Class (Mon – Th) : 8pm - 11pm (fixed)

15min Break 9 - 9:30pm

Fri : 7pm - 10pm , 8pm - 11pm (flexible)

★ Anurag Nampally ★

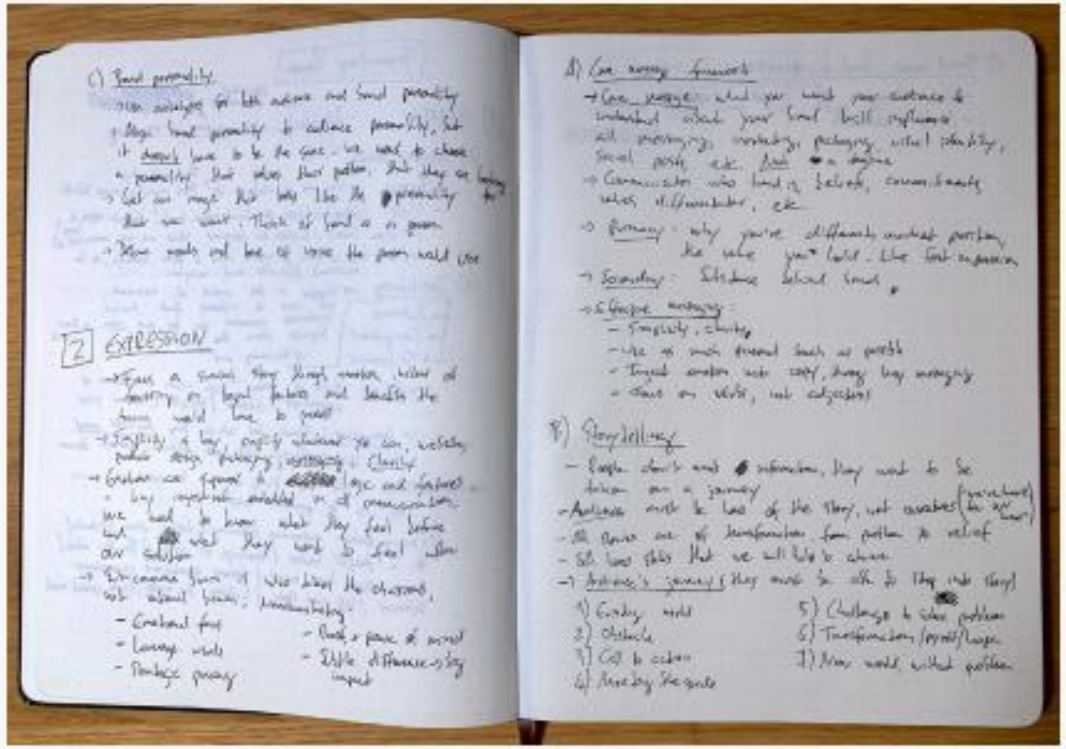
- Btech in Information Technology, VNR VJIET, Hyderabad (2018 - 2022)
- Masters in computer science, Indiana University Bloomington, USA (2023 - 25)

- 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**!

30 % Effort
70 % your effort

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



① write, understand

② 9 – 30min, 3 hrs
(middle you might be bored or sleepy)

③ Revise

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.

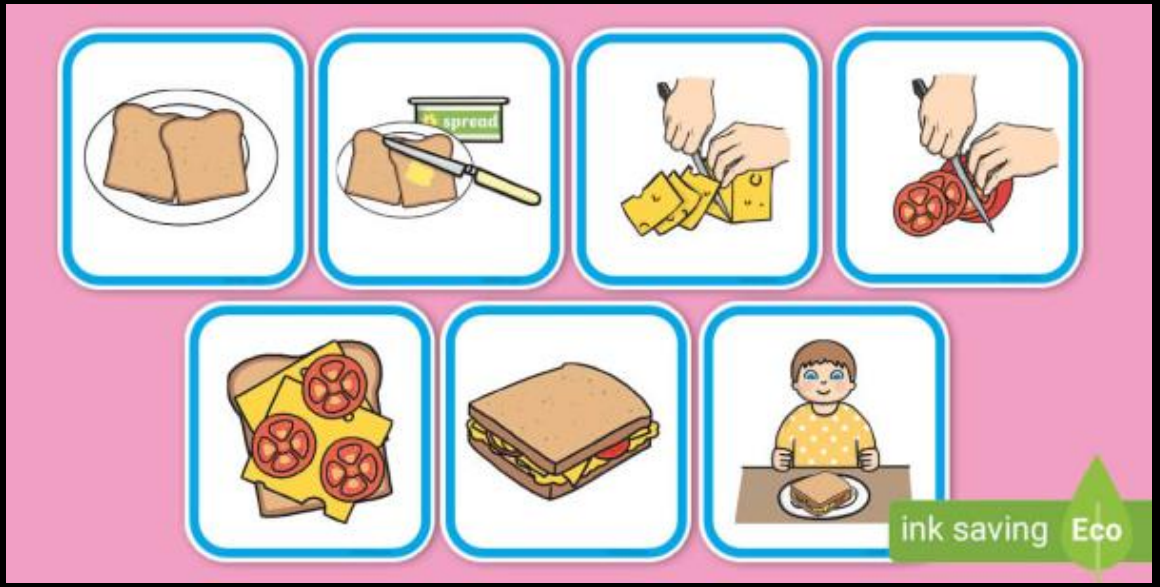
Ingredients: To make the sandwich, you need bread, peanut butter, and jelly.

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

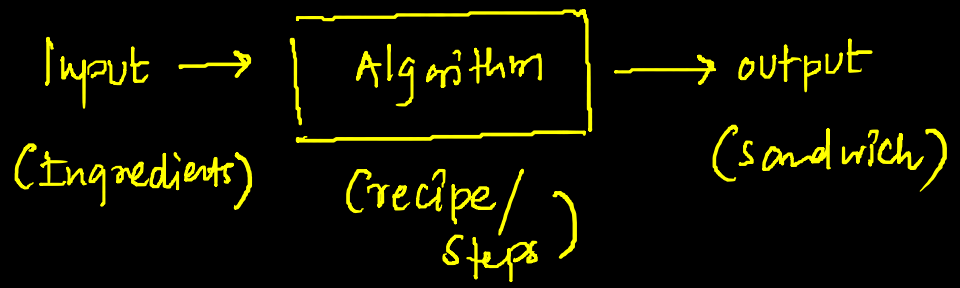
a. Take two slices of bread.
b. Spread peanut butter on one slice.
c. Spread jelly on the other slice.
d. Put the two slices together with the spreads facing each other.
e. Press gently to make the sandwich.

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

* Steps should be finite.
→ there should be some end.



What is an Algorithm ?

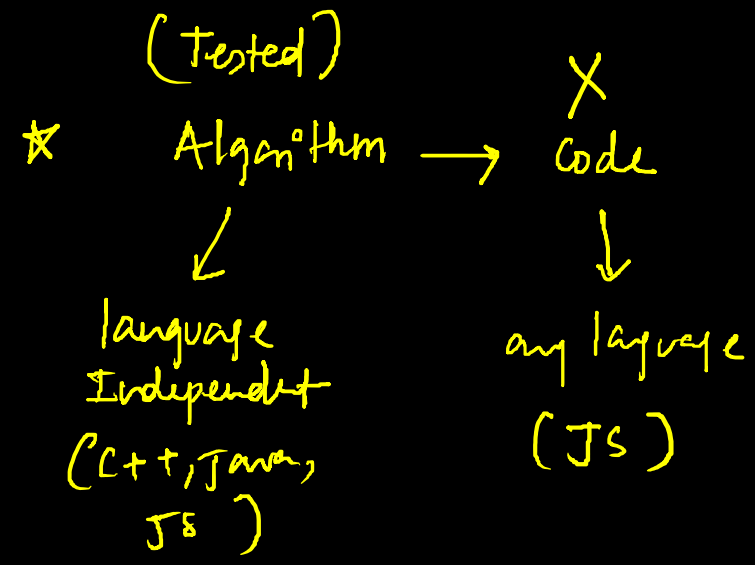


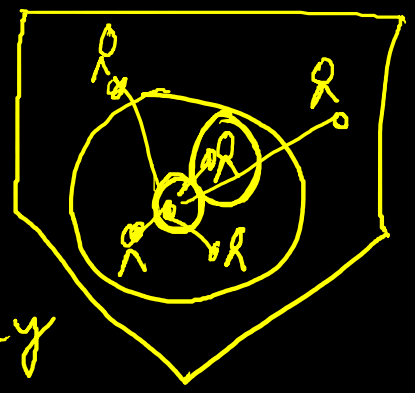
Finding the Largest Number in a List

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.





order given to nearest driver / delivery guy

Ip: your coord, all bike coords
Op: nearest bike

Think
Algorithm

How to solve this?

problem solving : coming up / thinking about an algorithm to solve such problems. → Code

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.

Input :

[12, 5, 23, 9, 18, 27, 6]

ouput :

largest number = 97

what if 10,000 numbers ?

- difficult to predict visually.
- Algorithm

1. Current Maximum = ~~12~~ ~~23~~ 97 → answer
2. next number = ~~5~~ ~~23~~ ~~9~~ ~~18~~ ~~27~~ 6

if → next number > Current Maximum

Current Maximum = next Number

if not → do nothing

5 > 12 X

23 > 12 ✓

9 > 23 X

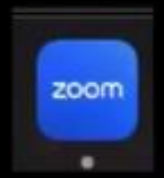
18 > 23 X

27 > 23 ✓

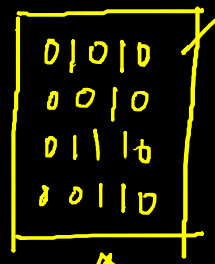
6 > 27 X

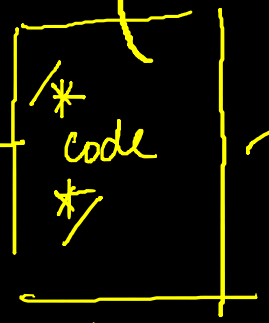
Why do we need languages ?

→ Because computers only understand 0's and 1's.
→ hence we use simple english based languages.

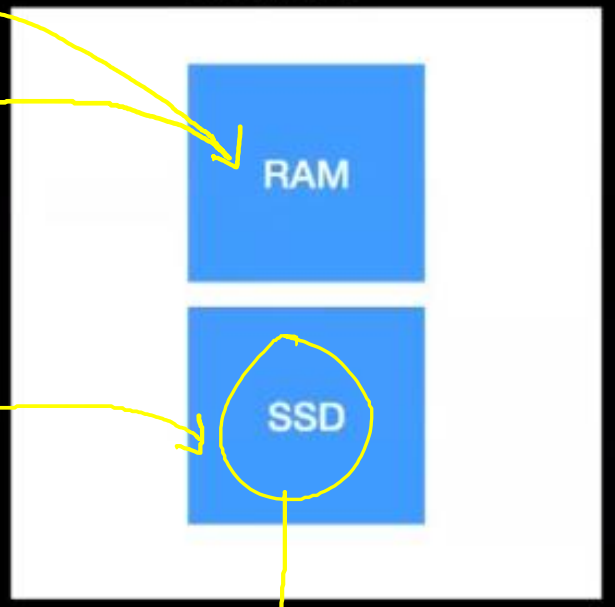


machine code





VLC
(C++, Java, python, JS, C)



(4GB, 8GB, 16GB)

Primary memory

Secondary memory

(512GB, 1TB)

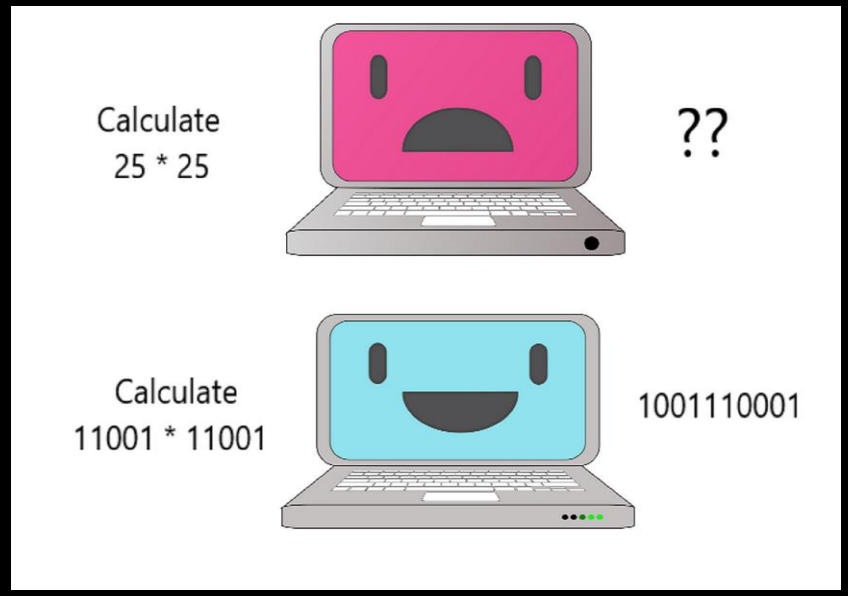
all files are stored here

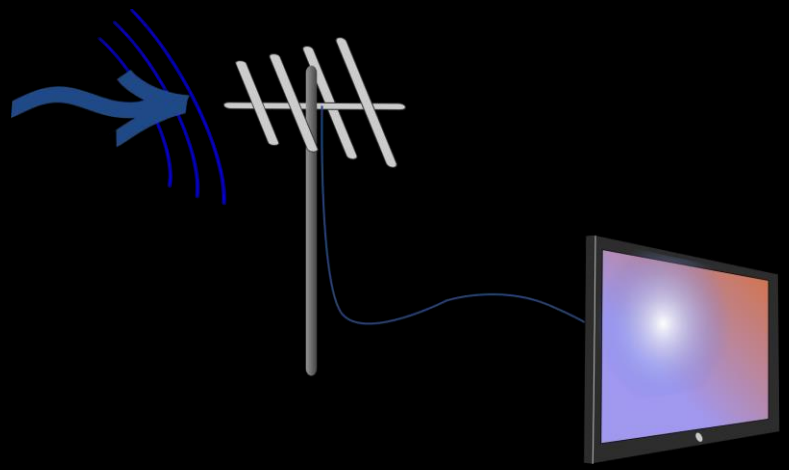
* Apps are running in RAM

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.

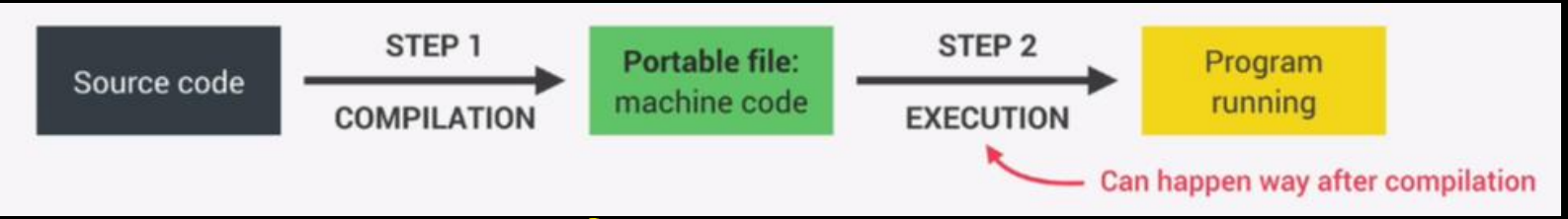


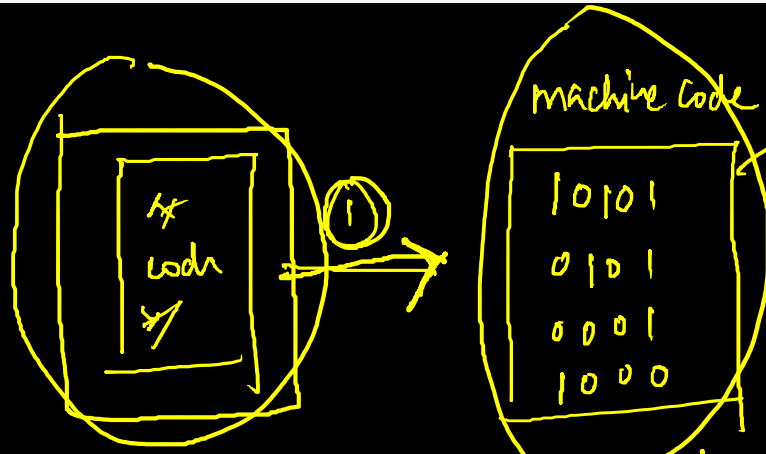


⇒ machines work on some electricity / battery.

⇒ electric signals → 0's (OFF, low voltage)
→ 1's (ON, high voltage)

Compiled Language





Shall this one run this in any computer you need not compile again

★ If there any any errors especially syntax error, compiler will notice and gives at an error, machine code is not generated.

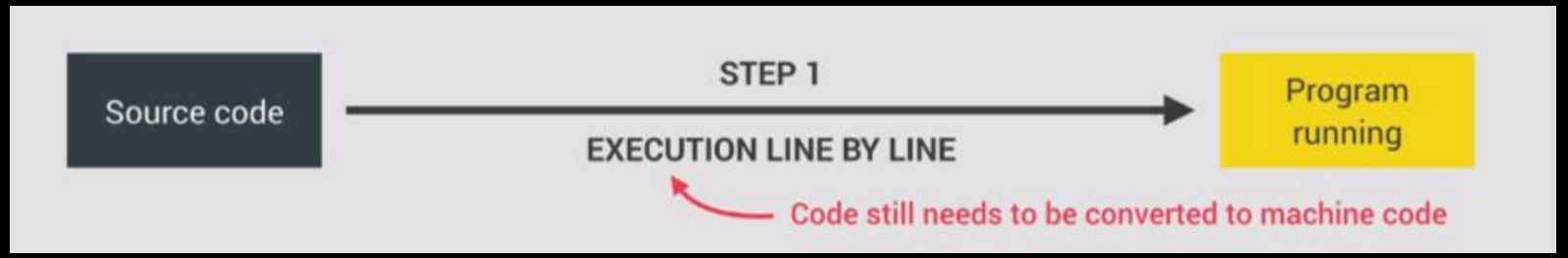
Compiler (software)

C++ → mingw Python → Python

Java → JVM JS → V8 engine

Interpreted Language

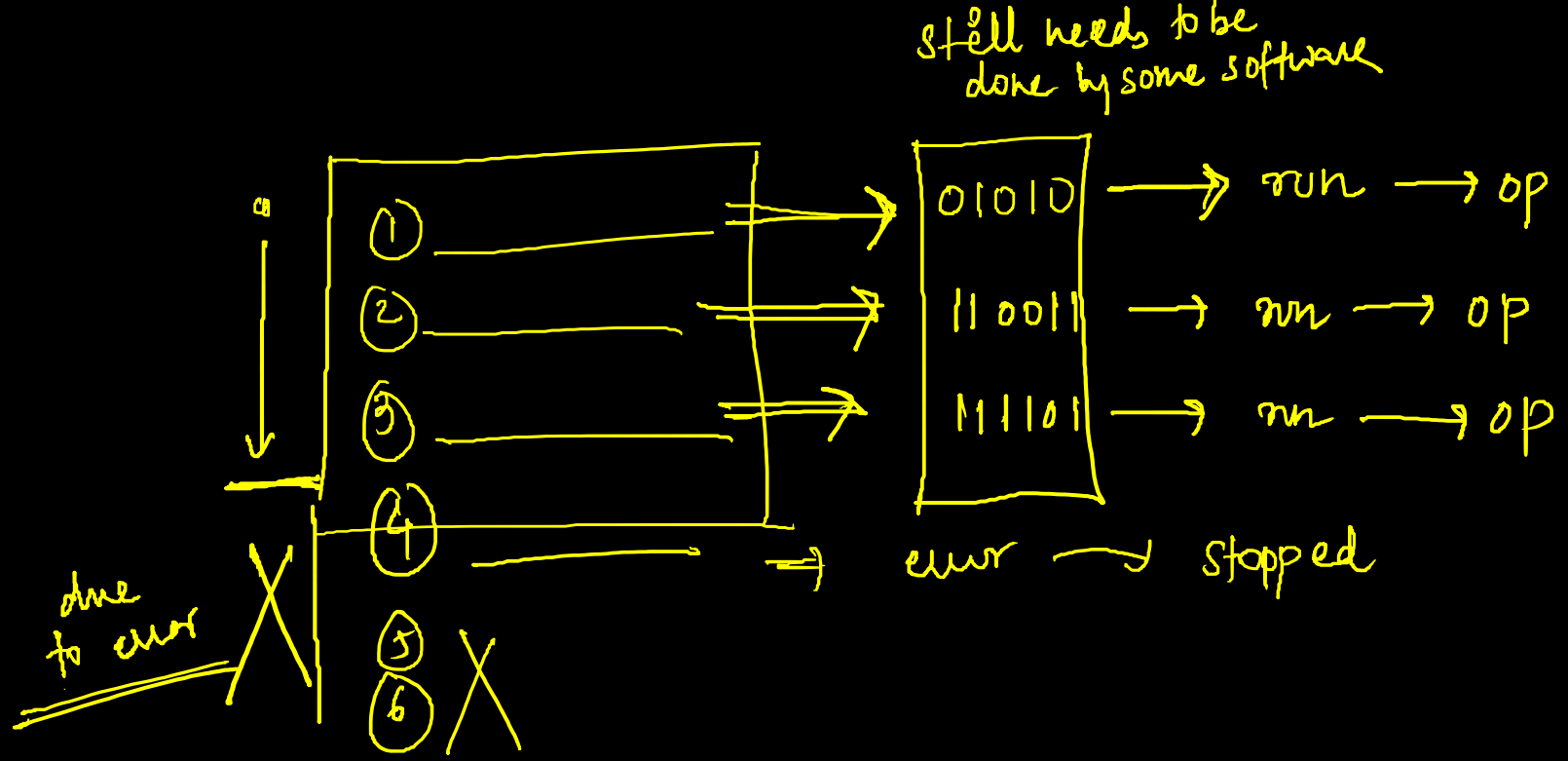
* partial op's are possible



= Python is a Interpreted language

= JS is a Interpreted language (wrong)

= Both Compiled & Interpreted (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

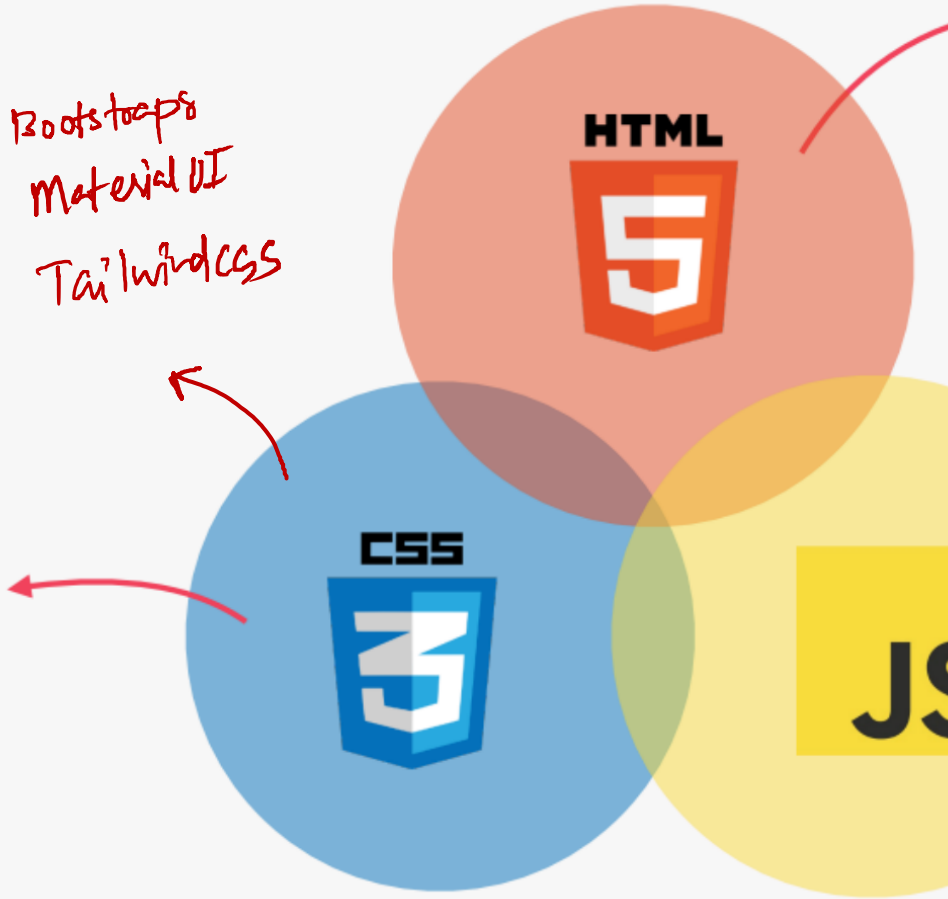
Break until 9pm

CONTENT

NOUNS

means "***paragraph***"

Bootstrap
Material UI
TailwindCSS



React,
Angular,
vue,

ADJECTIVES

```
p {color: red}
```

means "the paragraph text is ***red***"

VERBS

```
p.hide();
```

means "***hide*** the paragraph"

PRESENTATION

PROGRAMMING LANGUAGE:
BUILDING WEB APPLICATIONS

Front-end apps

Back-end apps

Dynamic effects and web applications in the browser 😍

Web applications on web servers

THIS COURSE

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

Native mobile applications

Native desktop applications

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