Data Structure

A method for organizing data in memory.

Array

A list of values indexed with integers.

operations

[], []=, length

```
var array = ["only item"]; // create an array
array[0] = "first element"; // change an item
array[1] = "another item"; // add an item
```

Try Me

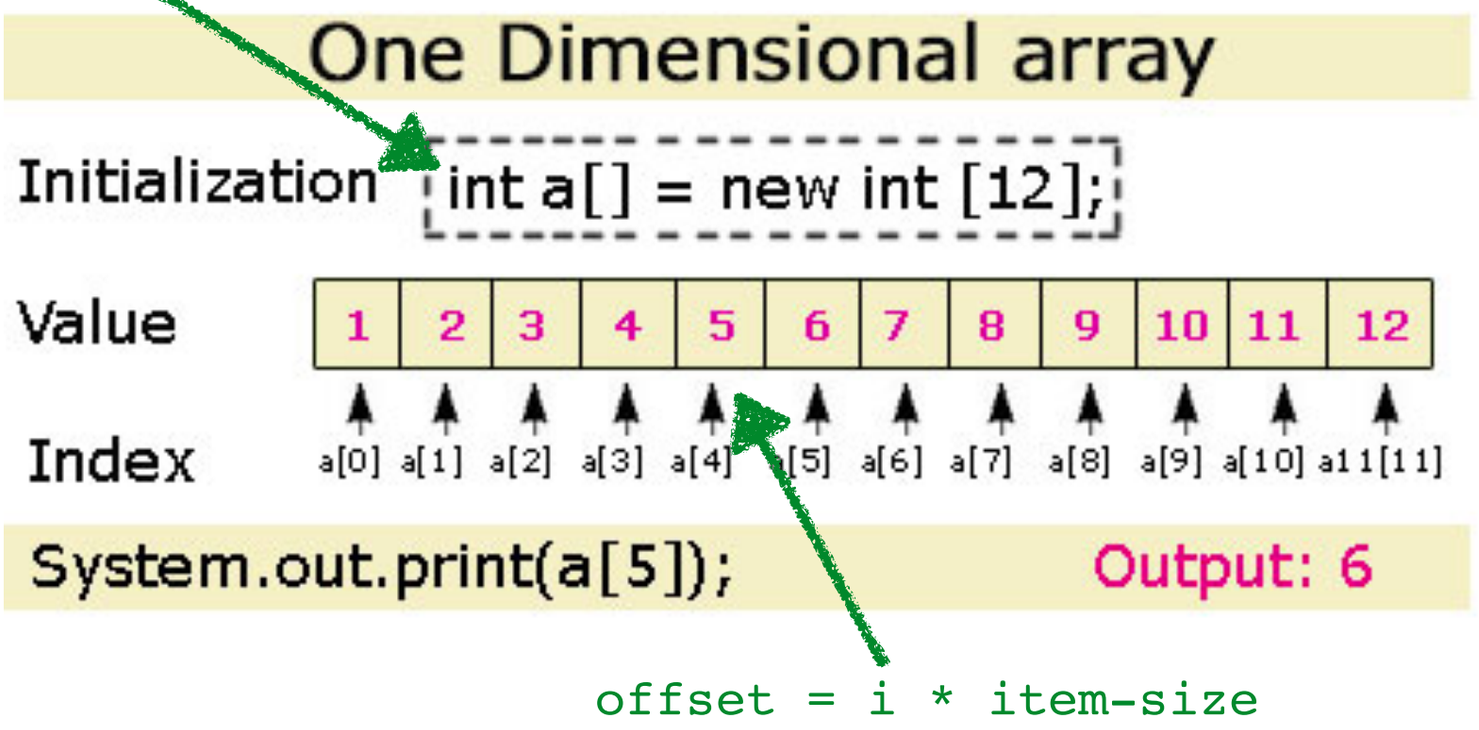
given an array, test it for duplicates
(return false if there are no duplicates, true if there's at least one)

```
function hasDuplicate(array) {
  for (var i = 0; i < array.length; i++) {
    for (var j = i + 1; j < array.length; j++) {
      if (array[i] === array[j]) {
        return true;
      }
    }
  }
  return false;
}
```

In almost every language...

array items are contiguous in memory.

every item is the same type



offset = i * item-size

JS Arrays are Objects

Available methods promote a concept of order:
push(), pop(), shift(), unshift()

Object indices are coerced into strings.

Length is just a property.

Try Me

```
var a = ["apple", "pear", "banana"];
console.log(a[0]);
console.log(a["0"]);

a.notAnInt = "another property";
console.log(a.notAnInt);
```

```
[ "apple", "pear", "banana" ]

is actually

{
  "0" : "apple",
  "1" : "pear",
  "2" : "banana"
}

(plus extra methods)
```

Array Methods

push, pop, shift, unshift

push()

Adds the given arguments to the end of the array.

pop()

Remove and return the last element of the array.

shift()

Remove and return the first element of the array.

Try Me

```
> kitties =
[ 'jude paw',
  'cat winslet',
  'chairman meow' ]

> kitties.shift();
'jude paw'

> kitties
[ 'cat winslet', 'chairman meow' ]
```

unshift()

Adds the given arguments to the beginning of the array.

Try Me

```
> var kitties = [];
undefined

> kitties.unshift("chairman meow");
1
```

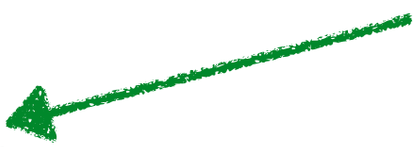
WHERE DID THIS VALUE COME FROM?

```
> kitties
[ 'chairman meow' ]

> kitties.unshift("jude paw", "cat winslet");
3

> kitties
[ 'jude paw',
  'cat winslet',
  'chairman meow' ]
```

THE LAST ARGUMENT IS UNSHIFTED FIRST.

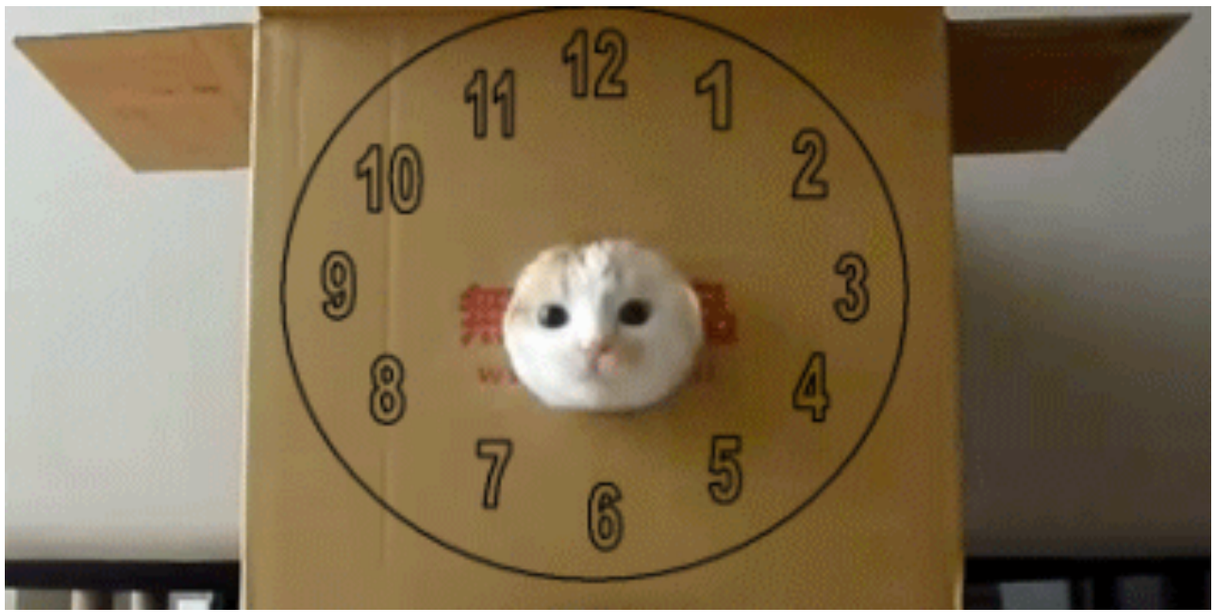


Algorithmic Analysis

determining the resources required for a piece of code to run

Time Complexity

execution time as a function of the input size


12
1
2
3
4
5
6
7
8
9
10
11

Space Complexity

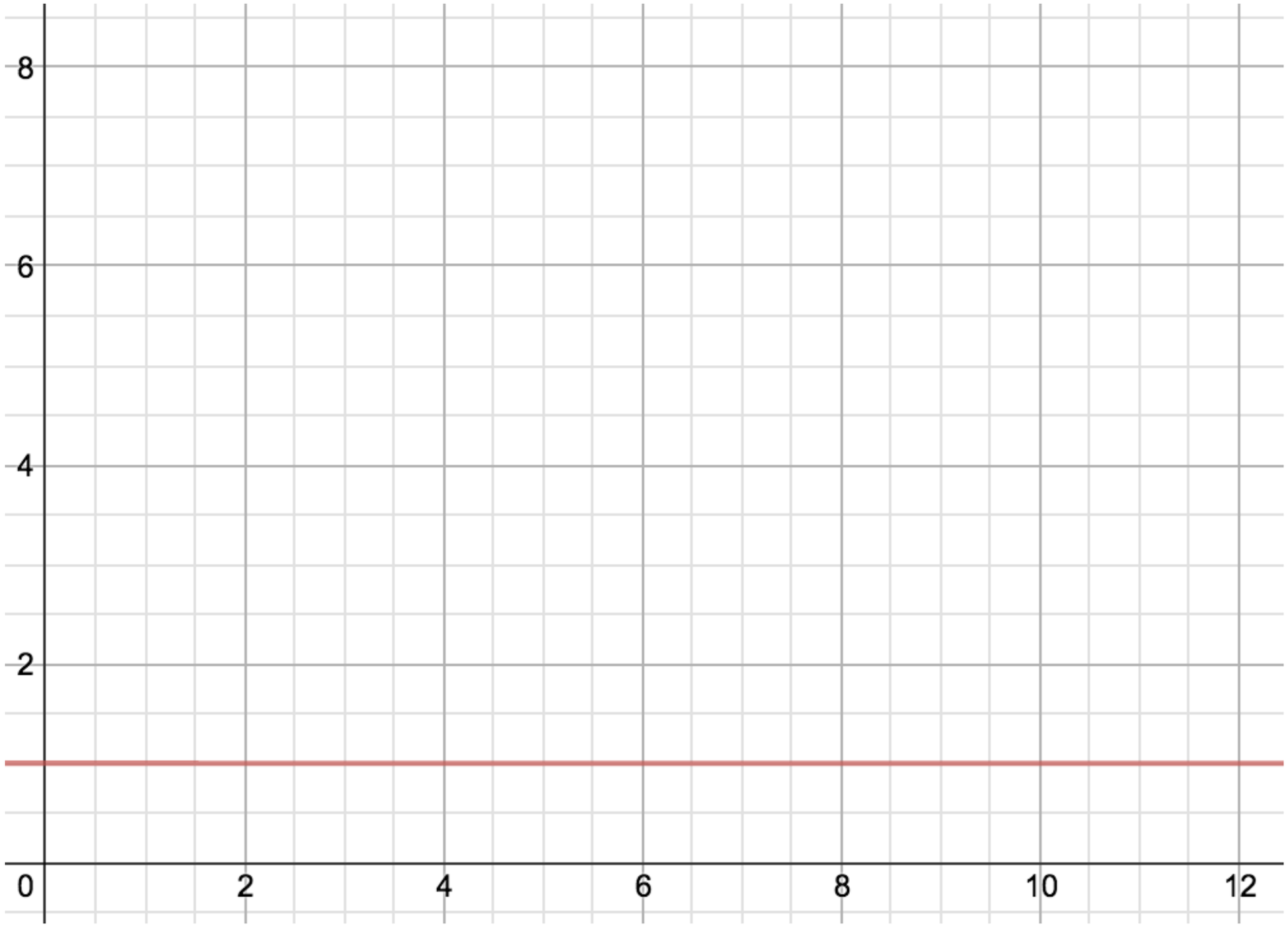
memory/disk space used, as a function of the input size

Big O notation

the vocabulary we use to discuss complexity.

O(1) : Constant

Input size is irrelevant.



https://www.desmos.com/calculator

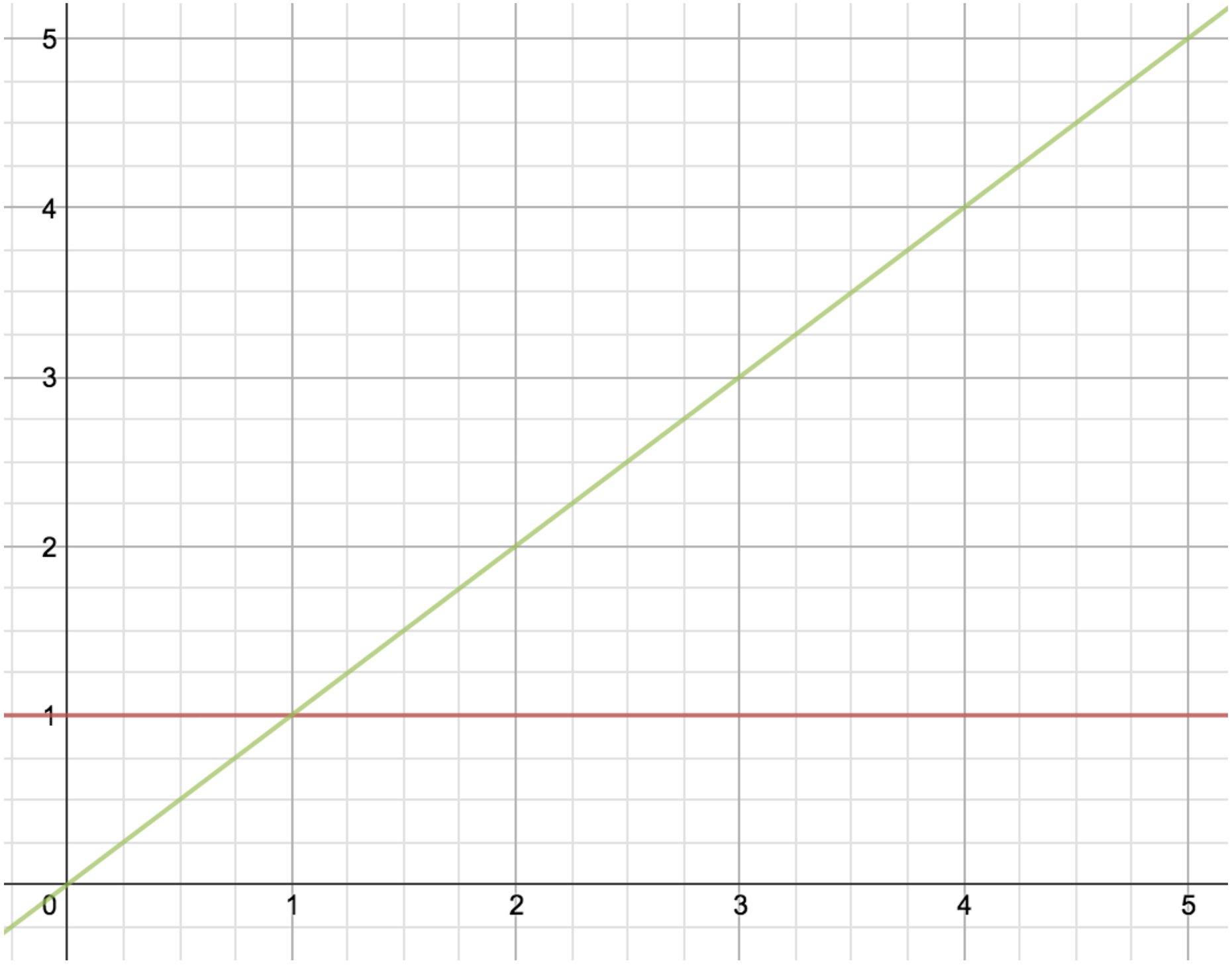
Try Me

implement push

```
function push(array, item) {
  array[array.length] = item;
}
```

O(n) : Linear

Complexity grows proportionately to the input size.



https://www.desmos.com/calculator

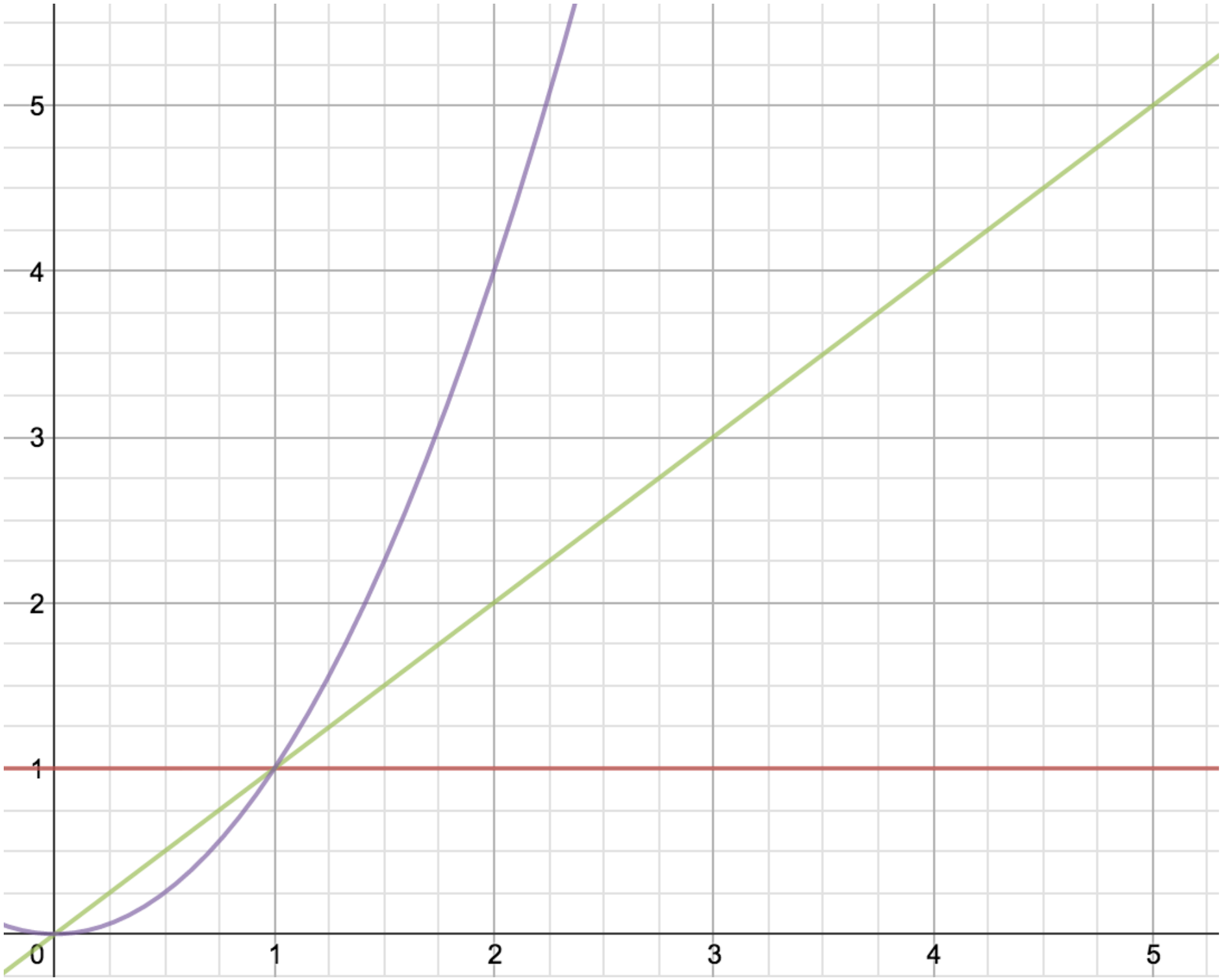
Try Me

implement unshift

```
function unshift(array, item) {
  for(var i = array.length - 1; i > 0; i--) {
    array[i+1] = array[i];
  }
  array[0] = item;
}
```

O(n^2) : Quadratic

Complexity grows proportionately to the input size multiplied by itself.



https://www.desmos.com/calculator

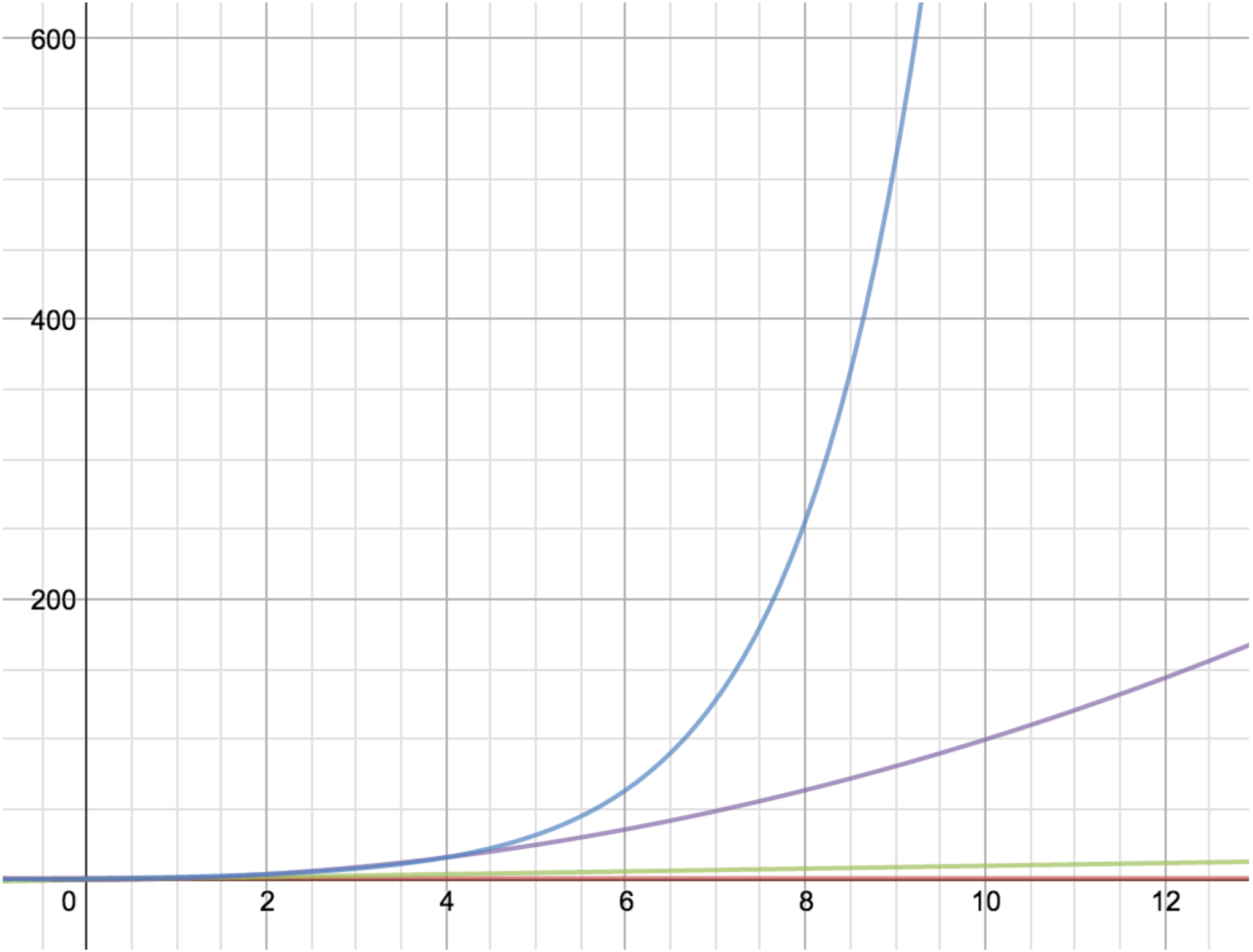
Explain

Why is the below O(n^2)?

```
function hasDuplicate(array) {
  for (var i = 0; i < array.length; i++) {
    for (var j = i + 1; j < array.length; j++) {
      if (array[i] === array[j]) {
        return true;
      }
    }
  }
  return false;
}
```

O(2^n) : Exponential

Shit gets real, real quick.



https://www.desmos.com/calculator

WESTERN CIVILIZATIONS
VOLUME 2
COFFIN STACEY
LERNER MEACHAM
NORTON
BAILEY
MAGIC AND SUPERSTITION IN EUROPE
Nature's Body
Holy Bible
NEW INTERNATIONAL VERSION
INTERNATIONAL BIBLE SOCIETY
DON'T SWEAT THE SMALL STUFF For Teens
SIGMUND FREUD
Civilization and Its Discontents
You Mean I'm Not Lazy, Stupid Or Crazy?!
Kate Kelly and Peggy Ramundo
The Truth ABOUT The Truth
Anderson
DISRUPTIVE RELIGION
SMITH
A BRIEF INTRODUCTION TO THE PHILOSOPHY OF MIND
CAMPBELL
The NIV STUDY BIBLE
Zondervan
Third Edition
HOLT RINEHART WINSTON

Dictionary

set of key value pairs

operations

initialize {}, add []=, get [], delete

a.k.a

hash, associative arrays, map,
symbol table

Dictionaries in JS

what does that sound like?

hasOwnProperty()

return true if the object has the given property (being in the prototype chain doesn't count).

Try Me

write a function that determines if an object inherits a given property.

```
function willInherit(obj, property) {
  return !!(obj[property] && !obj.hasOwnProperty(property));
}
```

Try Me

now, use a dictionary to help you test for duplicates

```
function hasDuplicate(array) {
  var seen = {};
  var current;
  for (var i = 0; i < array.length; i++) {
    current = array[i];
    if (seen[current]) {
      return true;
    }
    seen[current] = true;
  }
  return false;
}
```

white-boarding

Coding on a whiteboard --
part of most technical interviews.

paraphrase

Restate the problem in your own words.

ask questions

do you have to make any assumptions?
don't!
ask instead.

Try Me

how would you paraphrase "check for duplicates in an array"?

what questions would you ask?

draw out examples

sample input with expected output.

Try Me

now draw out examples for the
"check for duplicates in an array."

trace your code

walk through the code, line by line, noting (on the white-board) how variables change.

discuss complexity

volunteer analysis of your code.

Problems

In pairs, select from the following problems.
Write solutions on paper. One person writes, one person comments. Switch off each time you start a new problem.

Follow these steps:
1. paraphrase
2. ask questions
3. draw examples
4. code it
5. trace your code (with your examples)
6. discuss complexity

unique

implement a function called unique. it takes a array and returns a copy of the array with all duplicates removed.

frequency

given an array of all english words, figure out what the most common first letter is.

frequency 2

given an array of all english words, figure out what the most common letter (anywhere in a word) is.

shift(array, item)

implement shift for an array, without using any built-in helper methods.

pop(array)

implement pop, without using any array helper methods.