DATA STRUCTURES

WE'VE HIT A

TURNING POINT

IN THE COURSE

Binary Search Trees

Queues

Singly Linked Lists

Undirected Unweighted Graphs

Binary Heaps

Directed Graphs

Hash Tables

Doubly Linked Lists

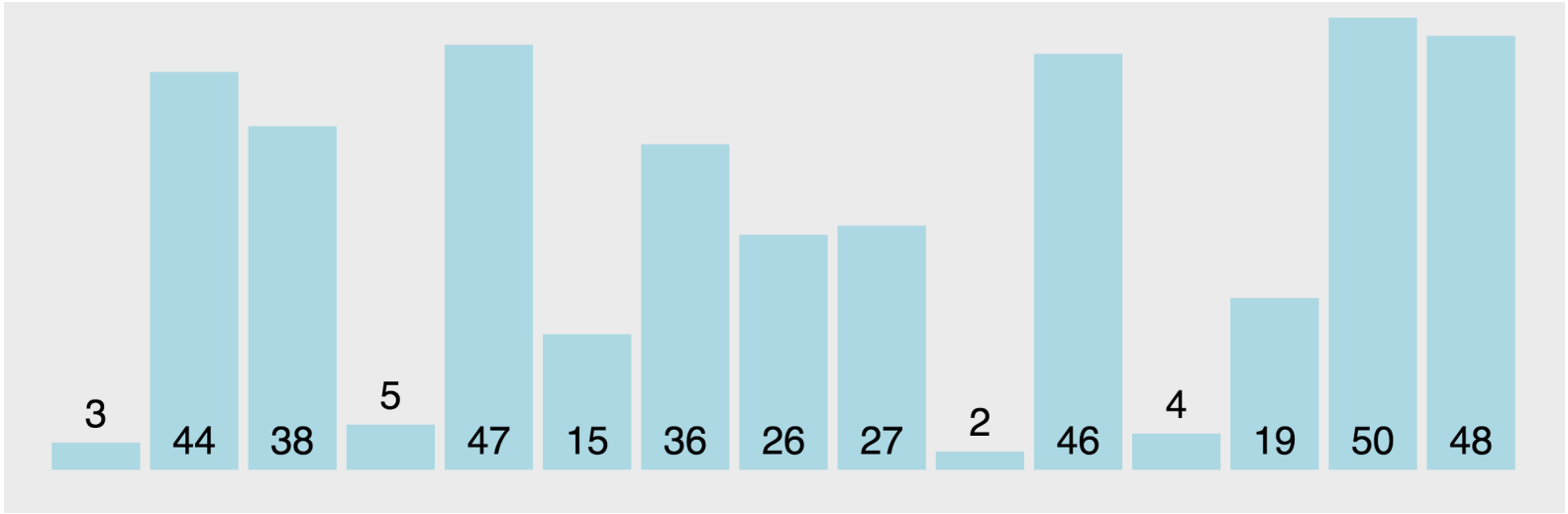
Stacks

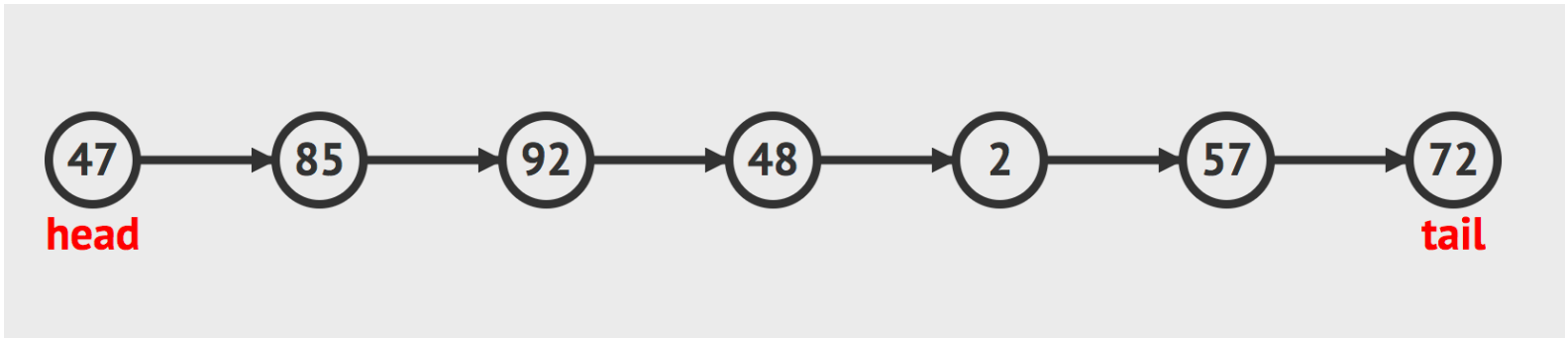
WHAT DO THEY DO?

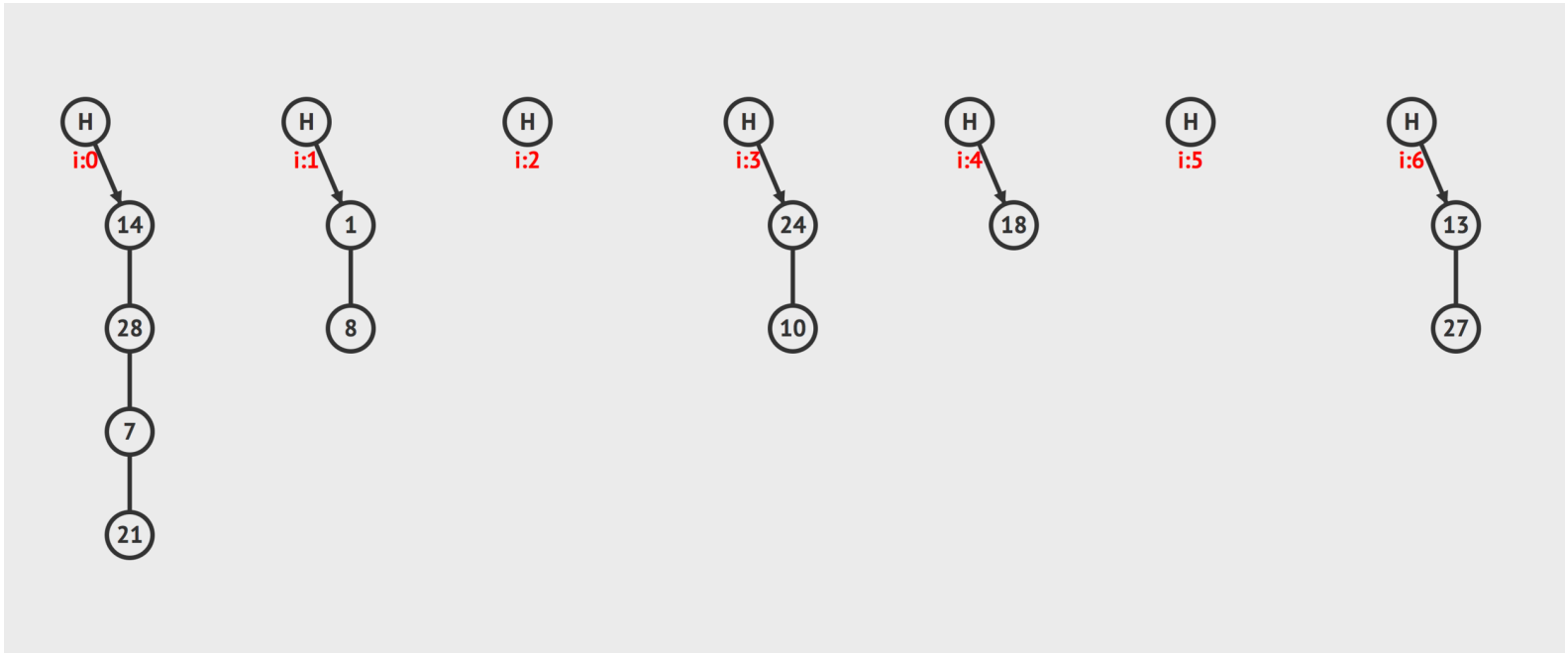
Data structures are collections of values, the relationships among them, and the functions or operations that can be applied to the data

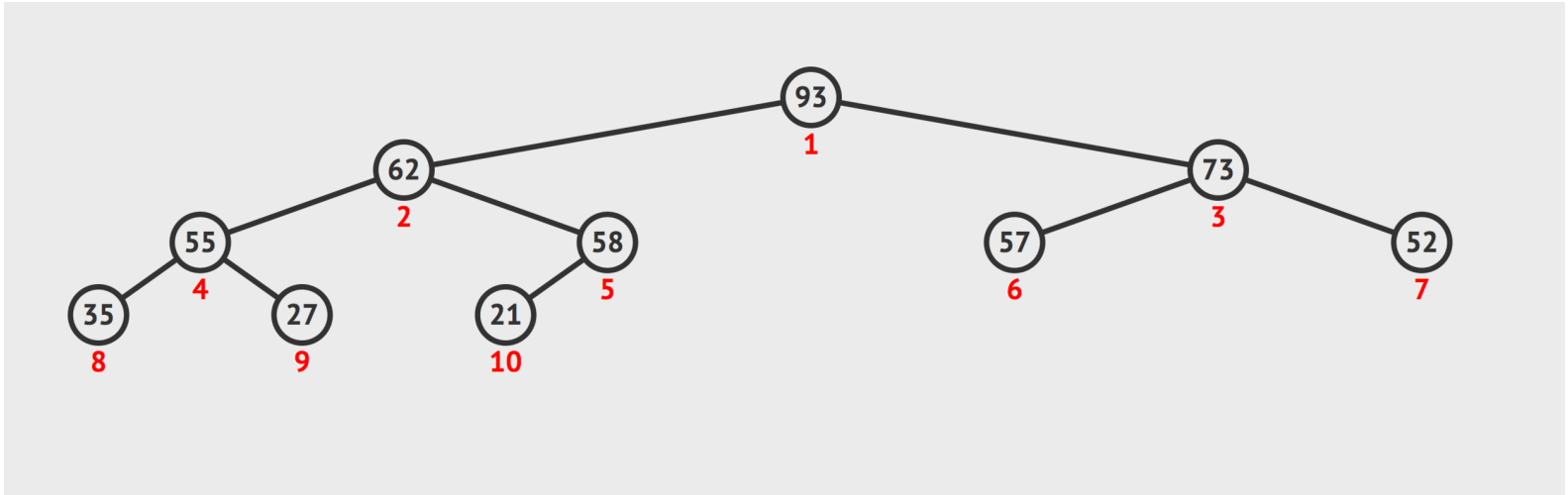
WHY SO MANY???

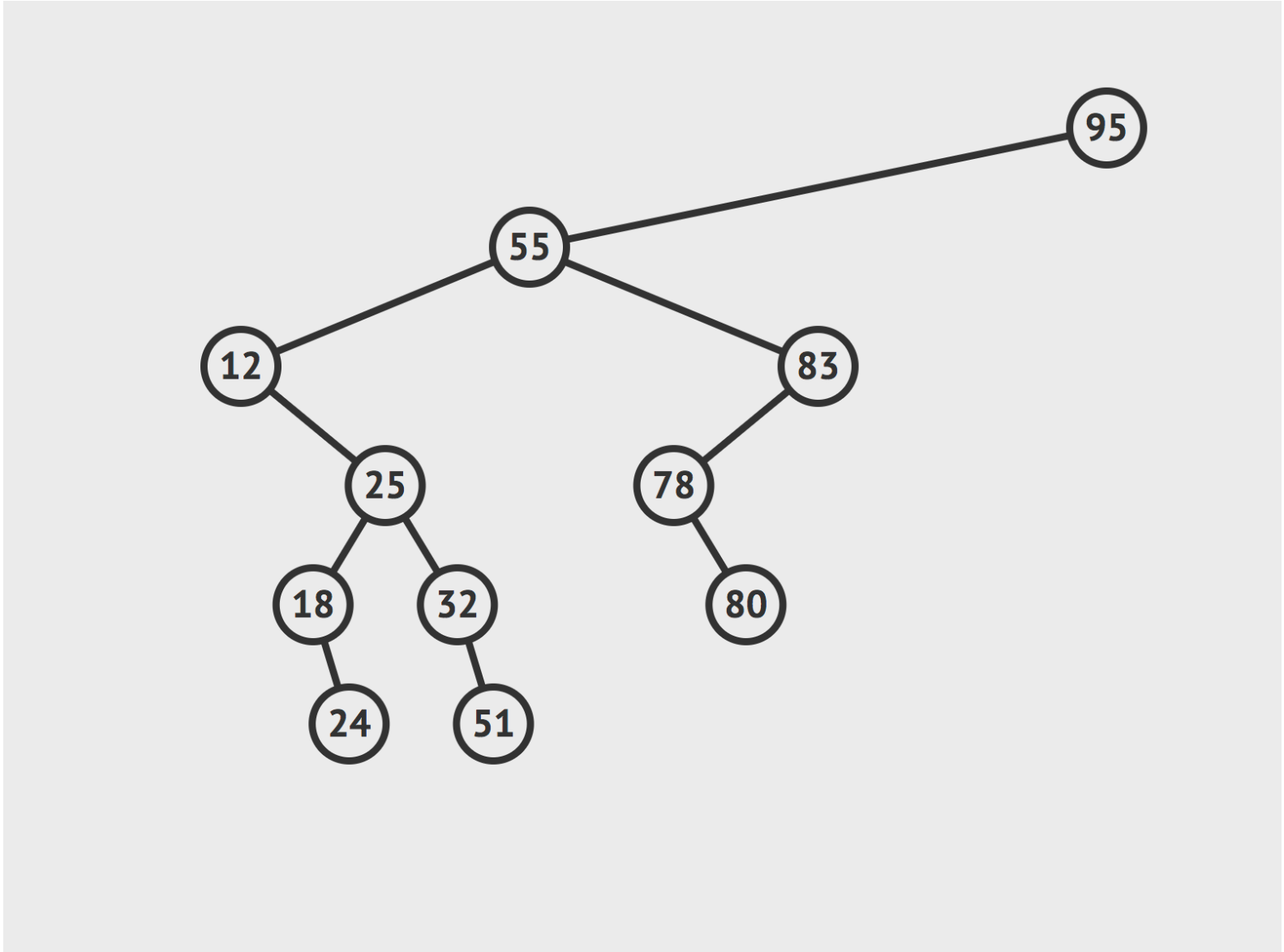
Different data structures excel at different things. Some are highly specialized, while others (like arrays) are more generally used.

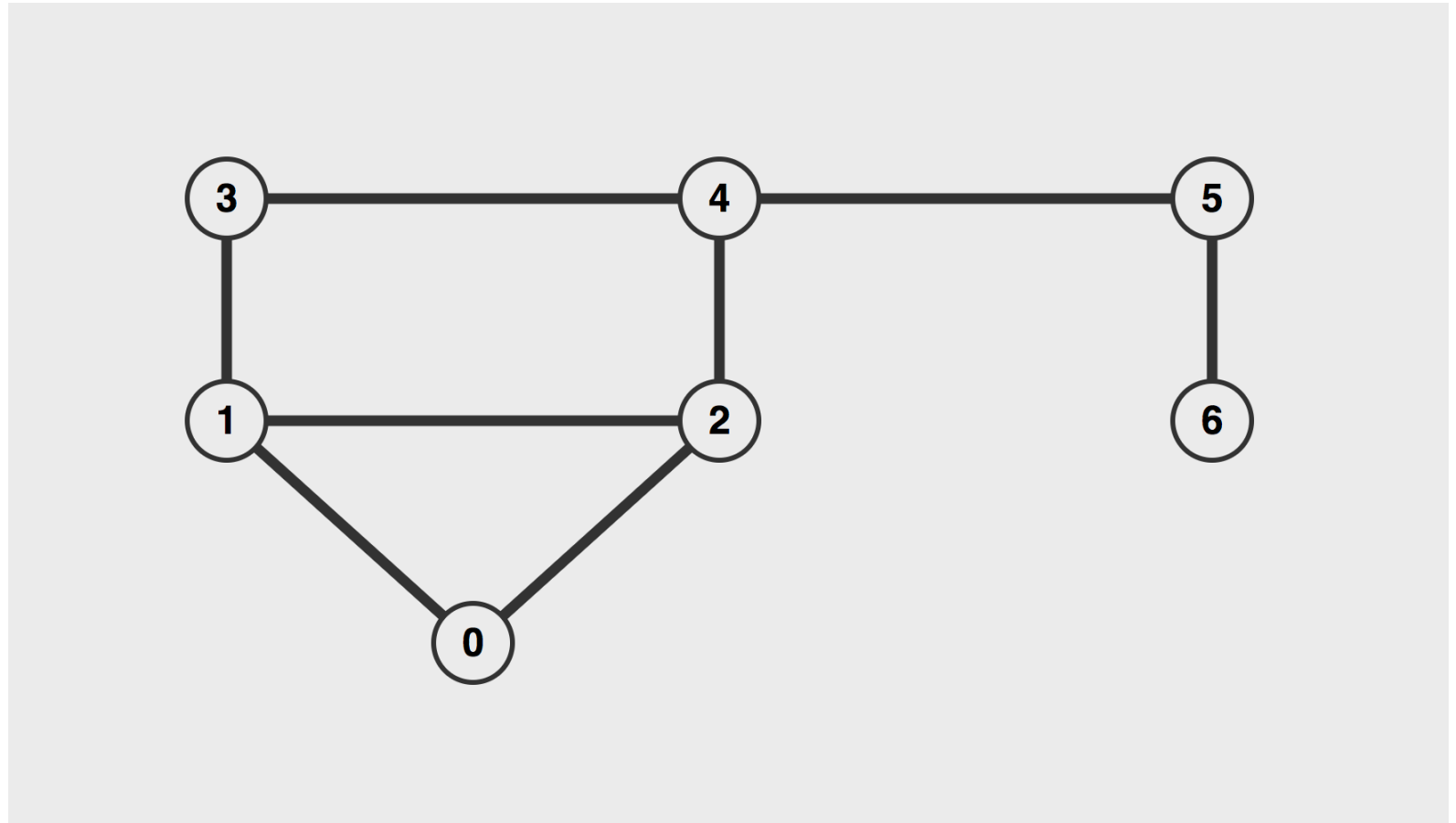

3
44
38
5
47
15
36
26
27
2
46
4
19
50
48

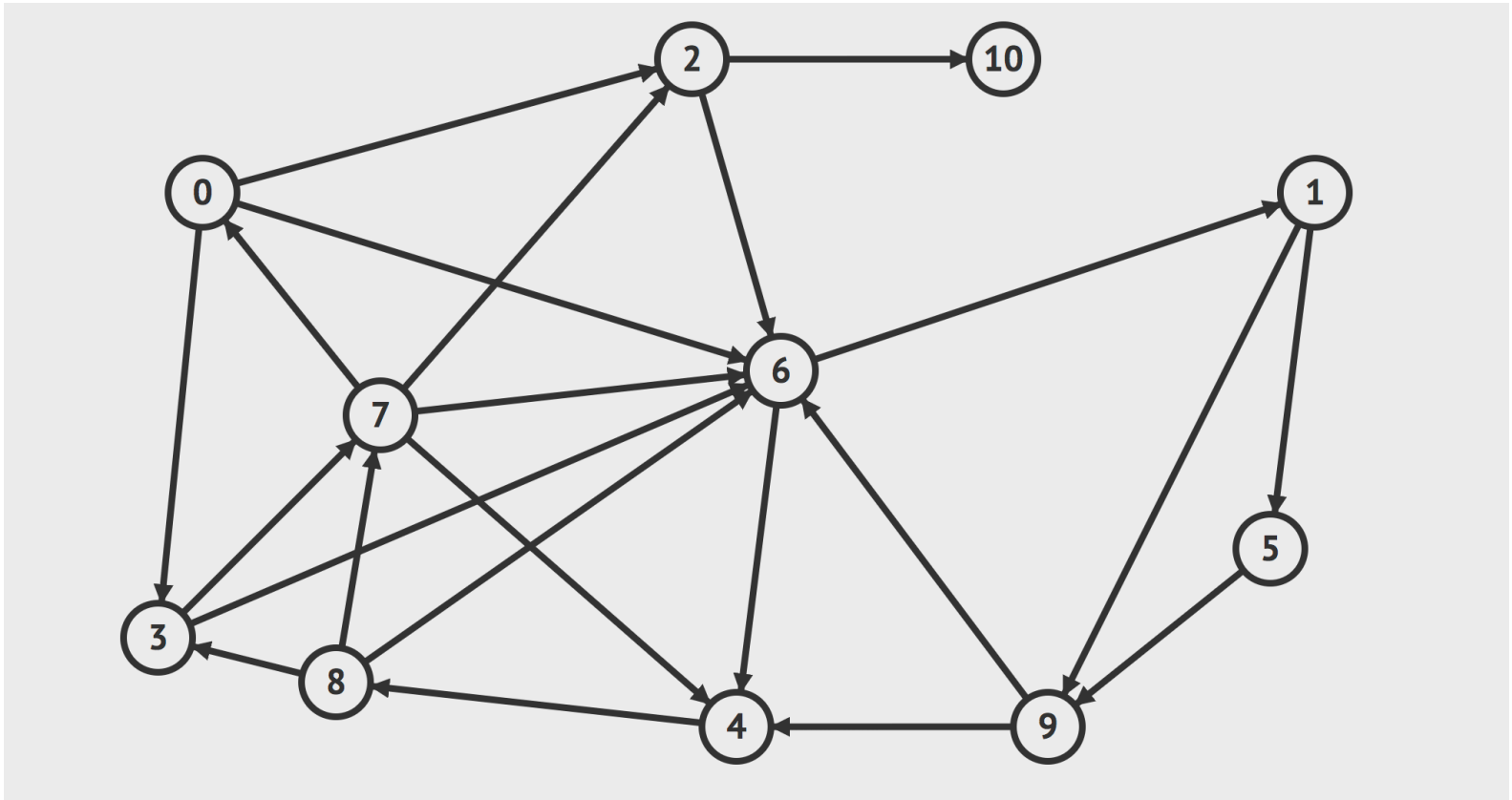

47
85
92
48
2
57
72
head
tail


H
i:0
14
28
7
21
H
i:1
1
8
H
i:2
H
i:3
24
10
H
i:4
18
H
i:5
H
i:6
13
27


93
1
62
2
73
3
55
4
58
5
57
6
52
7
35
8
27
9
21
10


95
55
12
83
25
78
18
32
80
24
51


3
4
5
1
2
6
0


2
10
0
1
6
7
5
3
8
4
9

WHY CARE?

The more time you spend as a developer, the more likely you'll need to use one of these data structures

You've already worked with many of them unknowingly!

And of course...INTERVIEWS

THERE IS NO ONE "BEST" DATA STRUCTURE

They all excel in different situations...

Otherwise why bother learning them all??

Working with map/location data?

Use a graph!

Need an ordered list with fast inserts/removals at the beginning and end?

Use a linked list!

Web scraping nested HTML?

Use a tree!

Need to write a scheduler?

Use a binary heap!

OK, Let's get going!

But first...a quick message

There is a ton of content to take in here...

don't get overwhelmed trying to master it all at once.

I BEG YOU

Learn Singly Linked Lists

TAKE A BREAK!

Learn Doubly Linked Lists

PAY ATTENTION TO THE PREREQUISITES FOR EACH SECTION!

OK, OK I'm done.

Now we just need to learn a bit of ES2015 syntax that we'll use along the way

ES2015
CLASS SYNTAX

OBJECTIVES

- Explain what a class is
- Understand how JavaScript implements the **idea** of classes
- Define terms like abstraction, encapsulation and polymorphism
- Use ES2015 classes to implement data structures

What is a class?

A blueprint for creating objects with pre-defined properties and methods

Does JavaScript really have them?

Ehh....not really

Why do we need to learn this?

We're going to implement **data structures** as **classes!**

THE SYNTAX

```
class Student {
    constructor(firstName, lastName){
        this.firstName = firstName;
        this.lastName = lastName;
    }
}
```

The method to create new objects **must** be called constructor

The class keyword creates a constant, so you can not redefine it. Watch out for the syntax as well!

Creating objects from classes

We use the new keyword

```
class Student {
    constructor(firstName, lastName){
        this.firstName = firstName;
        this.lastName = lastName;
    }
}

let firstStudent = new Student("Colt", "Steele");
let secondStudent = new Student("Blue", "Steele");
```

Instance Methods

```
class Student {
    constructor(firstName, lastName){
        this.firstName = firstName;
        this.lastName = lastName;
    }
    fullName(){
        return `Your full name is ${this.firstName} ${this.lastName}`;
    }
}

let firstStudent = new Student("Colt", "Steele");

firstStudent.fullName() // "Colt Steele"
```

Class Methods

```
class Student {
    constructor(firstName, lastName){
        this.firstName = firstName;
        this.lastName = lastName;
    }

    fullName(){
        return `Your full name is ${this.firstName} ${this.lastName}`;
    }

    static enrollStudents(...students){
        // maybe send an email here
    }
}

let firstStudent = new Student("Colt", "Steele");
let secondStudent = new Student("Blue", "Steele");

Student.enrollStudents([firstStudent, secondStudent])
```

How we'll be using classes

```
class DataStructure(){
    constructor(){
        // what default properties should it have?
    }
    someInstanceMethod(){
        // what should each object created from this class be able to do?
    }
}
```

We will be using the **constructor** and **instance methods** quite a bit!

We will almost **never** be using **static** methods

One gotcha with `this`

Inside all of our **instance** methods and **constructor**, the keyword `this` refers to the object created from that class (also known as an **instance**)

YOUR TURN

Recap

- Classes are blueprints that when created make objects known as **instances**
- Classes are created with the **new** keyword
- The **constructor** function is a special function that gets run when the class is instantiated
- Instance methods can be added to classes similar to methods in objects
- Class methods can be added using the **static** keyword