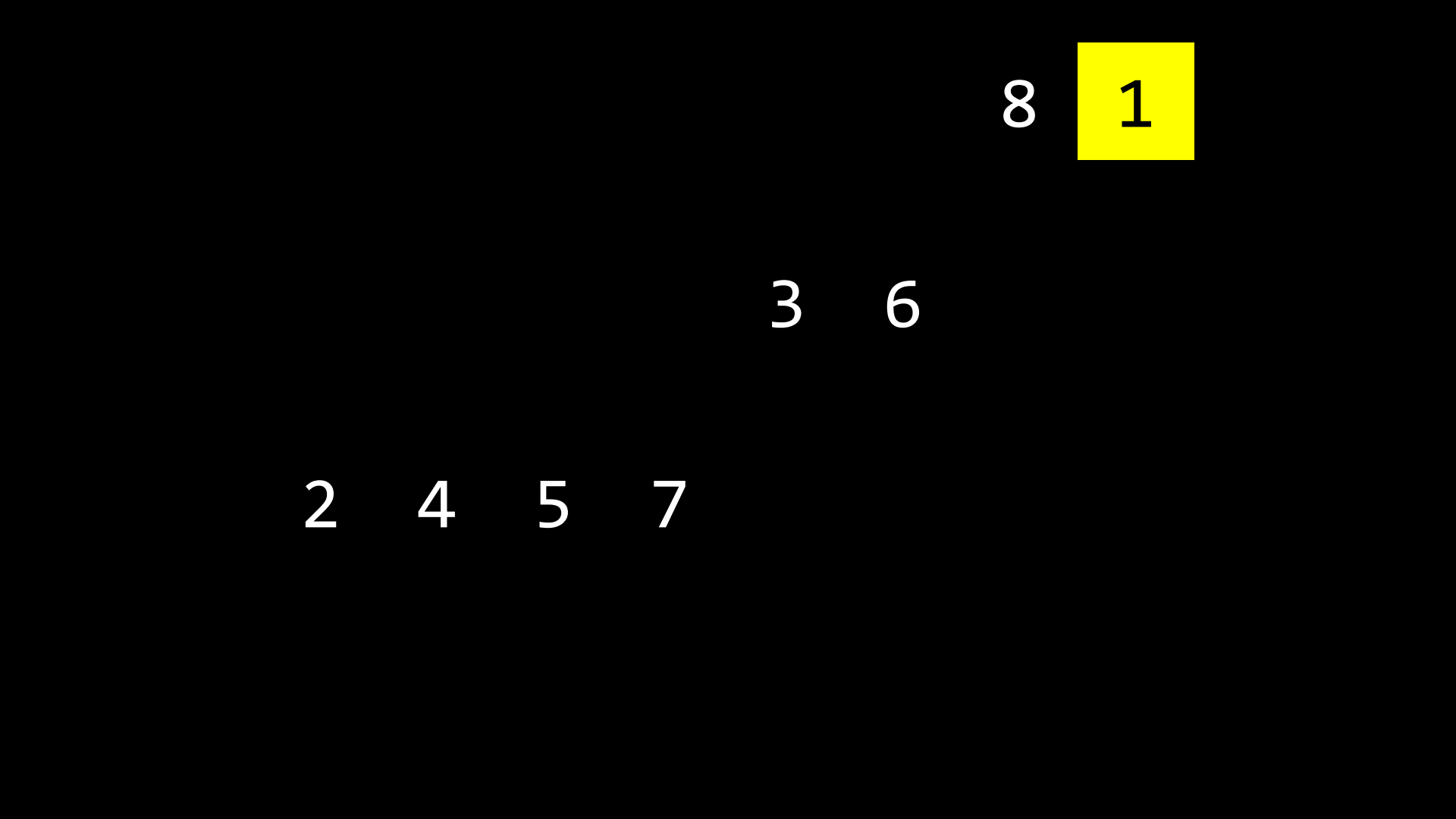
8 1

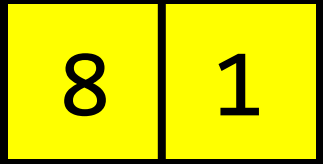
3 6

2 4 5 7


8 1
3 6
2 4 5 7

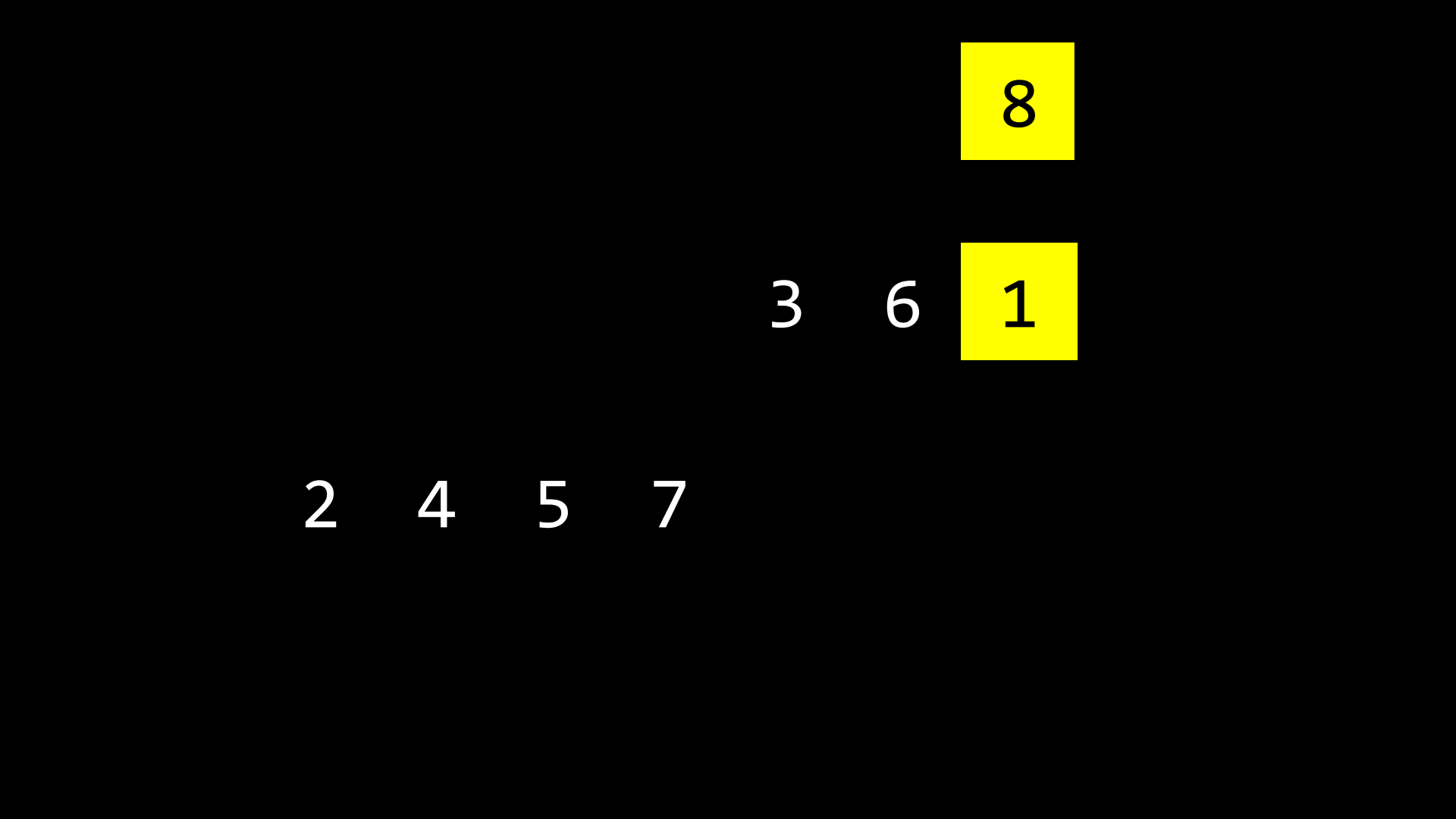

8
1
3
6
2
4
5
7


8
1
3
6
2
4
5
7



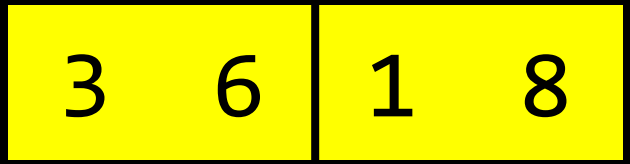
3 6

2 4 5 7

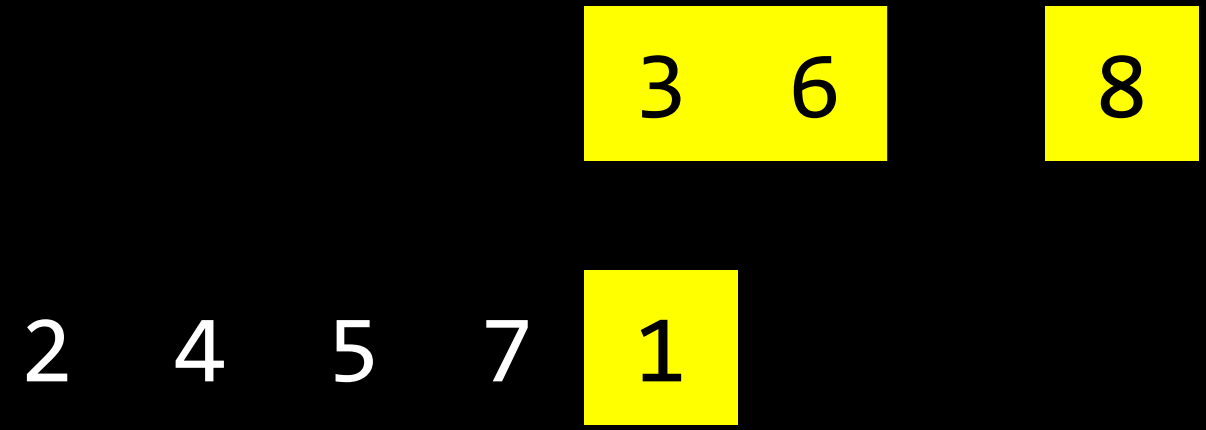

8
3 6 1
2 4 5 7

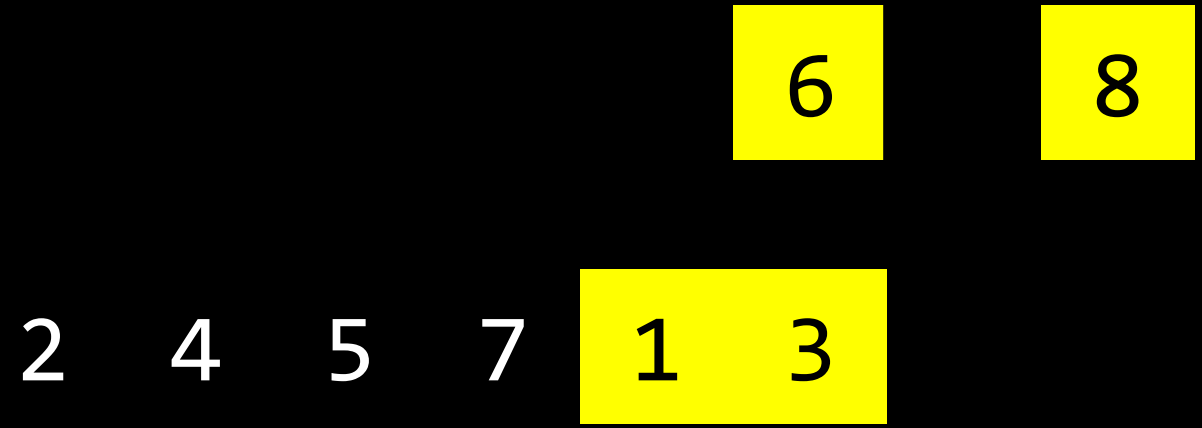
3 6 1 8

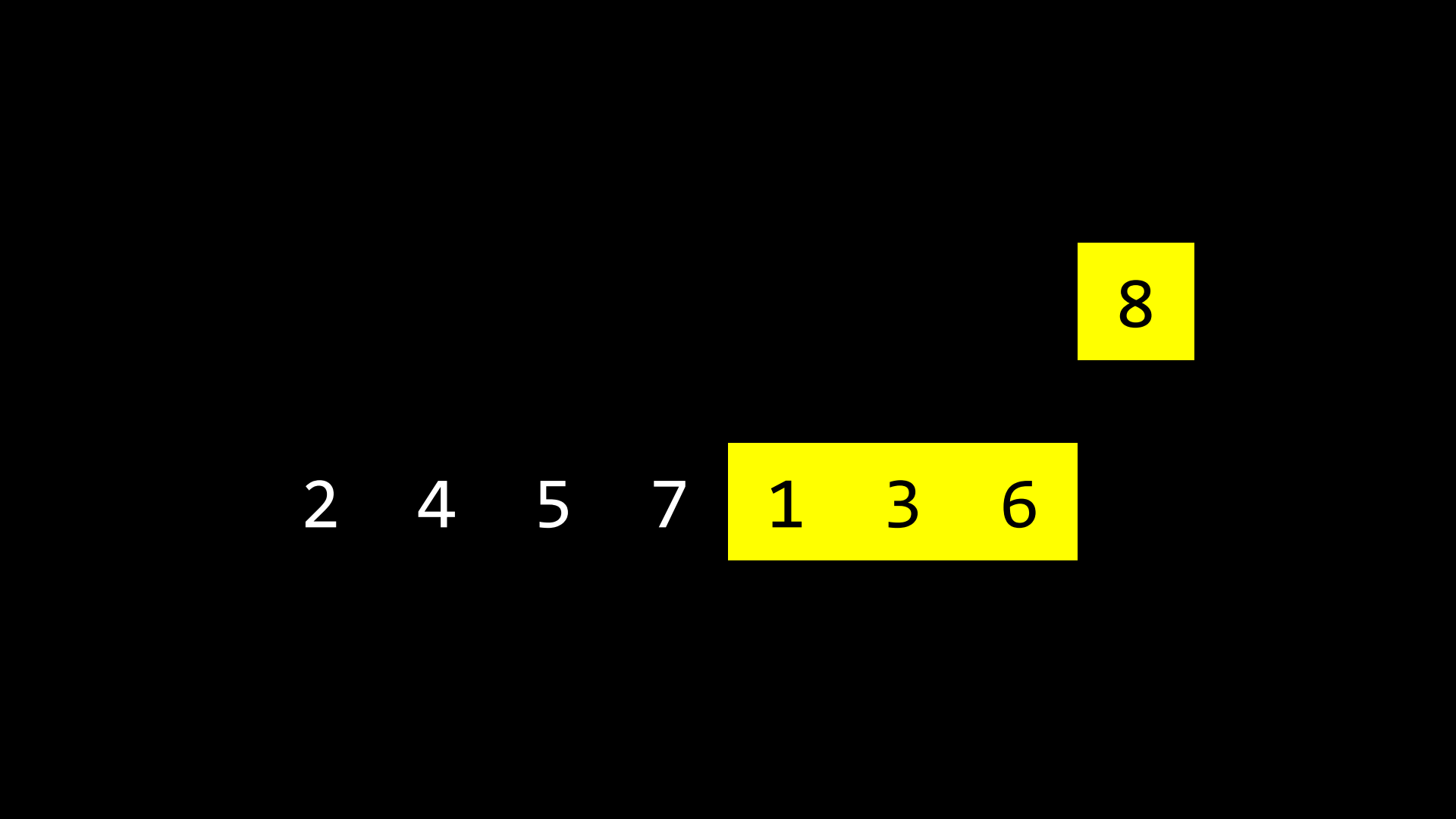
2 4 5 7

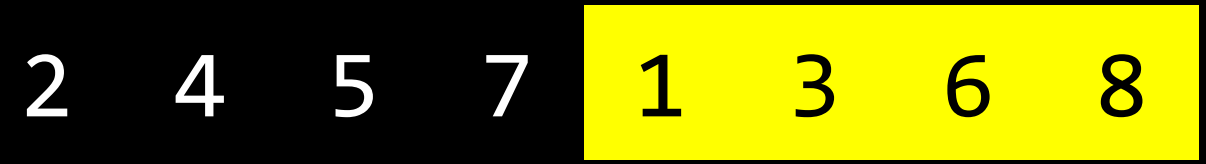


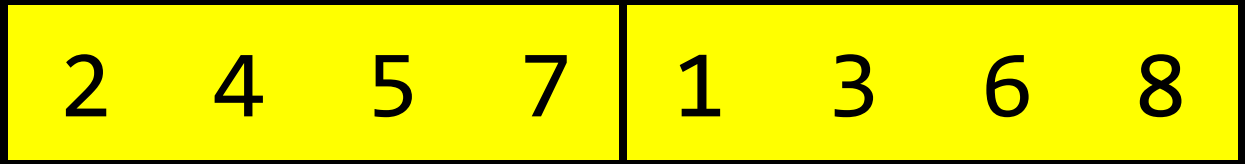
2 4 5 7

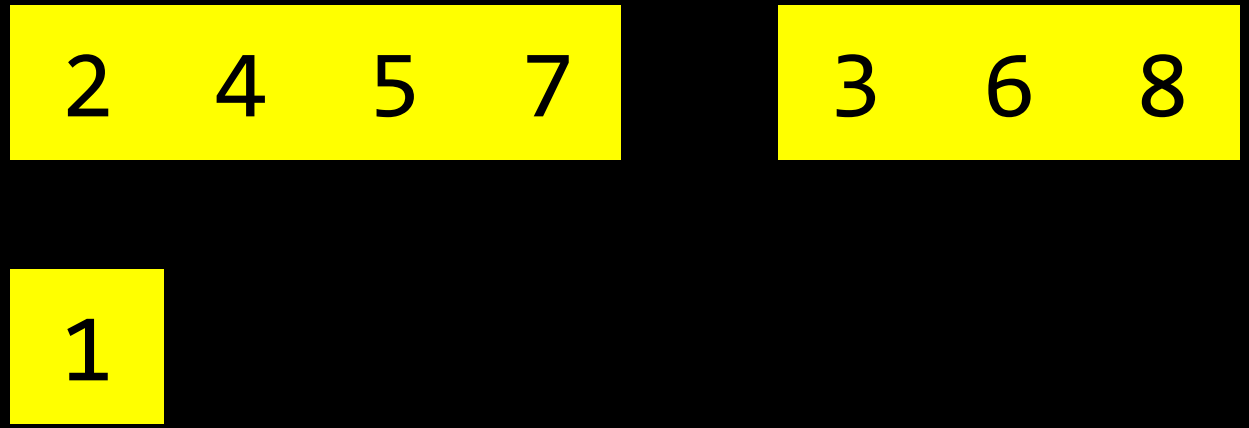

3 6
8
2 4 5 7
1

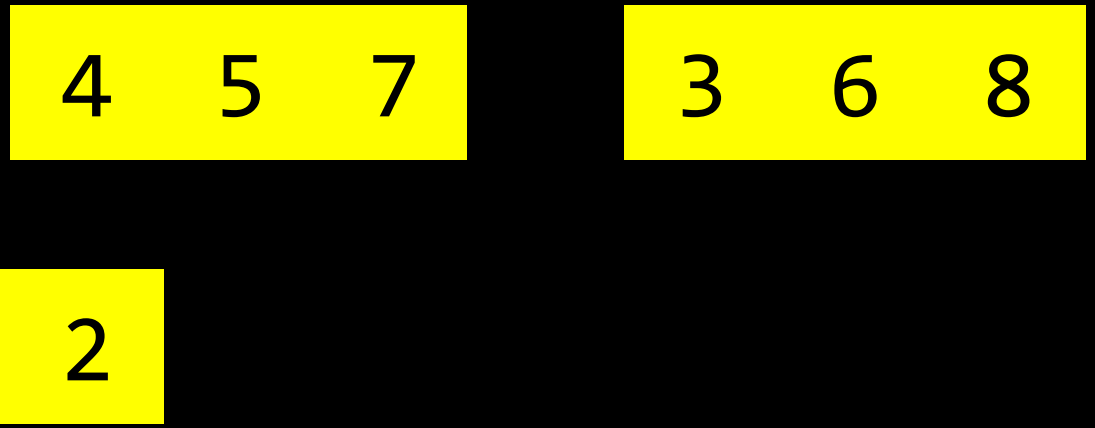

6
8
2
4
5
7
1
3

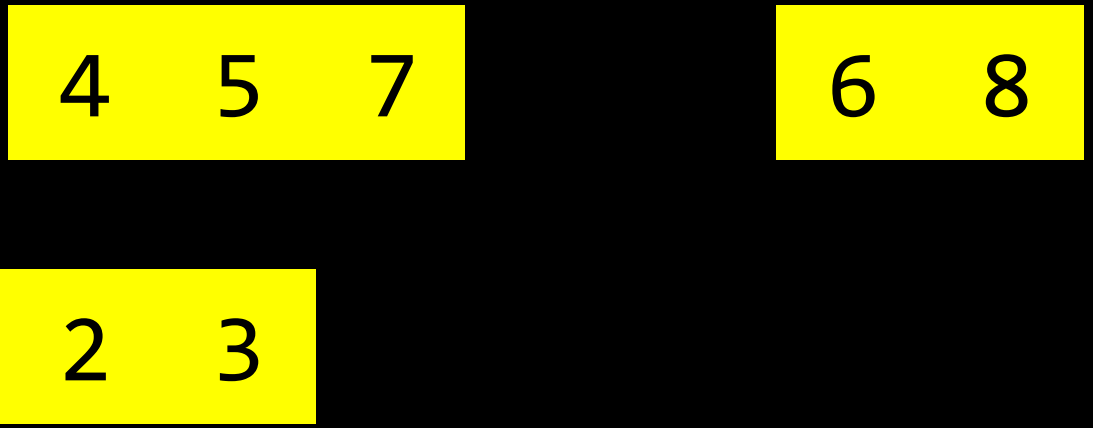

8
2 4 5 7 1 3 6

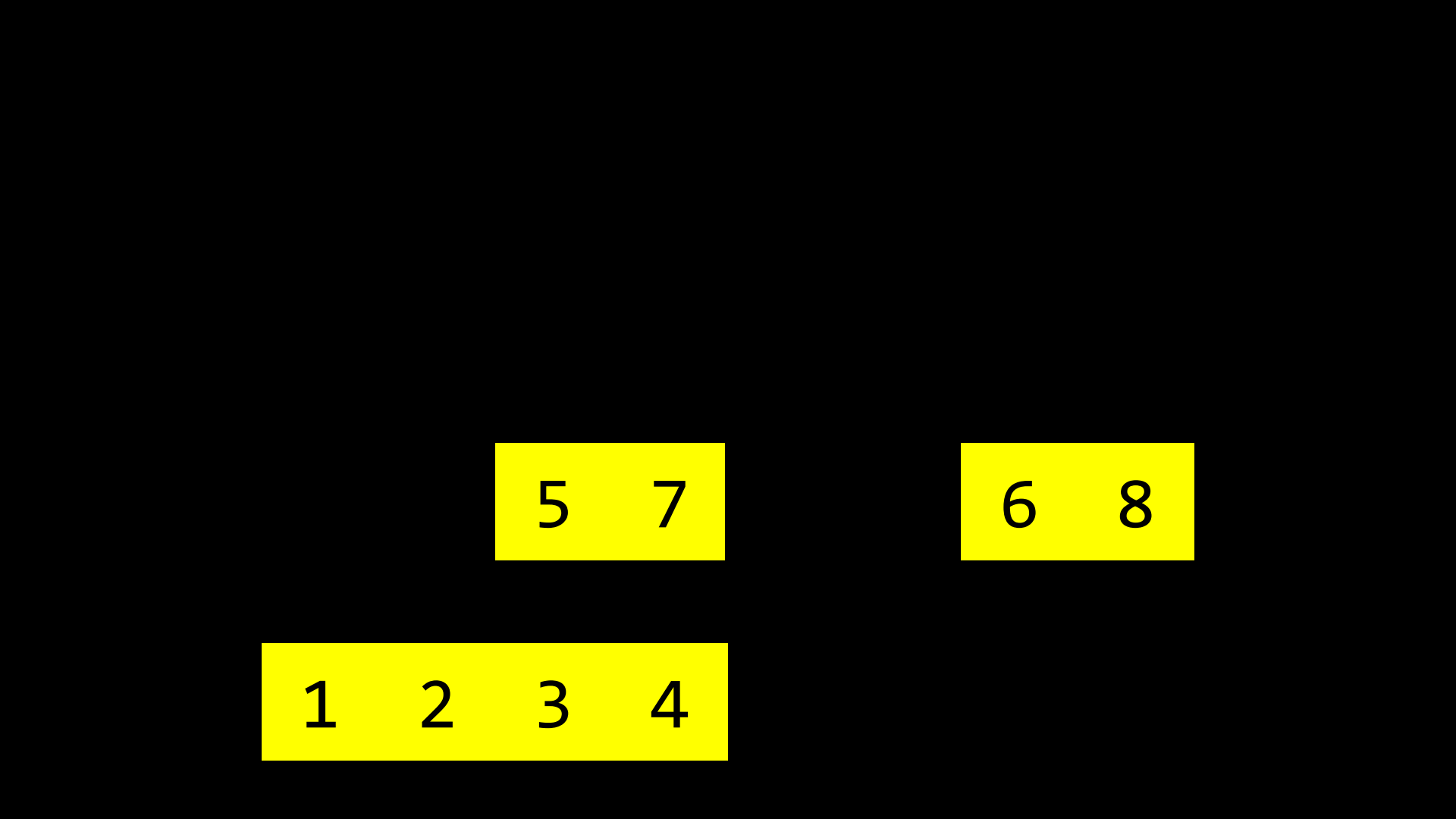

2 4 5 7 1 3 6 8

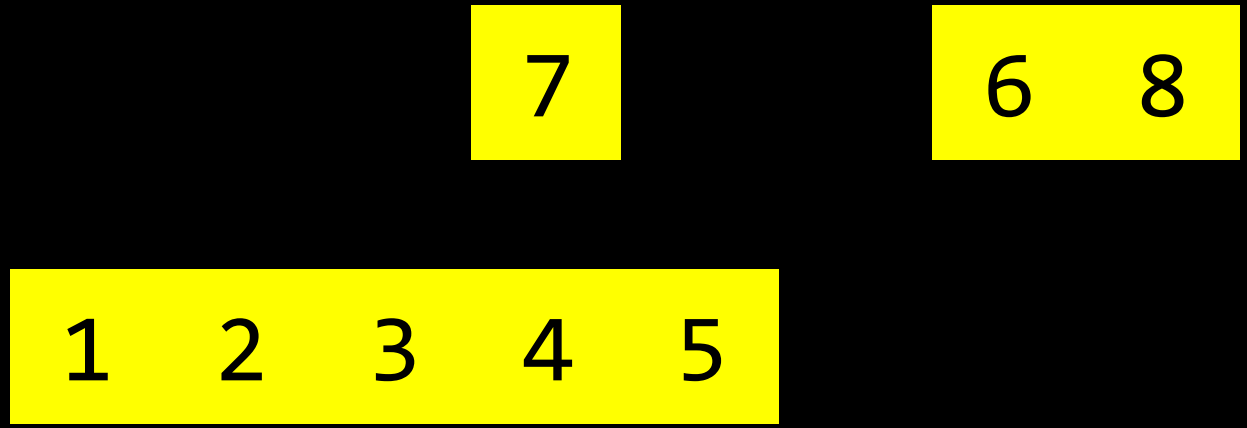

2 4 5 7 1 3 6 8

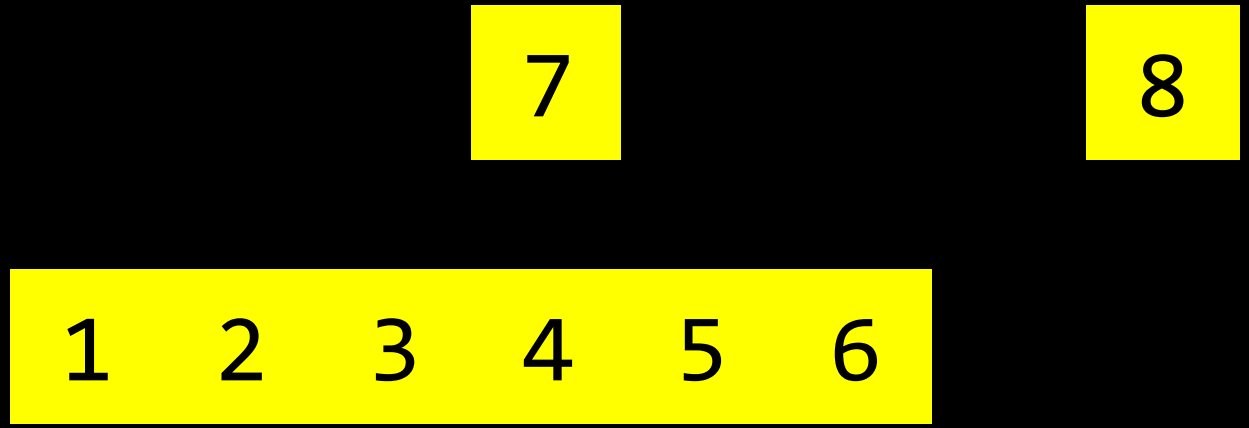

2 4 5 7
3 6 8
1

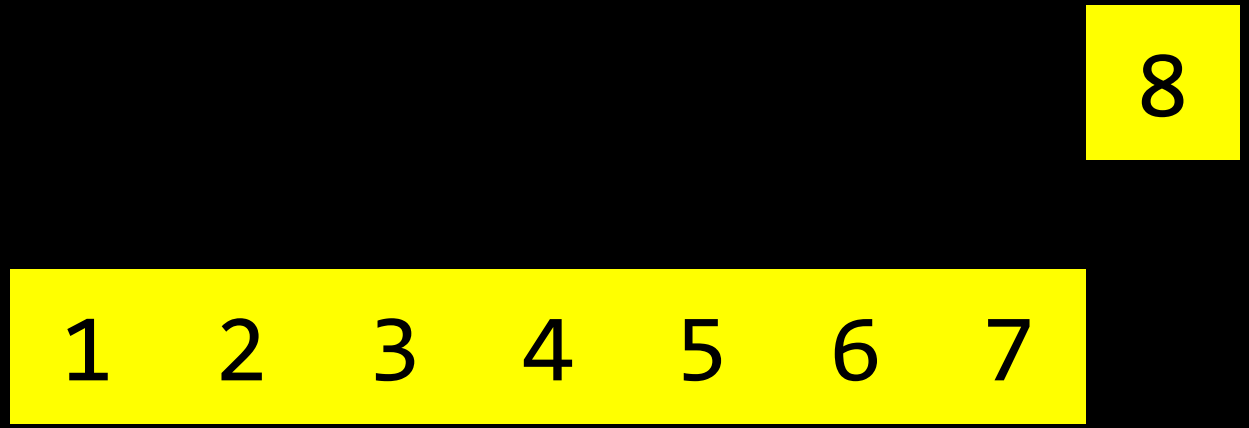

4 5 7
3 6 8
1 2


4 5 7
6 8
1 2 3

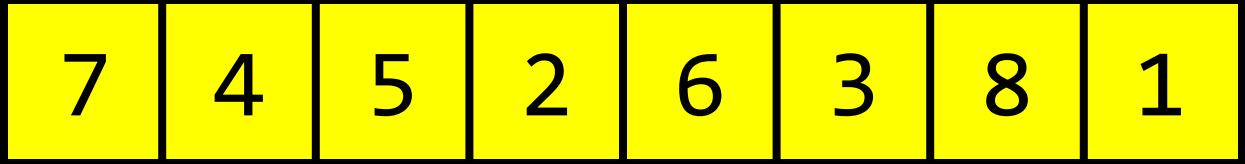

5 7
6 8
1 2 3 4

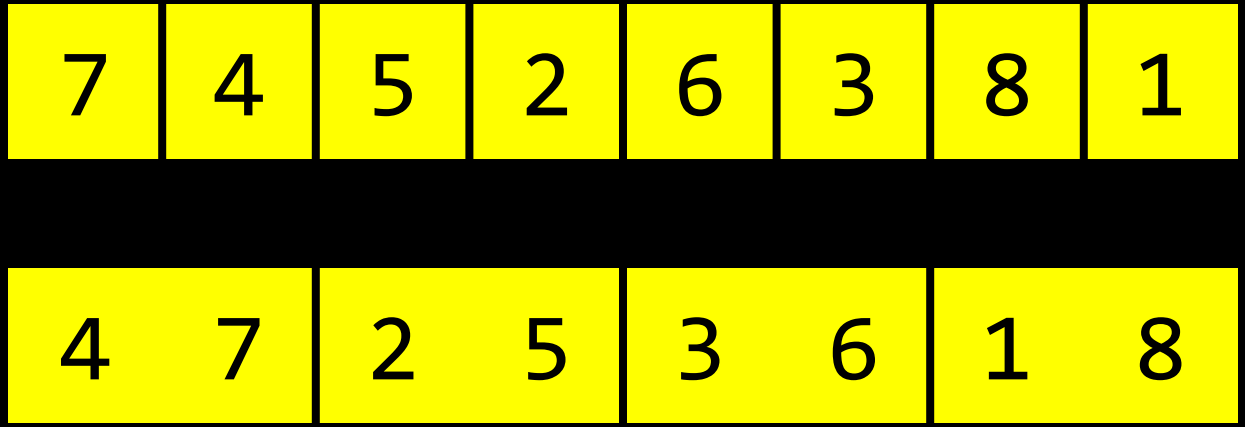

7
6 8
1 2 3 4 5

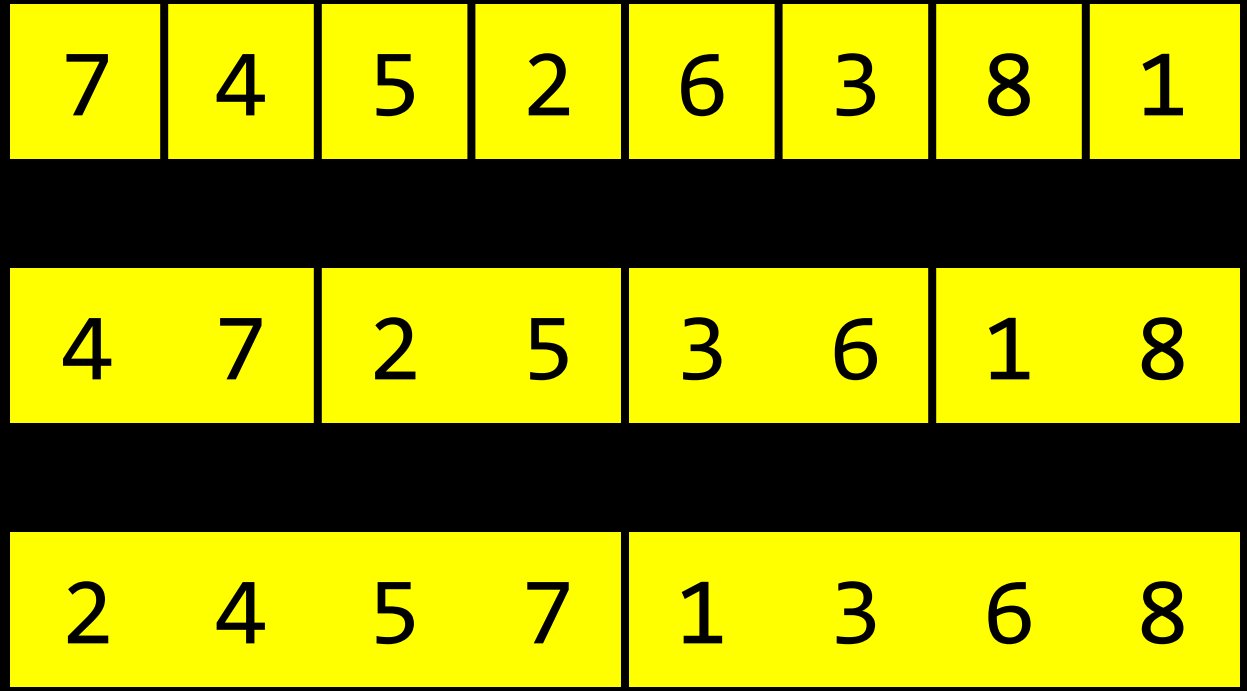

7
8
1 2 3 4 5 6

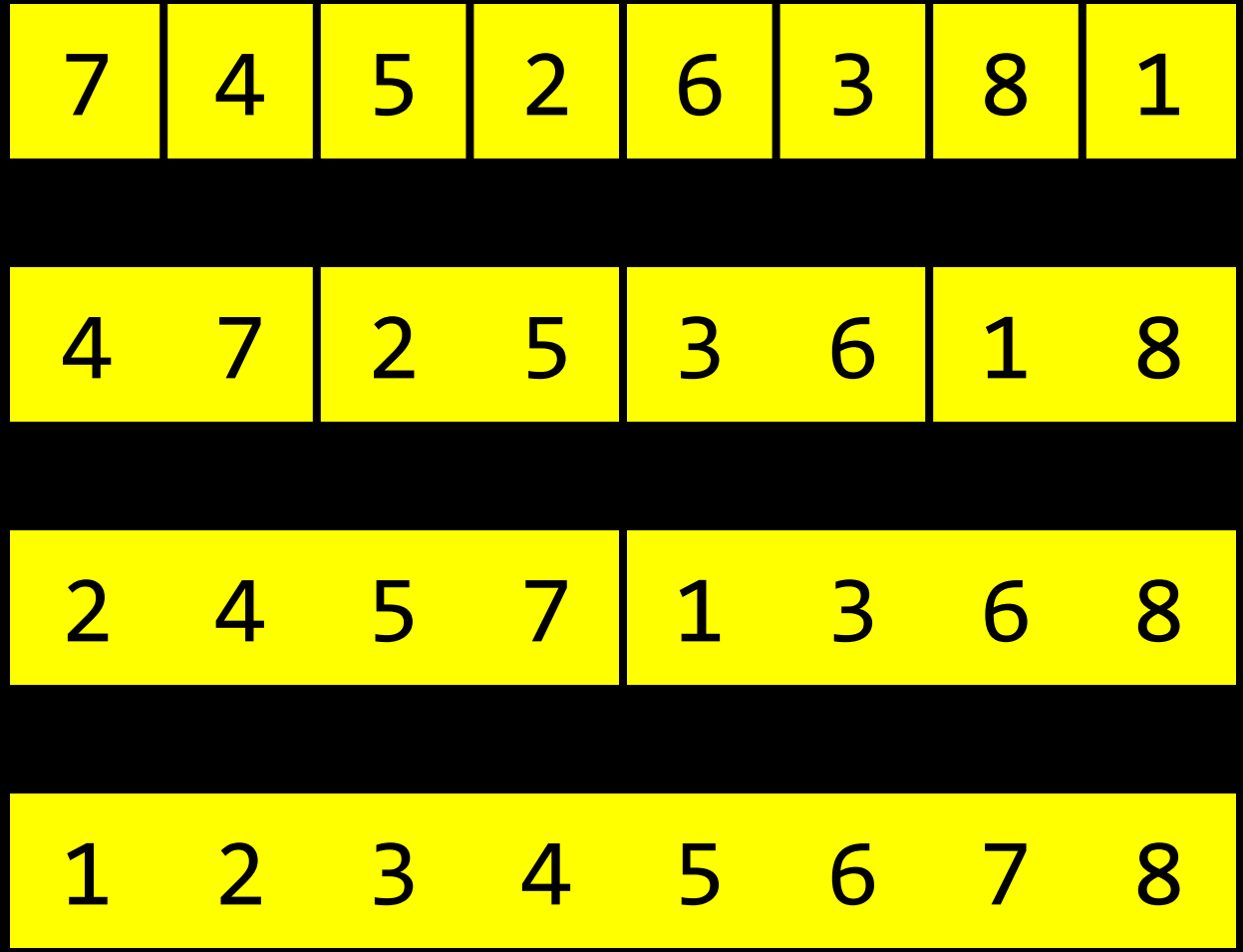

8
1 2 3 4 5 6 7

1 2 3 4 5 6 7 8


7
4
5
2
6
3
8
1


7 4 5 2 6 3 8 1
4 7 2 5 3 6 1 8


7 4 5 2 6 3 8 1
4 7 2 5 3 6 1 8
2 4 5 7 1 3 6 8


7 4 5 2 6 3 8 1
4 7 2 5 3 6 1 8
2 4 5 7 1 3 6 8
1 2 3 4 5 6 7 8

$O(n^2)$

$O(n \log n)$ merge sort

$O(n)$

$O(\log n)$

$O(1)$

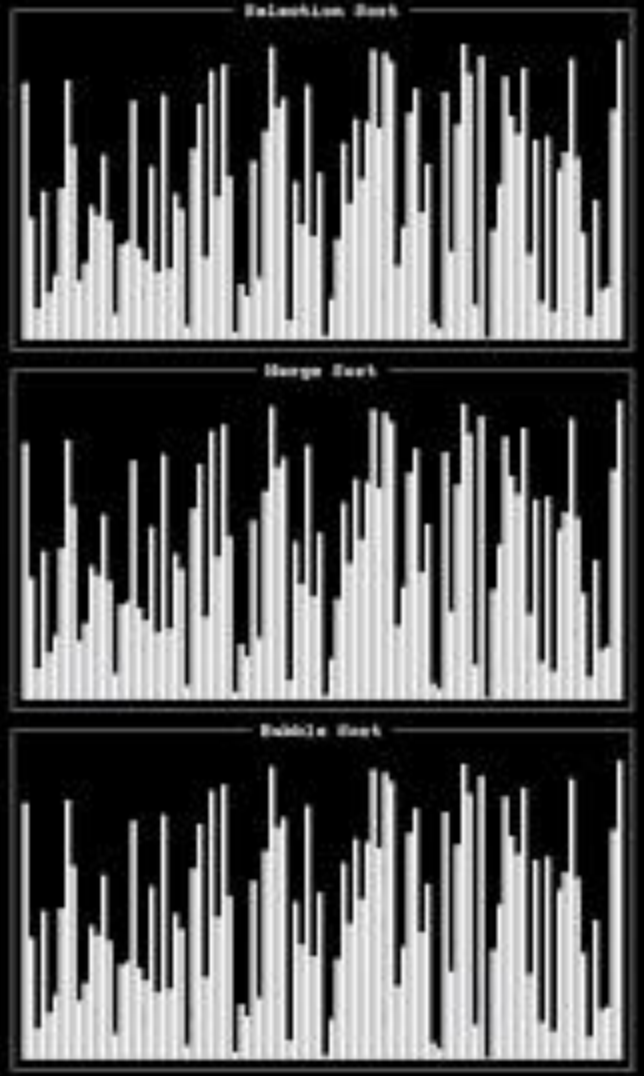
$\Omega(n^2)$

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

$\Omega(n)$

$\Omega(\log n)$

$\Omega(1)$


Selection Sort
Merge Sort
Bubble Sort

Assignment 1

Office Hours

Lab 0

CS50 for MBAs

Algorithms