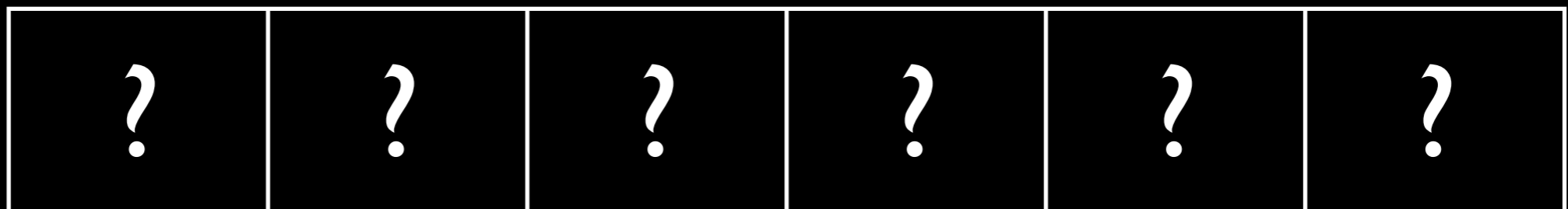


How to Search!

Something useful to do with data

Goal: Find a number!

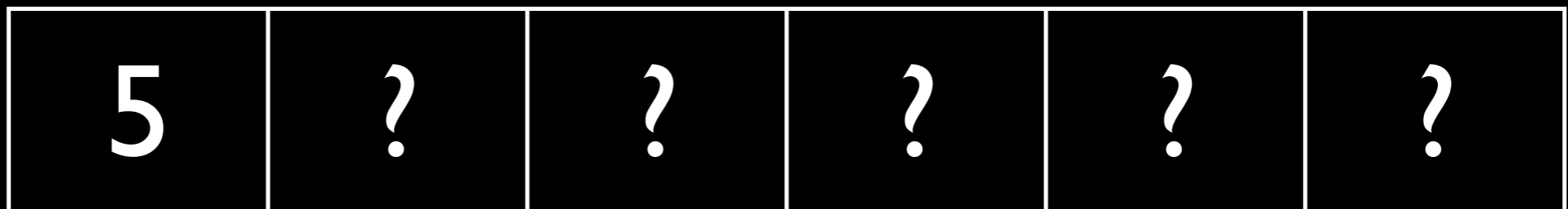
- Remember: Computer can only look at two things at a time



What we want: 42

Goal: Find a number!

- Remember: Computer can only look at two things at a time



What we want: 42

Goal: Find a number!

- Remember: Computer can only look at two things at a time

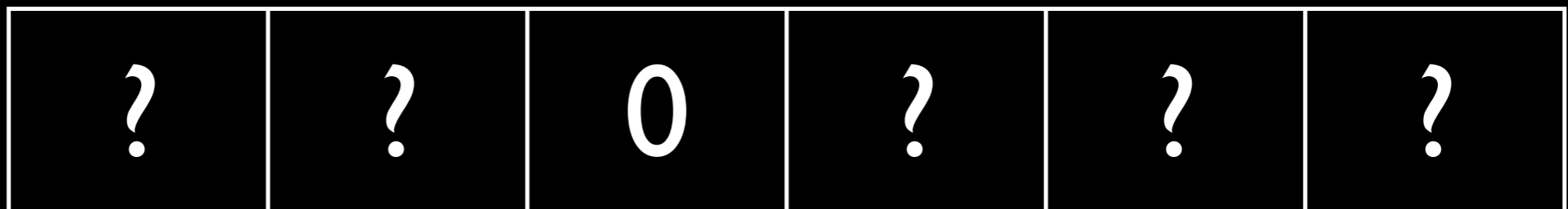
?	-42	?	?	?	?
---	-----	---	---	---	---



What we want: 42

Goal: Find a number!

- Remember: Computer can only look at two things at a time



What we want: 42

Goal: Find a number!

- Remember: Computer can only look at two things at a time

?	?	?	42	?	?
---	---	---	----	---	---



Success!

What we want: 42

Sequential Search

- Search through data items one after another
- Also called Linear Search
- Best search technique when data is out of order

Code v.1

```
int[] array = ...  
int target = ...;
```

```
int index = -1;
```

```
for(int i = 0; i < array.length; i++)  
    if(array[i] == target)  
    {  
        index = i;  
        break;  
    }
```

```
//here index is set correctly
```

Not Found
index == -1

Code v.2

```
public int search(int[] array, int target)
{
    for(int i = 0; i < array.length; i++)
        if(array[i] == target)
            return i;
    return -1;
}
```

Code v.3

```
public boolean contains(int[] array, int target)
{
    for(int i = 0; i < array.length; i++)
        if(array[i] == target)
            return true;
    return false;
}
```

Speed Analysis

4	0	-5	6	92	-4
---	---	----	---	----	----

If right at the beginning:

If right at the end:

If in the middle:

On average:

Speed Analysis

4	0	-5	6	92	-4
---	---	----	---	----	----

If right at the beginning: I check

If right at the end:

If in the middle:

On average:

Speed Analysis

4	0	-5	6	92	-4
---	---	----	---	----	----

If right at the beginning: I check

If right at the end: `array.length` checks

If in the middle:

On average:

Speed Analysis

4	0	-5	6	92	-4
---	---	----	---	----	----

If right at the beginning: 1 check

If right at the end: `array.length` checks

If in the middle: `array.length/2` checks

On average:

Speed Analysis

4	0	-5	6	92	-4
---	---	----	---	----	----

- If right at the beginning: 1 check
- If right at the end: `array.length` checks
- If in the middle: `array.length/2` checks
- On average: You have to check half the values in the array

Searching with Objects

```
public class Time
{
    private int hours;
    private int minutes;

    public int getHours() {
        return hours;
    }

    public int getMinute() {
        return minutes;
    }
}
```

Searching with Objects

Searching with Objects

```
public static void main(String[] args)
```

Searching with Objects

```
public static void main(String[] args)
{
```

Searching with Objects

```
public static void main(String[] args)
{
    //find the latest time
```

Searching with Objects

```
public static void main(String[] args)
{
    //find the latest time
    Time[] array = ...;
```

Searching with Objects

```
public static void main(String[] args)
{
    //find the latest time
    Time[] array = ...;

    Time latest = array[0];
}
```

Searching with Objects

```
public static void main(String[] args)
{
    //find the latest time
    Time[] array = ...;

    Time latest = array[0];
    for(int i = 1; i < array.length; i++)
```

Searching with Objects

```
public static void main(String[] args)
{
    //find the latest time
    Time[] array = ...;

    Time latest = array[0];
    for(int i = 1; i < array.length; i++)
    {
```

Searching with Objects

```
public static void main(String[] args)
{
    //find the latest time
    Time[] array = ...;

    Time latest = array[0];
    for(int i = 1; i < array.length; i++)
    {
        Time current = array[i];
```

Searching with Objects

```
public static void main(String[] args)
{
    //find the latest time
    Time[] array = ...;

    Time latest = array[0];
    for(int i = 1; i < array.length; i++)
    {
        Time current = array[i];
        if(current.getHours() > latest.getHours())
```

Searching with Objects

```
public static void main(String[] args)
{
    //find the latest time
    Time[] array = ...;

    Time latest = array[0];
    for(int i = 1; i < array.length; i++)
    {
        Time current = array[i];
        if(current.getHours() > latest.getHours())
            latest = current;
    }
}
```

Searching with Objects

```
public static void main(String[] args)
{
    //find the latest time
    Time[] array = ...;

    Time latest = array[0];
    for(int i = 1; i < array.length; i++)
    {
        Time current = array[i];
        if(current.getHours() > latest.getHours())
            latest = current;
        else if(current.getHours() == latest.getHours())
    }
```

Searching with Objects

```
public static void main(String[] args)
{
    //find the latest time
    Time[] array = ...;

    Time latest = array[0];
    for(int i = 1; i < array.length; i++)
    {
        Time current = array[i];
        if(current.getHours() > latest.getHours())
            latest = current;
        else if(current.getHours() == latest.getHours())
        {
```

Searching with Objects

```
public static void main(String[] args)
{
    //find the latest time
    Time[] array = ...;

    Time latest = array[0];
    for(int i = 1; i < array.length; i++)
    {
        Time current = array[i];
        if(current.getHours() > latest.getHours())
            latest = current;
        else if(current.getHours() == latest.getHours())
        {
            if(current.getMinutes() > latest.getMinutes())

```

Searching with Objects

```
public static void main(String[] args)
{
    //find the latest time
    Time[] array = ...;

    Time latest = array[0];
    for(int i = 1; i < array.length; i++)
    {
        Time current = array[i];
        if(current.getHours() > latest.getHours())
            latest = current;
        else if(current.getHours() == latest.getHours())
        {
            if(current.getMinutes() > latest.getMinutes())
                latest = current;
        }
    }
}
```

Searching with Objects

```
public static void main(String[] args)
{
    //find the latest time
    Time[] array = ...;

    Time latest = array[0];
    for(int i = 1; i < array.length; i++)
    {
        Time current = array[i];
        if(current.getHours() > latest.getHours())
            latest = current;
        else if(current.getHours() == latest.getHours())
        {
            if(current.getMinutes() > latest.getMinutes())
                latest = current;
        }
    }
}
```

Searching with Objects

```
public static void main(String[] args)
{
    //find the latest time
    Time[] array = ...;

    Time latest = array[0];
    for(int i = 1; i < array.length; i++)
    {
        Time current = array[i];
        if(current.getHours() > latest.getHours())
            latest = current;
        else if(current.getHours() == latest.getHours())
        {
            if(current.getMinutes() > latest.getMinutes())
                latest = current;
        }
    }
}
```

Searching with Objects

```
public static void main(String[] args)
{
    //find the latest time
    Time[] array = ...;

    Time latest = array[0];
    for(int i = 1; i < array.length; i++)
    {
        Time current = array[i];
        if(current.getHours() > latest.getHours())
            latest = current;
        else if(current.getHours() == latest.getHours())
        {
            if(current.getMinutes() > latest.getMinutes())
                latest = current;
        }
    }
    //latest will be assigned the right time
}
```

Searching with Objects

```
public static void main(String[] args)
{
    //find the latest time
    Time[] array = ...;

    Time latest = array[0];
    for(int i = 1; i < array.length; i++)
    {
        Time current = array[i];
        if(current.getHours() > latest.getHours())
            latest = current;
        else if(current.getHours() == latest.getHours())
        {
            if(current.getMinutes() > latest.getMinutes())
                latest = current;
        }
    }
    //latest will be assigned the right time
}
```

Faster Search

- There is a faster way to search if one condition is met: the data is **sorted**
- Dictionary Example
 - How to search a dictionary?
 - How many page-checks to find a word?

Binary Search

- Idea
 - Take sorted data
 - Repeatedly split into smaller groups
 - Divide and Conquer strategy!
 - Much faster than Linear Search

Binary Search #1

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

Binary Search #1

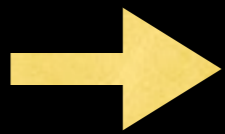
-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

target = 21

Binary Search #1

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 0

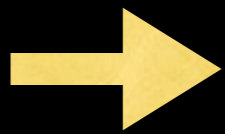


target = 21

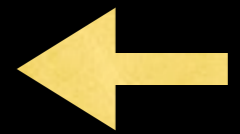
Binary Search #1

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 0



hi: 8

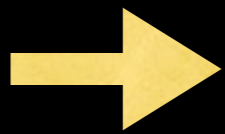


target = 21

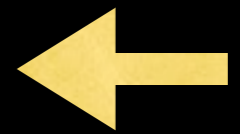
Binary Search #1

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 0



hi: 8



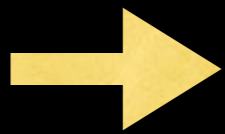
target = 21

mid = (lo + hi)/2

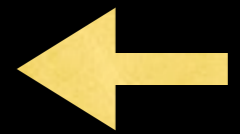
Binary Search #1

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 0



hi: 8



mid = 4

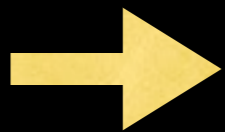
target = 21

$mid = (lo + hi) / 2$

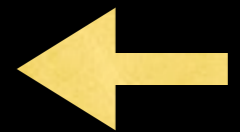
Binary Search #1

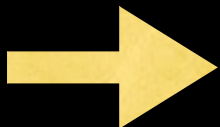
-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 0



hi: 8



mid = 4 



target = 21

$mid = (lo + hi) / 2$

$lo = mid + 1$

Binary Search #1

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

target = 21

mid = (lo + hi)/2

Binary Search #1

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 5


target = 21

mid = (lo + hi)/2

Binary Search #1

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 5
→

hi: 8
←

target = 21

mid = (lo + hi)/2

Binary Search #1

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 5
→

hi: 8
←

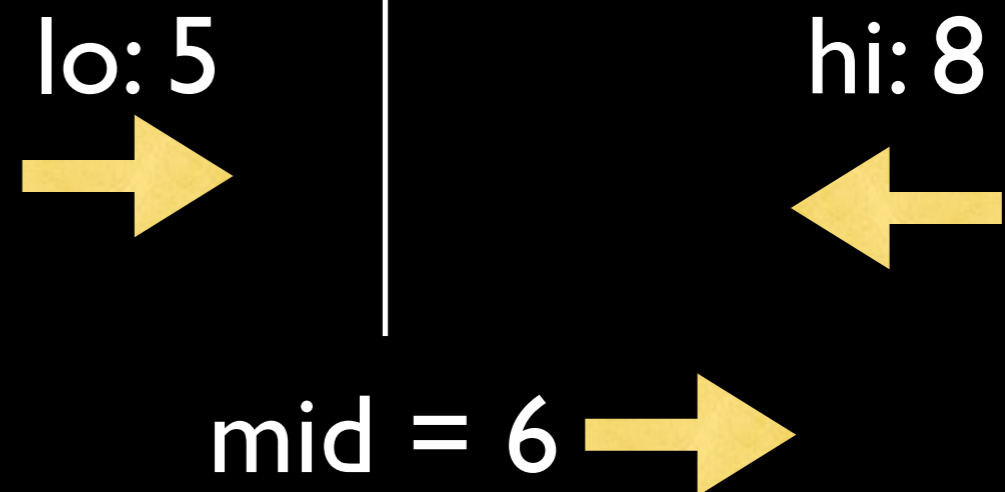
↑
mid = 6

target = 21

$mid = (lo + hi) / 2$

Binary Search #1

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----



target = 21

$mid = (lo + hi) / 2$

$lo = mid + 1$

Binary Search #1

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

target = 21

mid = (lo + hi)/2

Binary Search #1

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 7


target = 21

mid = (lo + hi)/2

Binary Search #1

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 7 hi: 8
→ ←

target = 21

mid = (lo + hi)/2

Binary Search #1

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 7 hi: 8


mid = 7

target = 21

mid = (lo + hi)/2

Binary Search #1

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 7 hi: 8


mid = 7

target = 21

mid = (lo + hi)/2

Success!

Binary Search #2

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

Binary Search #2

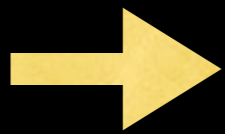
-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

target = 3

Binary Search #2

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 0

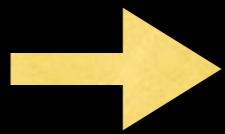


target = 3

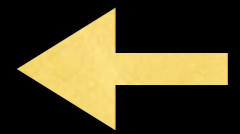
Binary Search #2

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 0



hi: 8

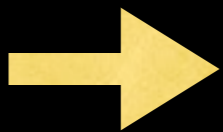


target = 3

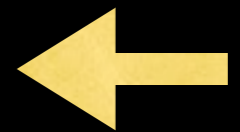
Binary Search #2

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 0



hi: 8



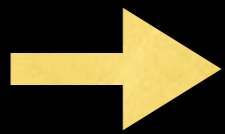
target = 3

mid = (lo + hi)/2

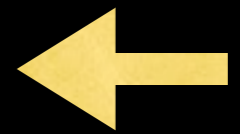
Binary Search #2

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 0



hi: 8



mid = 4

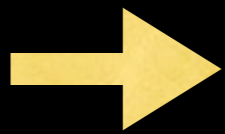
target = 3

$mid = (lo + hi) / 2$

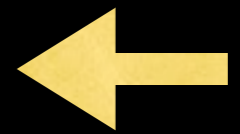
Binary Search #2

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 0



hi: 8



← mid = 4



target = 3

$mid = (lo + hi) / 2$

$hi = mid - 1$

Binary Search #2

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

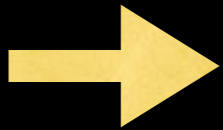
target = 3

mid = (lo + hi)/2

Binary Search #2

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 0



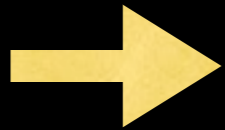
target = 3

mid = (lo + hi)/2

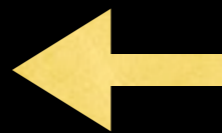
Binary Search #2

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 0



hi: 3



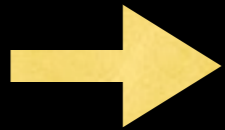
target = 3

$mid = (lo + hi) / 2$

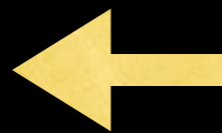
Binary Search #2

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 0



hi: 3



mid = 1

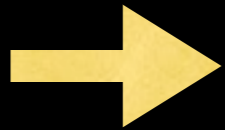
target = 3

$mid = (lo + hi) / 2$

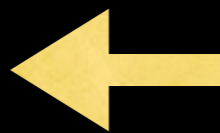
Binary Search #2

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

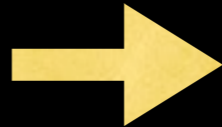
lo: 0



hi: 3



mid = 1



target = 3

$mid = (lo + hi) / 2$

$lo = mid + 1$

Binary Search #2

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

target = 3

mid = (lo + hi)/2

Binary Search #2

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

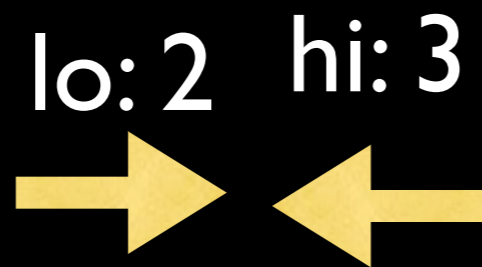
lo: 2


target = 3

$mid = (lo + hi) / 2$

Binary Search #2

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

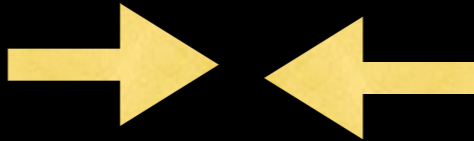


target = 3

$mid = (lo + hi) / 2$

Binary Search #2

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 2 hi: 3


mid = 2

target = 3

mid = (lo + hi)/2

Binary Search #2

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 2 hi: 3
→ ←

mid = 2 →

target = 3

$mid = (lo + hi) / 2$

$lo = mid + 1$

Binary Search #2

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

target = 3

mid = (lo + hi)/2

Binary Search #2

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 3

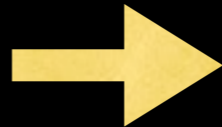

target = 3

mid = (lo + hi)/2

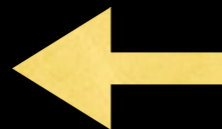
Binary Search #2

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 3



hi: 3



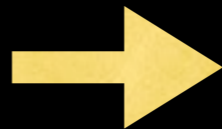
target = 3

$mid = (lo + hi) / 2$

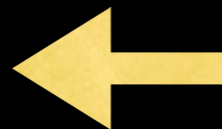
Binary Search #2

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 3



hi: 3



mid = 3

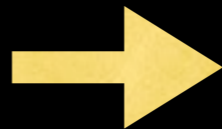
target = 3

$mid = (lo + hi) / 2$

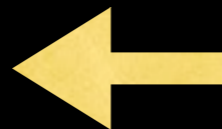
Binary Search #2

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 3



hi: 3



mid = 3

target = 3

Success!

$mid = (lo + hi) / 2$

Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

Binary Search #3

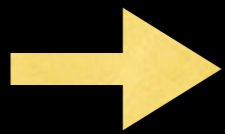
-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

target = 44

Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 0

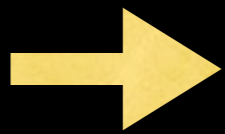


target = 44

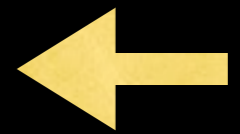
Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 0



hi: 8

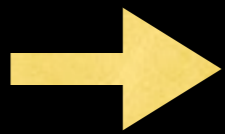


target = 44

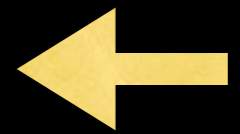
Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 0



hi: 8



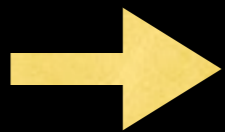
target = 44

mid = (lo + hi)/2

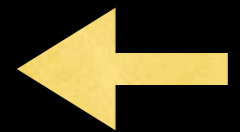
Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 0



hi: 8



mid = 4

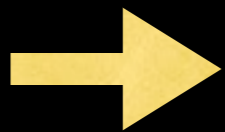
target = 44

$mid = (lo + hi) / 2$

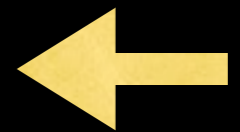
Binary Search #3

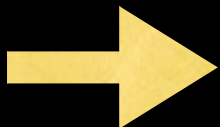
-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 0



hi: 8



mid = 4 



target = 44

$mid = (lo + hi) / 2$

$lo = mid + 1$

Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

target = 44

mid = (lo + hi)/2

Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 5


target = 44

mid = (lo + hi)/2

Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 5
→

hi: 8
←

target = 44

mid = (lo + hi)/2

Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 5
→

hi: 8
←

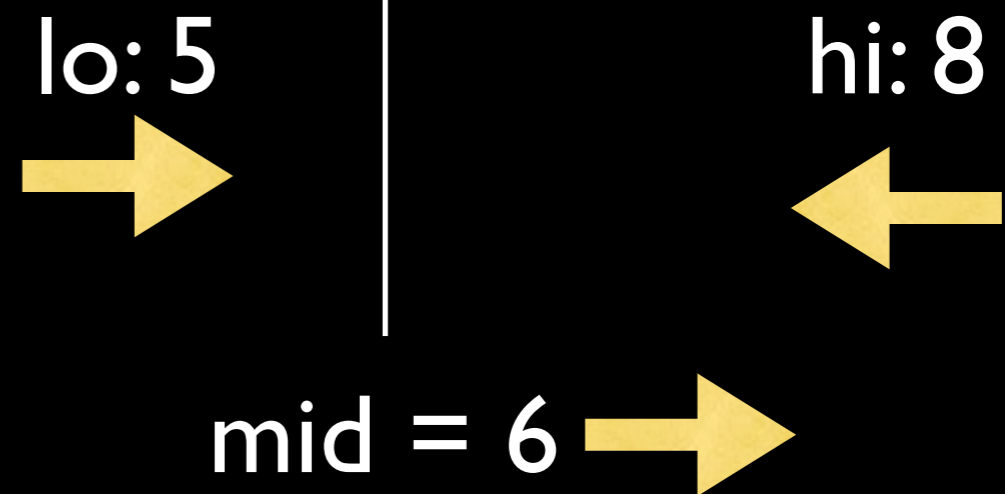
↑
mid = 6

target = 44

$mid = (lo + hi) / 2$

Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----



target = 44

$mid = (lo + hi) / 2$

$lo = mid + 1$

Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

target = 44

mid = (lo + hi)/2

Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 7


target = 44

mid = (lo + hi)/2

Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 7 hi: 8
→ ←

target = 44

mid = (lo + hi)/2

Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 7 hi: 8


mid = 7

target = 44

mid = (lo + hi)/2

Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 7 hi: 8
→ ←

mid = 7 →

target = 44

mid = (lo + hi)/2

lo = mid + 1

Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

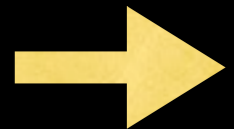
target = 44

mid = (lo + hi)/2

Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 8



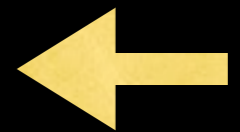
target = 44

mid = (lo + hi)/2

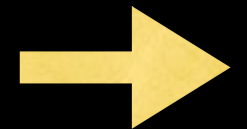
Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

hi: 8



lo: 8



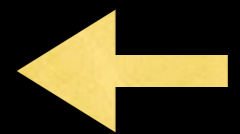
target = 44

mid = (lo + hi)/2

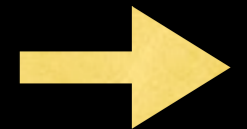
Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

hi: 8



lo: 8



mid = 8

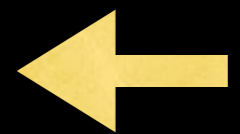
target = 44

$mid = (lo + hi) / 2$

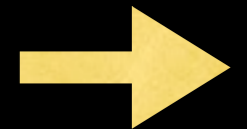
Binary Search #3

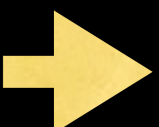
-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

hi: 8



lo: 8



mid = 8 

target = 44

$mid = (lo + hi) / 2$

$lo = mid + 1$

Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

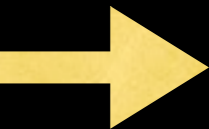
target = 44

mid = (lo + hi)/2

Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

lo: 9



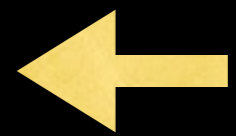
target = 44

mid = (lo + hi)/2

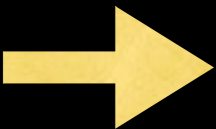
Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

hi: 8



lo: 9



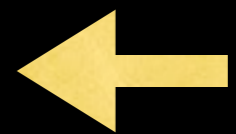
target = 44

$mid = (lo + hi) / 2$

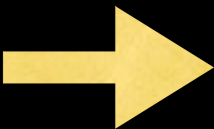
Binary Search #3

-5	-2	0	3	7	10	12	21	43
----	----	---	---	---	----	----	----	----

hi: 8



lo: 9



target = 44

mid = (lo + hi)/2

Failure!

Binary Search v1

```
public int binarySearch(int[] array, int target) {
    int lo = 0;
    int hi = array.length-1;

    while(lo <= hi) {
        int mid = (lo + hi)/2;
        if(array[mid] == target)
            return mid;
        else if(array[mid] < target)
            lo = mid + 1;
        else
            hi = mid - 1;
    }

    return -1;
}
```

Binary Search v2

```
public int binarySearch(int[] array, int target) {  
    return binarySearch(array, target, 0, array.length-1);  
}
```

```
public int binarySearch(int[] array, int target, int lo, int hi) {  
    if(lo > hi)  
        return -1;  
  
    int mid = (lo + hi)/2;  
  
    if(array[mid] == target)  
        return mid;  
    else if(array[mid] < target)  
        return binarySearch(array, target, mid + 1, hi);  
    else  
        return binarySearch(array, target, lo, mid - 1);  
}
```

Speed Analysis

- Binary Search is repeated division by 2
- The question is, how many times do you have to divide by 2 (at worst) to find what you're looking for?
- Repeated division by 2 is the same as \log_2
- $\log_2(16) = 4$

Comparison

$n / 2$

$\log_2(n)$

array.length	Sequential	Binary
4	2	2
1,024	512	10
1,048,576	524,288	20