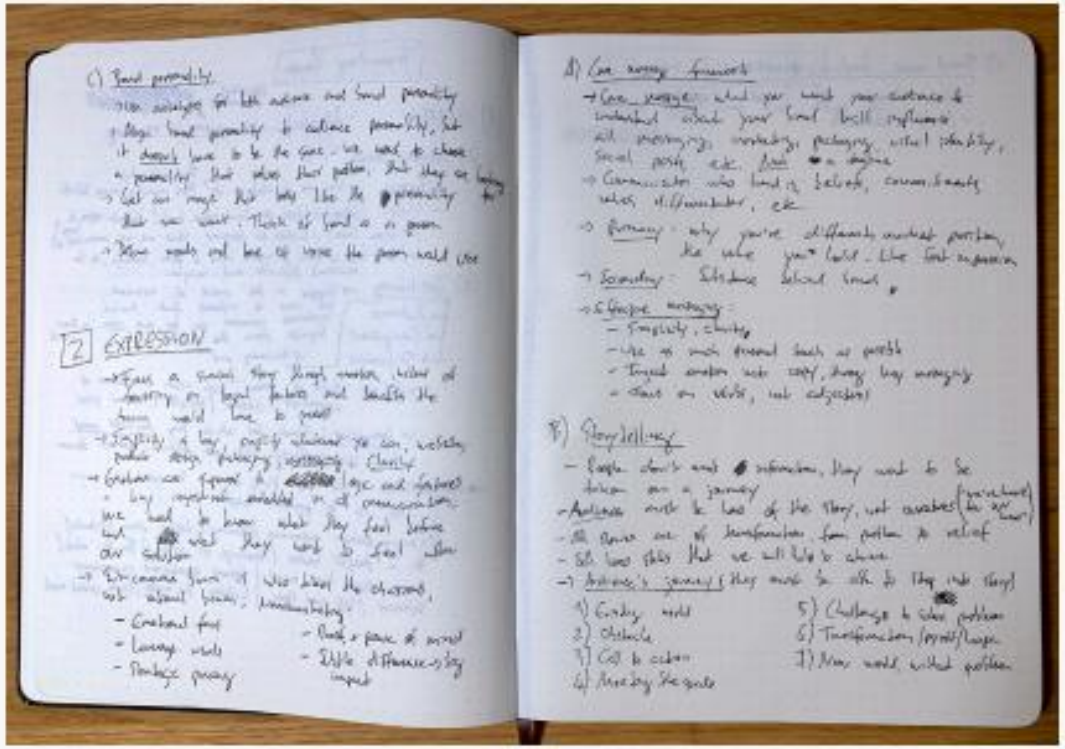
★ Anurag Nampally ★

- Btech in Information Technology, VNR VJIET, Hyderabad ($^{20}_{18}$ - $^{20}_{22}$)
- Masters in computer science, Indiana University, USA ($^{20}_{23}$ - $^{20}_{25}$)
  Bloomington

- ADP Member Technical Intern (3-2 btech, 2020) (3M's)
- JP Morgan chase Software Engineer Intern (4-2 btech, 2022) (5M's)
- AccioJob (4-2 btech, 2022 - present) (1.5 Yr +)
- JPMorgan chase Software Engineer 1 (2022 - 2023) (1 Yr)
- Algotutor (2023 - present) (5M's +)

👩‍💻 **You need to code along with me!** You will learn **ZERO** JavaScript skills by just sitting and watching me code. You have to code **YOURSELF**!

📝 **If you want the course material to stick, take notes.** Notes on code syntax, notes on theory concepts, notes on everything!

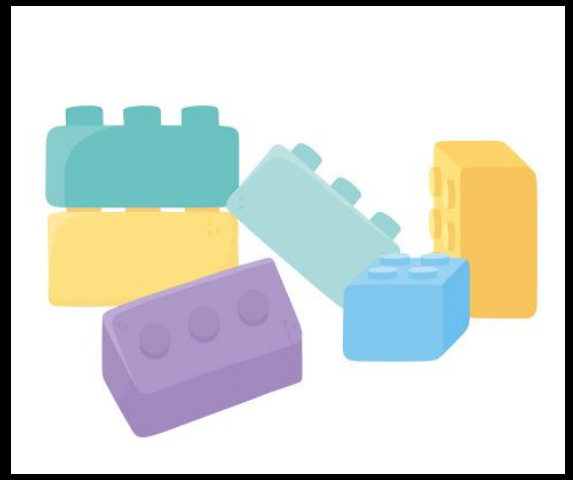


Totally non-coding... Try to understand a single word 😂

😱 **If this is your first time ever programming, please don't get overwhelmed.** It's 100% normal that **you will not understand everything** at the beginning. ***Just don't think "I guess coding is not for me"!***

In the first sections of the course, don't bother understanding WHY things work the way they do in JavaScript. Also, don't stress about efficient code, or fast code, or clean code. While learning, we just want to make things **WORK**. We will understand the **WHY** later in the course.

What is programming ?

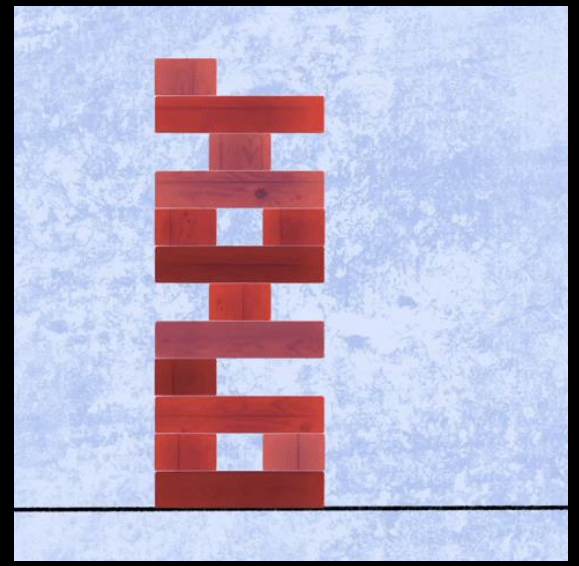


Let say you have some buidling blocks. You and your friend want to create something cool with them, like a tower or castle.

Now You have a plan in your mind, but you need to tell your friend what to do.

Why should your friend listen to you and follow your instructions. Imagine you have a magic wand. You say things like "Put a blue block on top of the red one" or "Stack three green blocks."

Programming is like giving instructions to your friend, who is helping you build. But there's a special language you both need to speak, and that language is called "code."

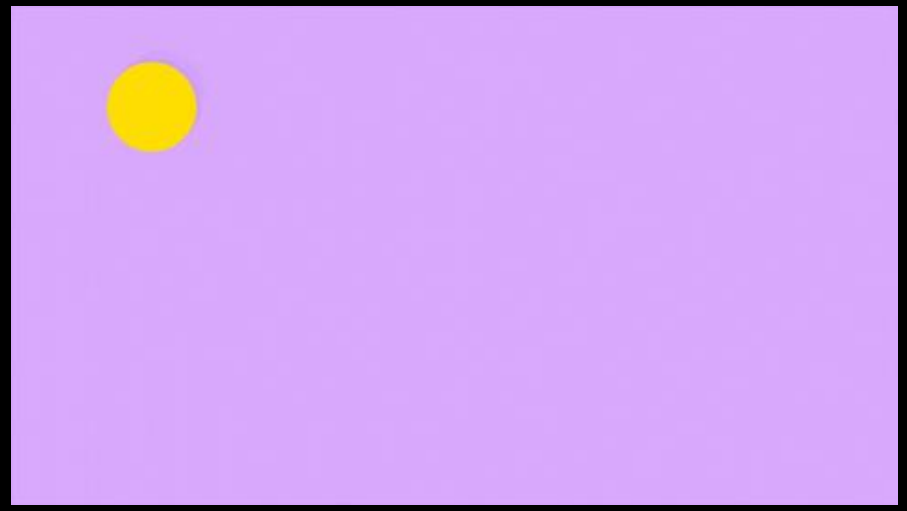


Sometimes, your friend might make a mistake and put a block in the wrong place. No worries! You can use your magic wand again to correct the instructions and help your friend fix it.

Sometimes, your friend might make a mistake and put a block in the wrong place. No worries! You can use your magic wand again to correct the instructions and help your friend fix it.

Once your tower or castle is complete, you both step back and admire your work. You can even add more instructions to make it do special things, like adding a door that opens or making a flag on top that can wave.

That's what programming is like! It's giving clear instructions (using code) to make your ideas come to life using computers. Just like building with colorful blocks, but in a magical language that computers understand.

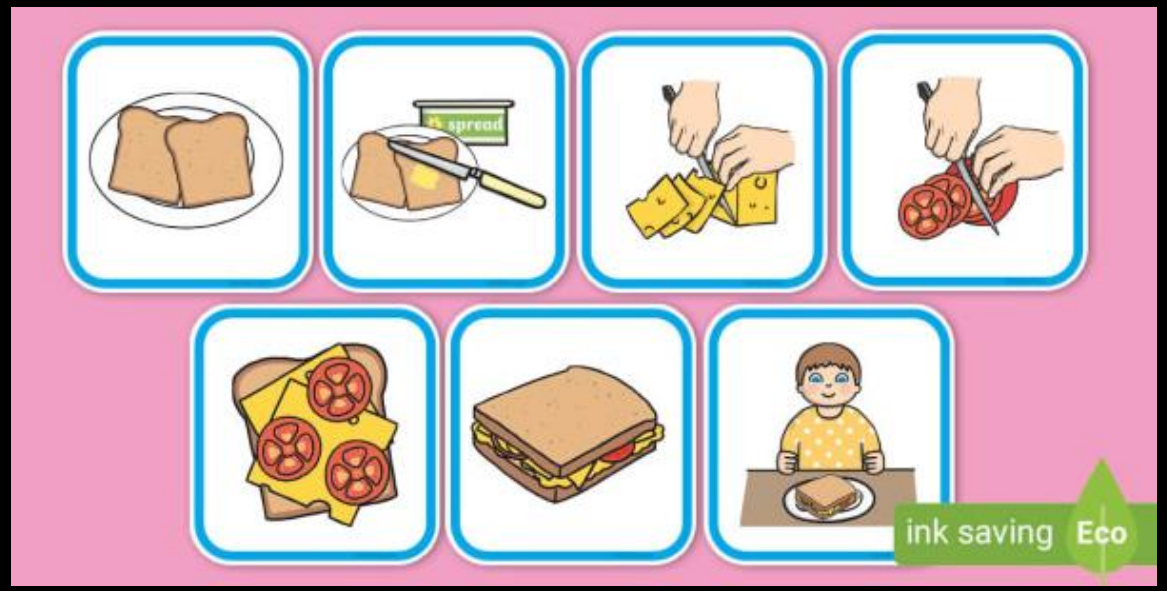


Ingredients: To make the sandwich, you need bread, peanut butter, and jelly.

Steps: The recipe (algorithm) gives you a set of clear steps to follow:

a. Take two slices of bread.
b. Spread peanut butter on one slice.
c. Spread jelly on the other slice.
d. Put the two slices together with the spreads facing each other.
e. Press gently to make the sandwich.

Result: If you follow these steps correctly, you'll end up with a delicious peanut butter and jelly sandwich.



What is an Algorithm ?

Finding the Largest Number in a List

Inputs:
- A list of numbers.

Output:
- The largest number in the list.

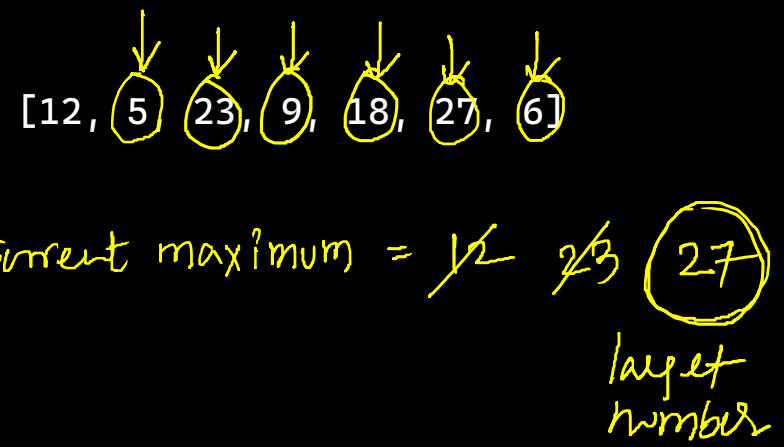
Steps:
1. Start with the first number in the list and call it the "current maximum."
2. Compare the "current maximum" with the next number in the list.
    1. If the next number is larger than the "current maximum," update the "current maximum" to be the next number.
    2. If the next number is not larger, keep the "current maximum" as it is.
3. Repeat step 2 for each number in the list until you have compared all the numbers.
4. The "current maximum" after going through the entire list is the largest number.

Algorithm
(you should think this logic)

Example

Steps:

1. Start with the first number in the list and call it the "current maximum."
2. Compare the "current maximum" with the next number in the list.
    1. If the next number is larger than the "current maximum," update the "current maximum" to be the next number.
    2. If the next number is not larger, keep the "current maximum" as it is.
3. Repeat step 2 for each number in the list until you have compared all the numbers.
4. The "current maximum" after going through the entire list is the largest number.



5 > cm ⇒ 5 > 12 ✗

23 > cm ⇒ 23 > 12 ✓

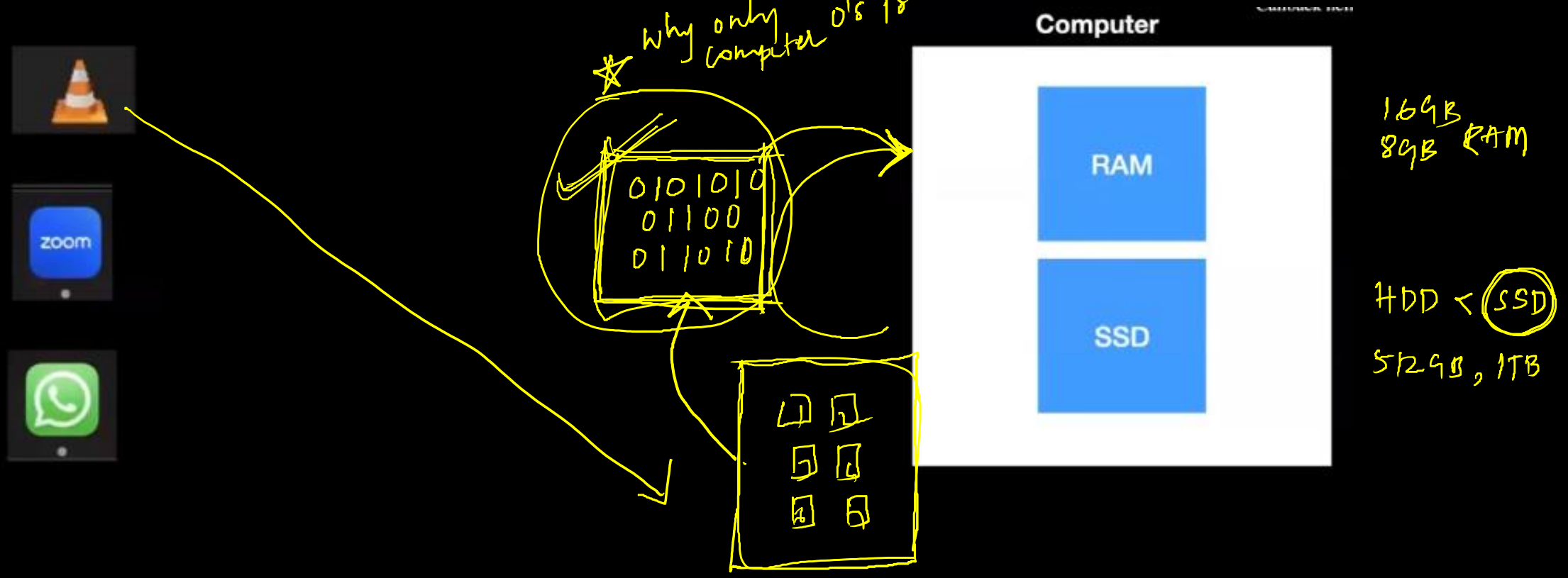
9 > cm ⇒ 9 > 23 ✗

18 > cm ⇒ 18 > 23 ✗

27 > cm ⇒ 27 > 23 ✓

6 > cm ⇒ 6 > 27 ✗

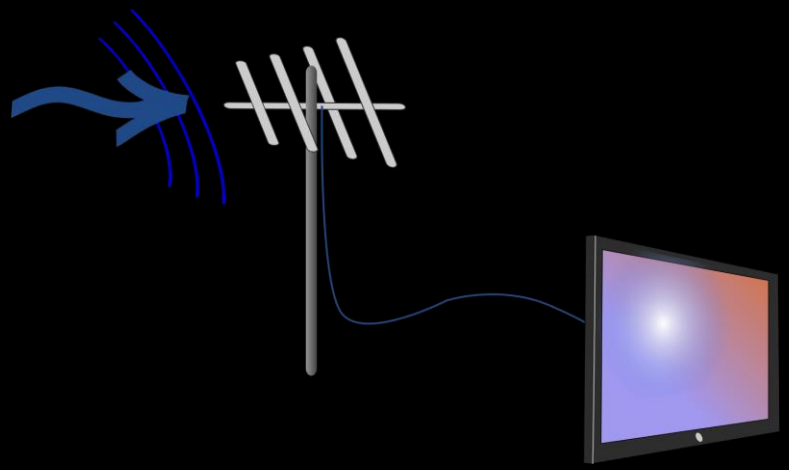
Why do we need languages ? (C++, Java, Python, JavaScript, etc.....)

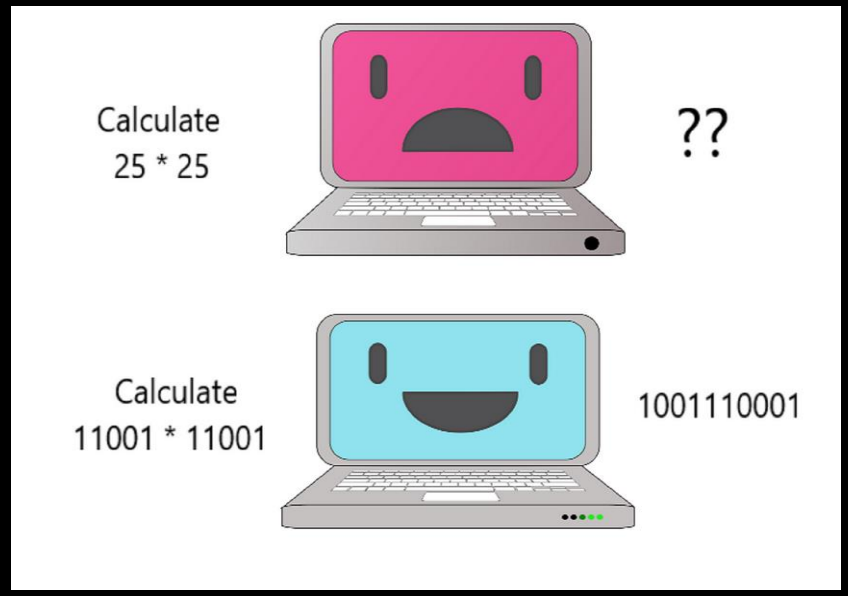


Computers can understand only binary 0’s and 1’s

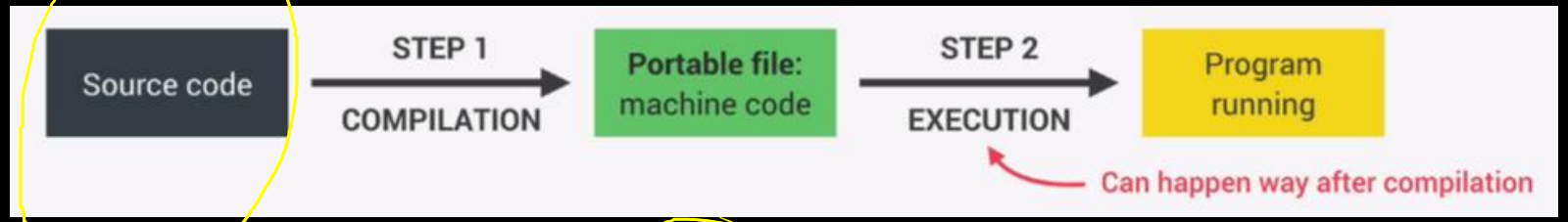
1.In the morning, you turn on your TV and watch news, cartoons, movies or whatever you like. Your TV receives Radio waves, that it changes to picture.

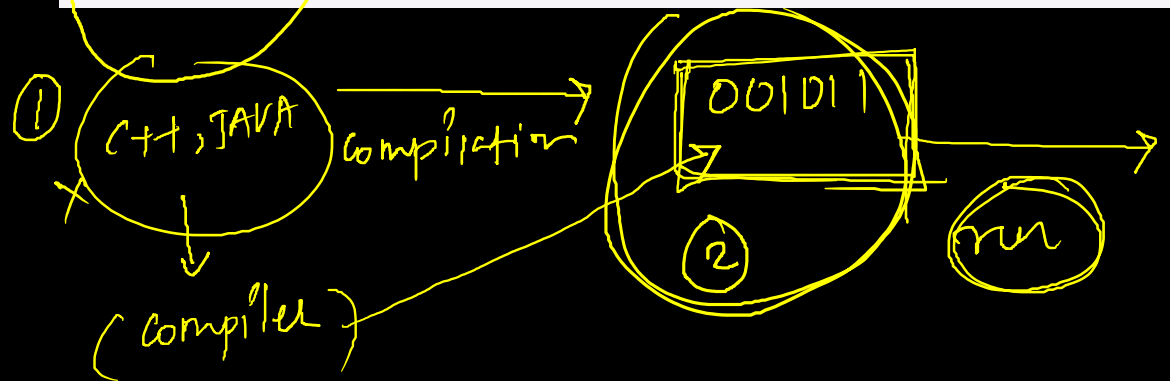
2.Your phone rings; you pick it up and start to talk with your friend, family member or colleague. During your conversation, your mobile changes your spoken words to electrical signals, that travel all the way to the receiver, and then his/her phone changes the signals to sound again, and Vice Versa.





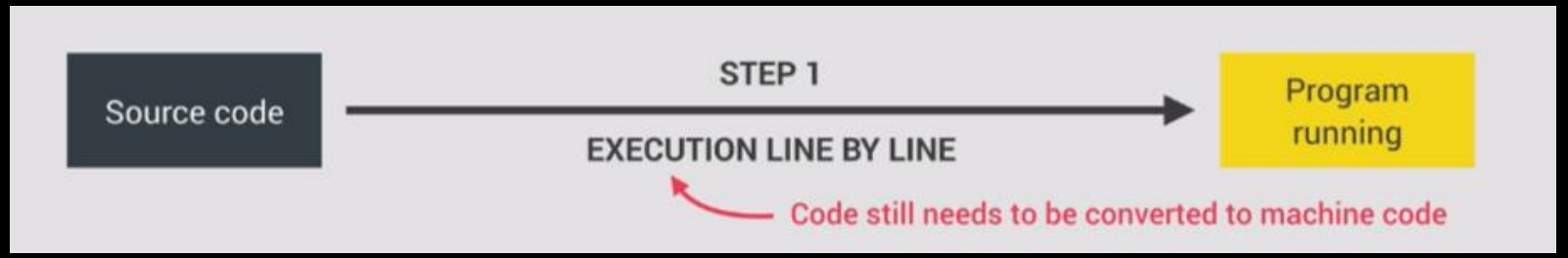
Compiled Language

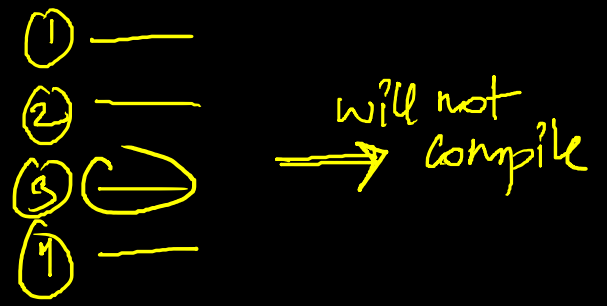


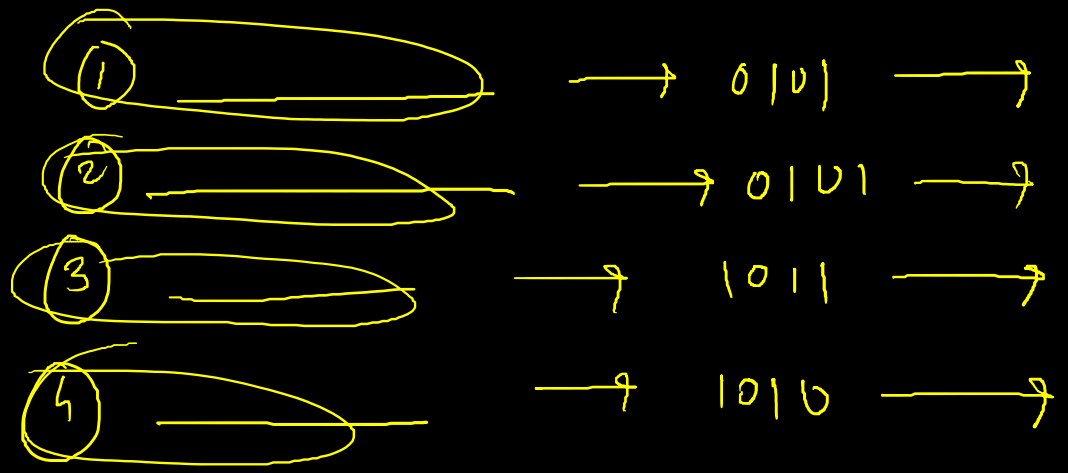


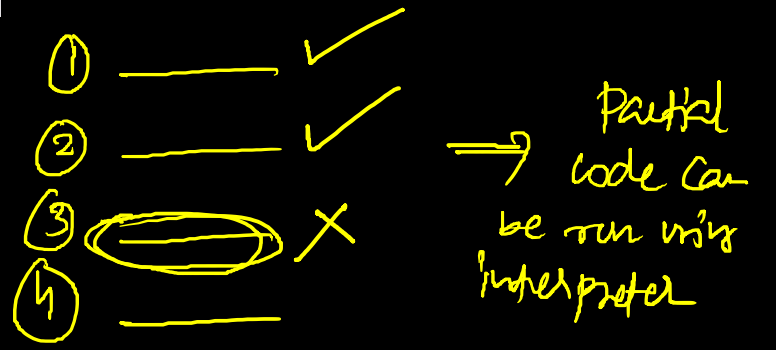
C++, JAVA
are compiled

Interpreted Language (JS, Python)









★ JS is a ~~interpreted~~

both compiled and interpreted

JIT (Just in time compilation)

Compiled vs Interpreted

Let's say you want to make a cup of tea, you go to Google and search for the ingredients list, this is what you came up with:

Tea powder, Water, Sugar (optional), Milk

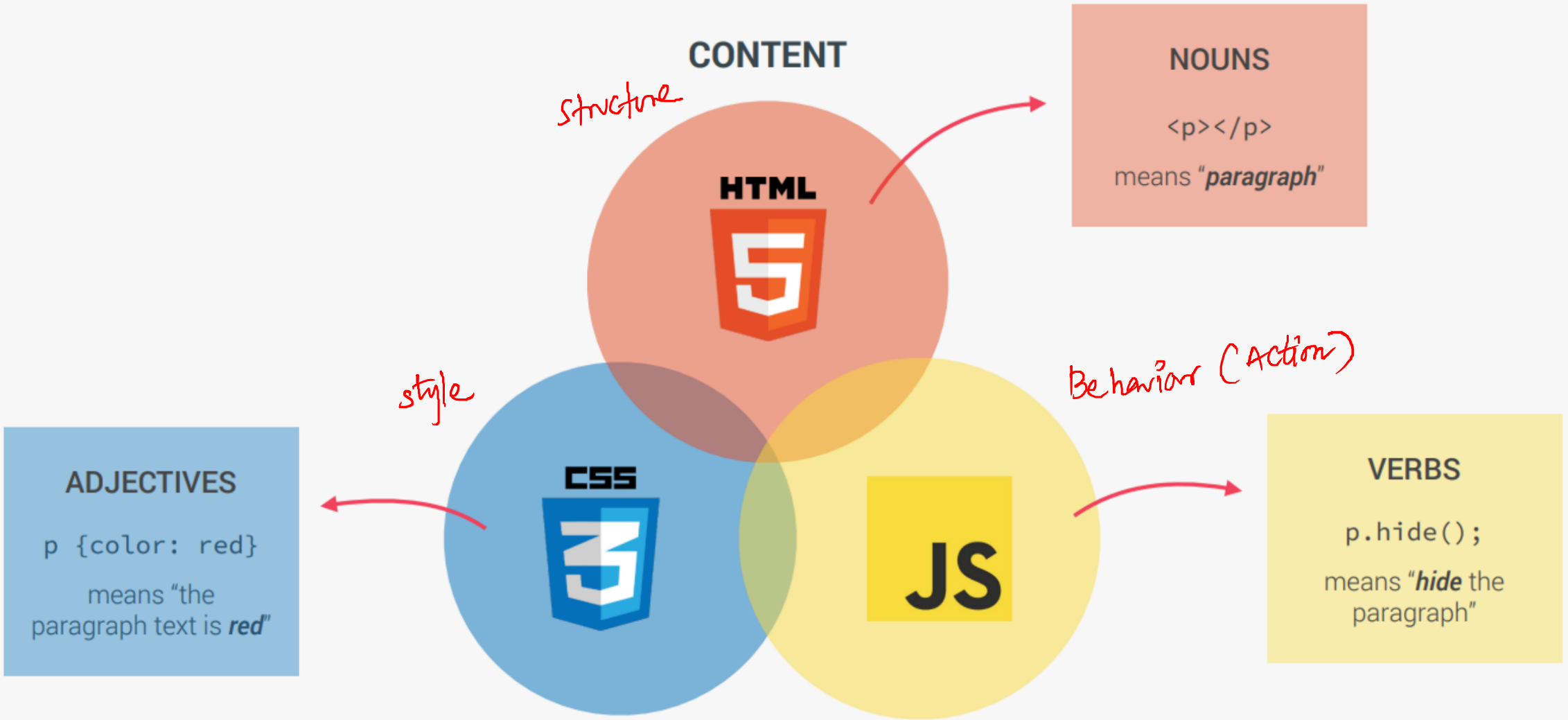
There are 2 ways to make the tea, the Compiler or the Interpreter way.

a. The compiler will first, before doing any preparation, organize all the ingredients in front of him, the specific amounts of every single ingredient, only then, he will prepare using the ready components of the tea.

b. The interpreter will take his cup and will start by reading the ingredients, line by line. he will boil the water, add the tea powder, mix them for a few minutes, add sugar or milk if desired, and then strain the tea into the cup.

The build (preparation) time of the compiler will be longer than the interpreter's. However, the run (steeping) time will be much shorter.

THE ROLE OF JAVASCRIPT IN WEB DEVELOPMENT



Front-end apps

Back-end apps

Dynamic effects and web applications in the browser 😍

Web applications on web servers

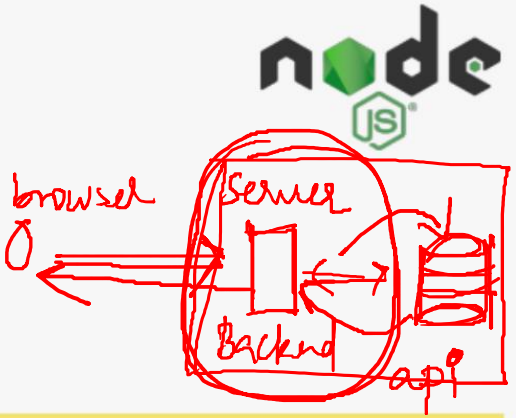
THIS COURSE

JS

ReactJS AngularJS VueJS

NextJS

JS



100% based on JavaScript. They might go away, **but JavaScript won't!**

Native mobile applications

Native desktop applications

JS

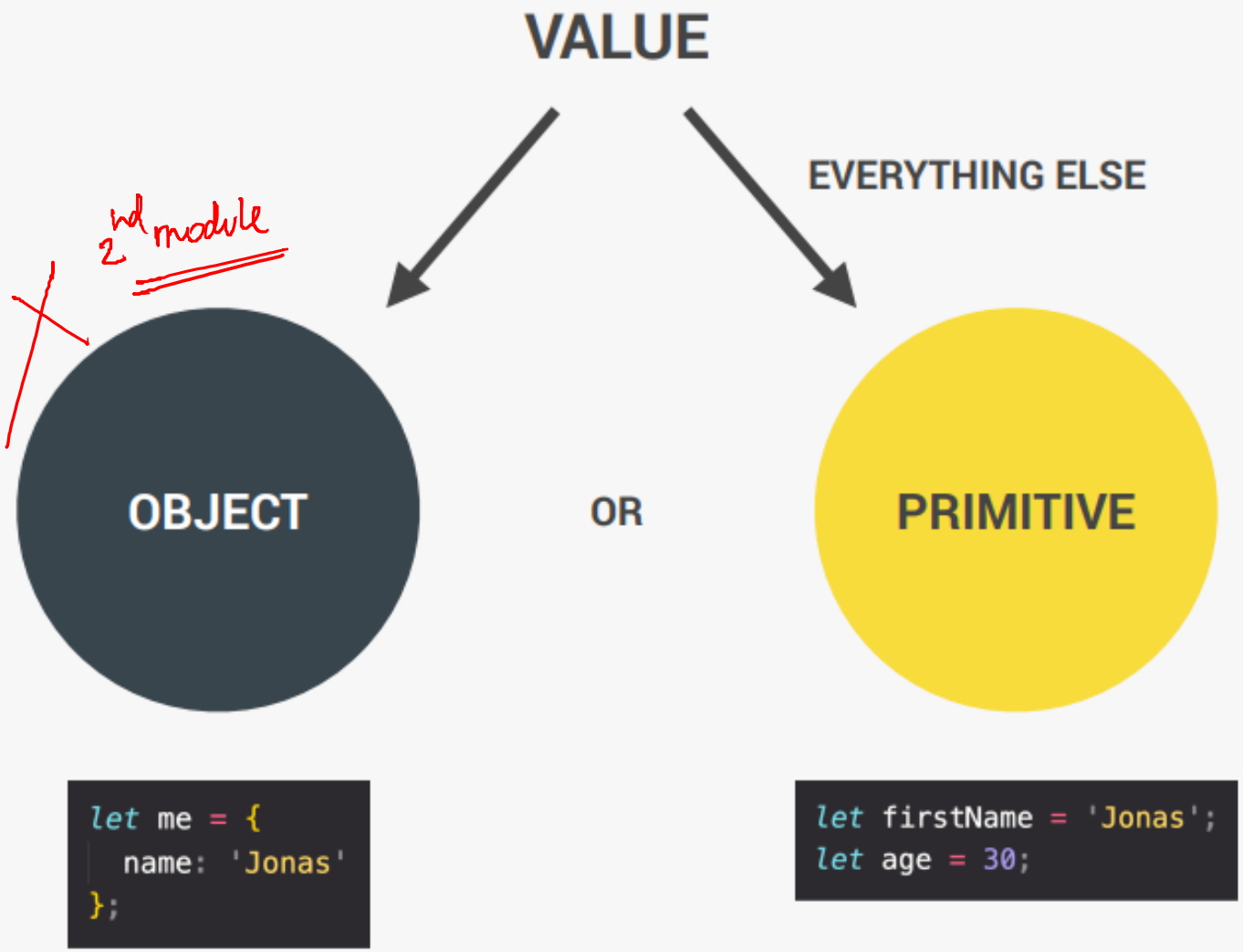
IONIC

JS

JS

1. Variable names cannot start with number
2. Can only have letters, underscores, numbers, dollar
3. CamelCase, 1st letter small
4. reserved keywords (new, function, name etc...)
5. all upperCase ⇒ Constant
6. Variable names should be descriptive, v. imp for cleaner code, name your variables according to what value it holds

OBJECTS AND PRIMITIVES



THE 7 PRIMITIVE DATA TYPES

1. **Number:** Floating point numbers 👉 Used for decimals and integers `let age = 23;`
2. **String:** Sequence of characters 👉 Used for text `let firstName = 'Jonas';`
3. **Boolean:** Logical type that can only be `true` or `false` 👉 Used for taking decisions `let fullAge = true;`
4. **Undefined:** Value taken by a variable that is not yet defined ('empty value') `let children;`
5. **Null:** Also means 'empty value'
6. **Symbol (ES2015):** Value that is unique and cannot be changed *[Not useful for now]*
7. **BigInt (ES2020):** Larger integers than the Number type can hold

☝️ **JavaScript has dynamic typing:** We do ***not*** have to manually define the data type of the value stored in a variable. Instead, data types are determined **automatically**.

Value has type, NOT variable!