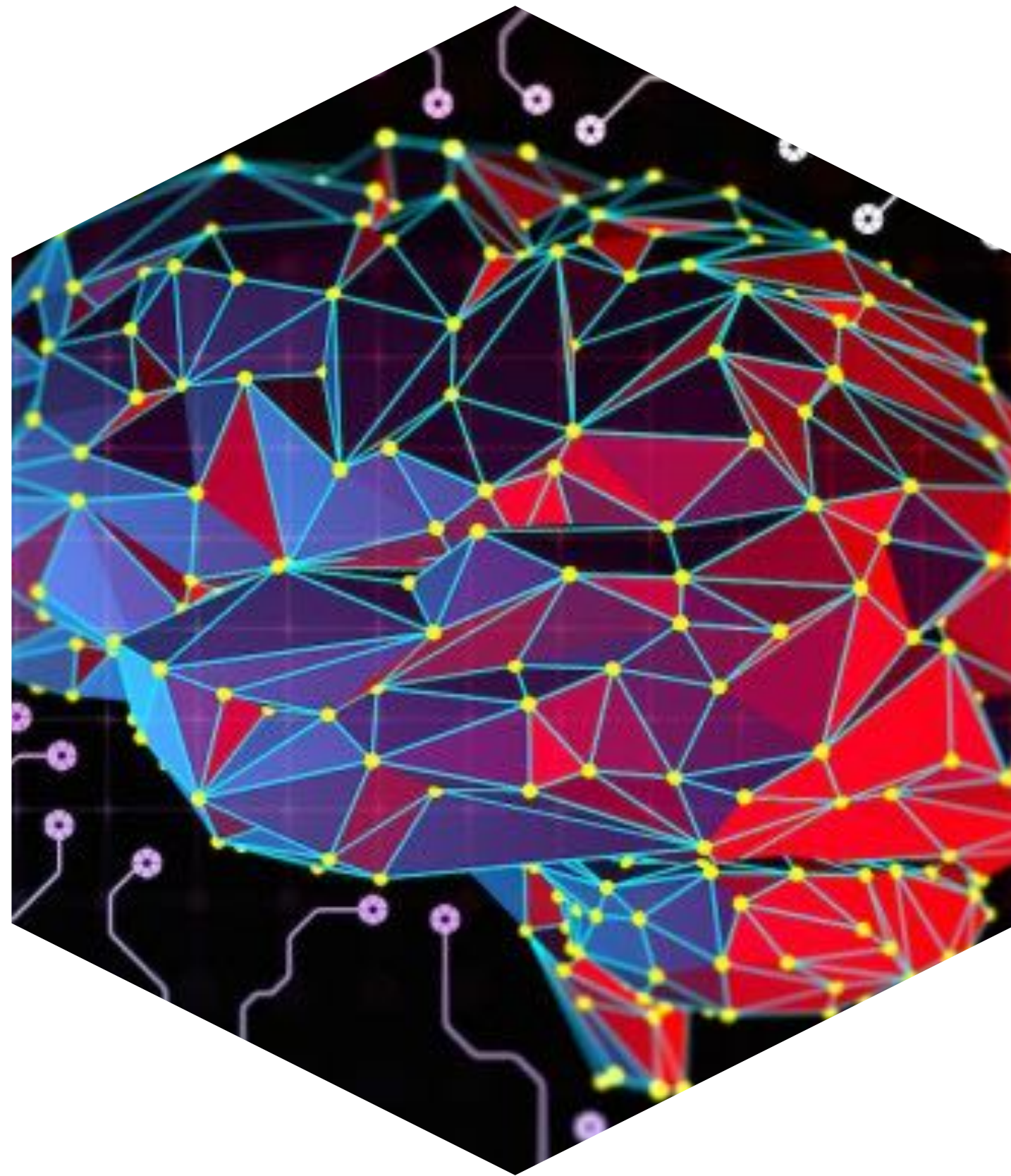
The background is a solid blue color with a complex, abstract pattern of white and light blue lines and circles. The lines form a network-like structure, with some circles acting as nodes or hubs. The overall effect is a high-tech, digital aesthetic.

DATA STRUCTURES AND ALGORITHMS



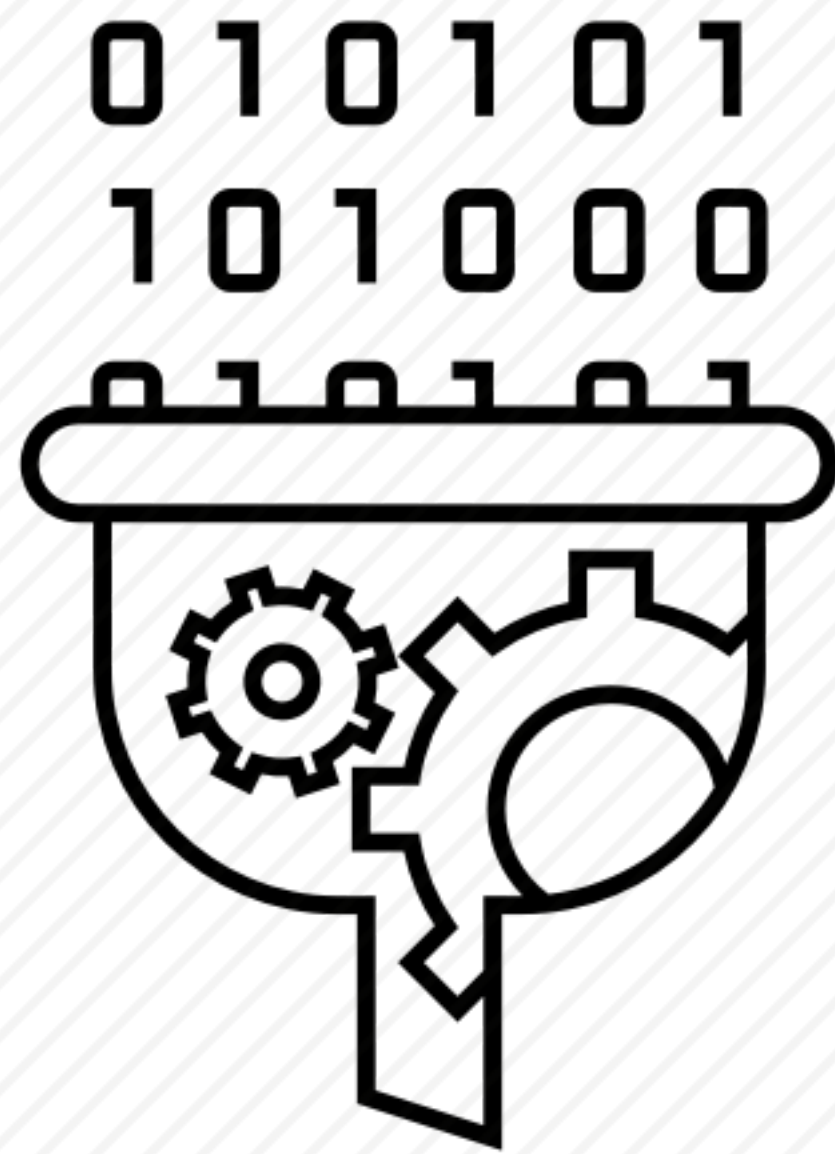
Jan P.

[GitHub.com/fafk](https://github.com/fafk)
<https://fafk.github.io>

JavaScript, Rust, Java, LISP & crypto
patomil@gmail.com

OVERVIEW

- ✓ Why algorithms and data structures?
- ✓ Arrays & Objects
- ✓ Queue, Stack
- ✓ Trees
- ✓ Big O (time/space complexity)
- ✓ Recursion



algorithm: a step-by-step procedure for solving a problem

Computation:
input data
+
instructions (algorithm)
=>
output data

A computer program is a set of instructions (an algorithm) operating on data structures.

We can make our program run efficiently by choosing the right data structures and algorithms.

Basic data structures

Array

A sequence of indexed values.

```
const array = ["a", "b", "c"];
Array.of("a", "b", "c");
new Array("a", "b", "c");

console.log(array[0], array[1]);
// => "a"
console.log(array[1]);
// => "b"

array.push("new element");
array.pop("get last element");
console.log(array.length);
// => 3
```

Arrays are everywhere.

Queue

FiFo - first in, first out

```
const queue = [];

// put value on end of queue
queue.push(1);

// take first value from queue
queue.shift();
```

“Queue” and “stack” are commonplace terminology.

Stack

FiLo - First in, last out

[StackOverflow.com?](https://stackoverflow.com/)

```
var queue = [];

// push value onto the stack
queue.push(1);

// take the value from the top
var value = queue.pop();
```


There are no “maps” on the low level. We can have a look at how a map can be implemented with an array using modulo.

Objects (Maps)

Object:

```
{  
  name: "Jan",  
  hobbies: ["JavaScript", "Books"],  
  height: 181  
}
```

Objects store structured data.

Sometimes we call objects “key value stores”.
Sometimes we call them “hash maps”.
Sometimes we call them “dictionaries”.

Array of objects:

```
[{  
  name: "Jan",  
  hobbies: ["JavaScript", "Books"],  
  height: 181  
}, {  
  name: "Bob",  
  hobbies: ["PHP", "Hiking"],  
  height: 176  
}, {  
  name: "Alice",  
  hobbies: ["Haskell", "Cycling"],  
  height: 171  
}]
```

Objects are native to JavaScript —that’s not the case in low-level programming languages.

We access values (after the “:”) by keys:
{“myKey”: “I am a string value!”, “numberKey”: 5}

```
const obj = {"myKey": "I am a value!", "numberKey": 123};  
console.log(obj.myKey);  
=> "I am a value!"  
console.log(5 * obj.myKey);  
=> 25
```

Traversing an array

```
const arr = [2, 4, 6];

let i = 0;

while (i < arr.length){
  console.log("Value:", arr[i]);
  console.log("Value squared:", arr[i]**2);
  i++; // i = i + 1;
}

for (i = 0; i < arr.length; i++) {
  console.log(arr[i]);
}
```

We traverse arrays to process the data stored in them.

In real life we do this very often.

Let's code a real-life example: how many city bikes are there in the 2nd district?

<https://repl.it/@fafk/CityBikes>

```
const arrOfObjects = [{val: 2}, {val: 4}];
i = 0;

while (i < arrOfObjects.length) {
  console.log("Value:", arrOfObjects[i].val);
  i++; // i = i + 1;
}
```

Sorting arrays

You will probably never need to write a sorting algorithm outside of an interview.
And yet, people might ask about it.

Learning about sorting algorithms is a good segue into a discussion on complexity.

👉 **Bubble Sort** <https://repl.it/@fafk/BubbleSort>

👉 **Merge Sort**

Sorting in JavaScript

```
console.log(["c", "b", "a"].sort());  
=> ["a", "b", "c"]
```



```
console.log([5,3,10].sort());  
=> [ 10, 3, 5 ]
```



```
console.log([5,3,10].sort((a, b) => a > b))  
=> [3, 5, 10]
```

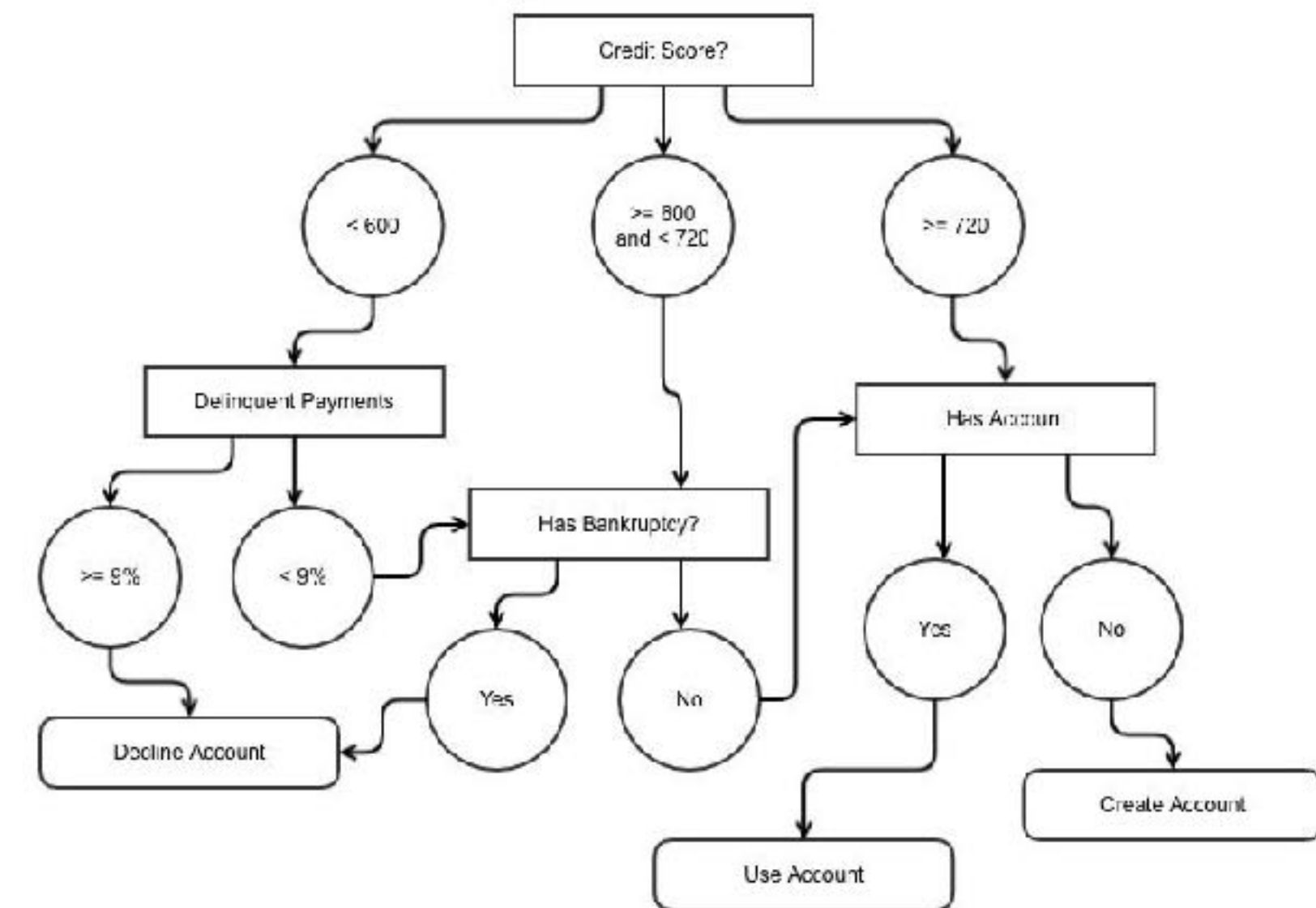
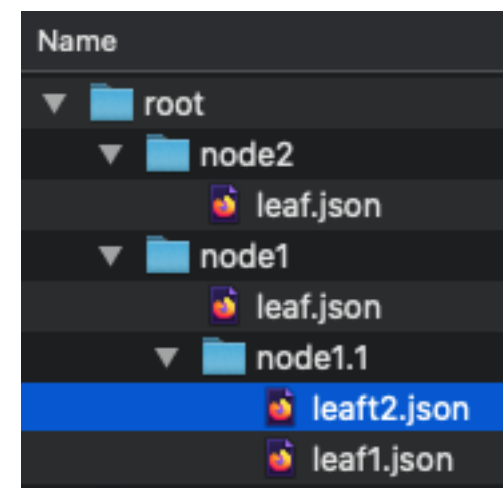
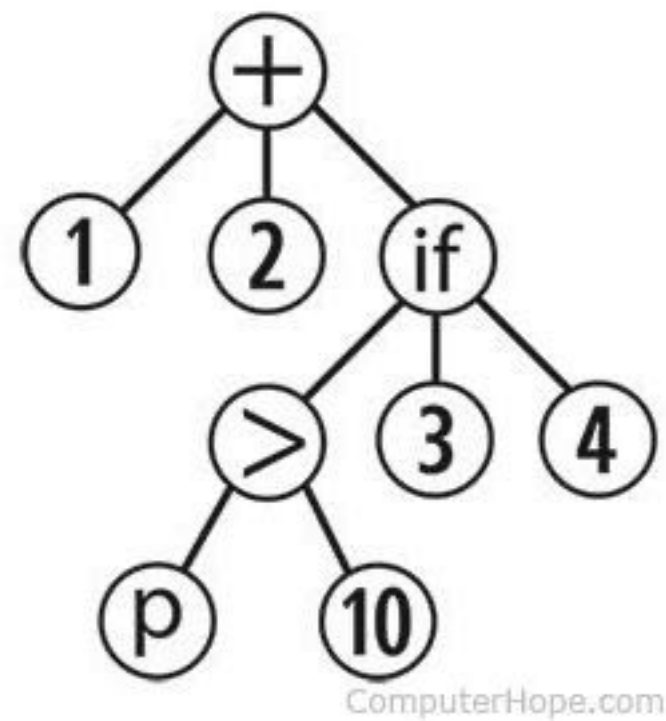


```
const unsorted = [{name: "Zeus"}, {name: "Andromeda"}];  
console.log(unsorted.sort((a, b) => a.name > b.name))  
=> [{name: "Andromeda"}, {name: "Zeus"}]
```



Trees

Explain why the tree on the left is a computer program/algorithm.



Trees. are. everywhere.

Trees?



Define and draw a graph, show what makes graph a tree.

Directed acyclic graphs. Graphs?

Root node, node, leaf.

Parent, child. Ancestor, descendant.

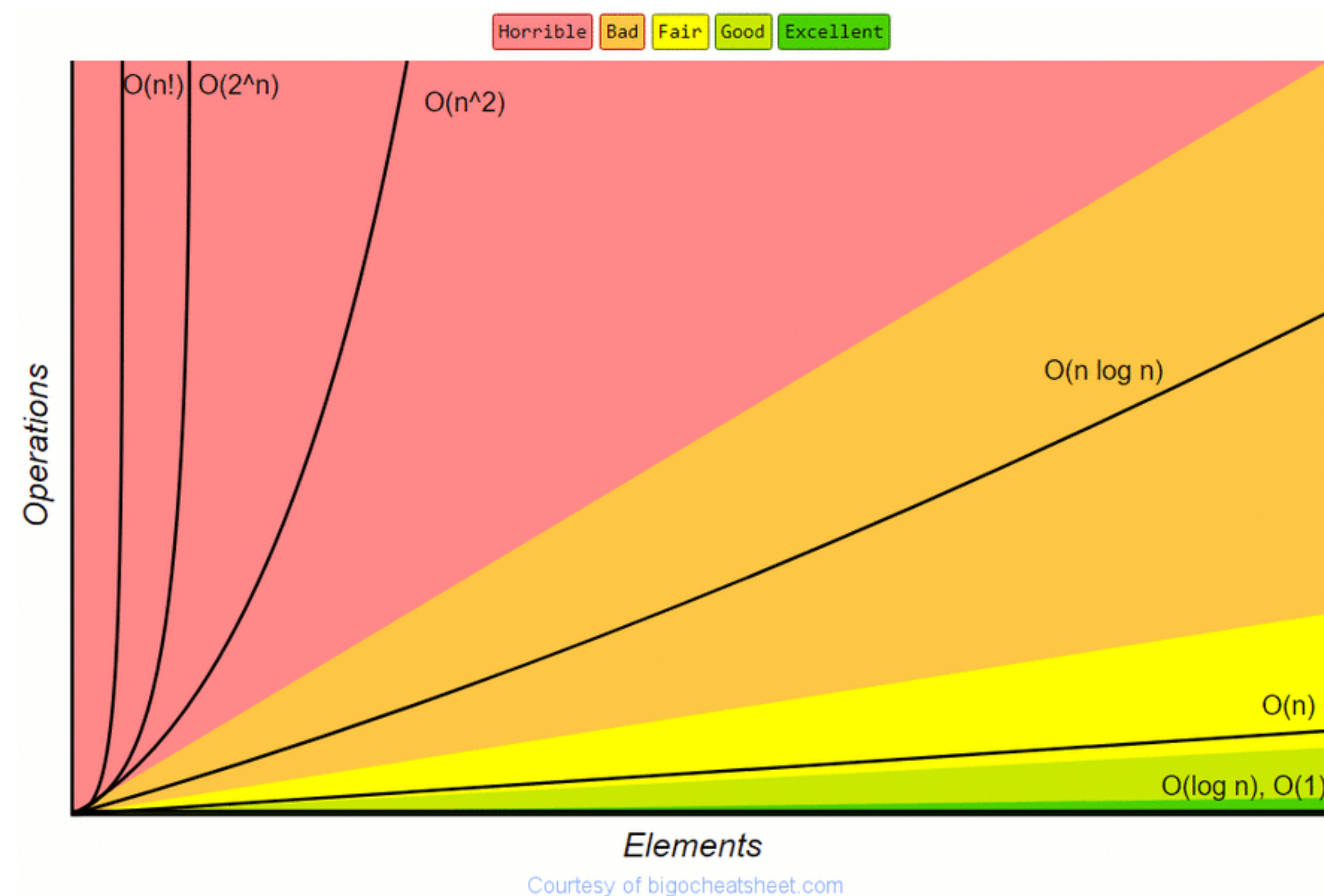
Binary tree.

Complexity

Explain big O notation (worst case stripped off constants).

Explain motivation: to reason about algorithms. Explain on bubble sort.

For an input of length N ... how many steps will an algorithm take to finish?



Recursion

Let's talk about what recursion is, how it's used to traverse lists and how it's used to implement merge sort.

Requires a stop condition.

Can be used instead of iteration.

<https://twitter.com/joelnet/status/1214268814294016000>

Find how many bikes are there in the 2nd district using recursion!

<https://repl.it/@fafk/CityBikes>



Thanks!

Any questions?

You can find me at patomil@gmail.com