CYDEO

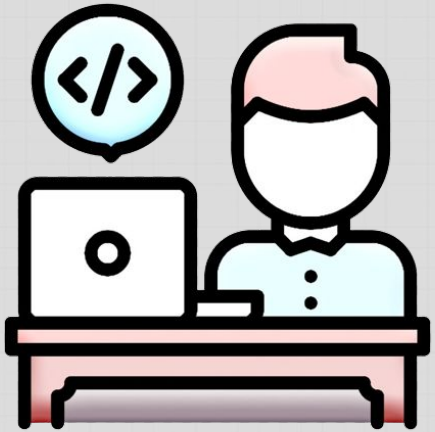
JavaScript Programming Day01

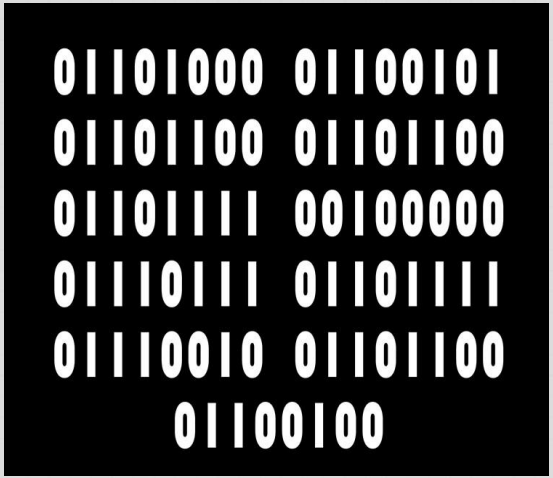
Content

- Introduction to JavaScript Programming
- VS Code Configurations
- Variables and Data Types
- Operators
- Decision Makings
- Loops
- Perplexity AI

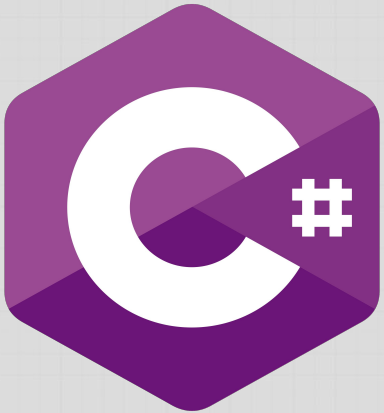
Programming Language

- A computer language
- Used by programmers to Communicate with computers





Most Popular Programming Languages

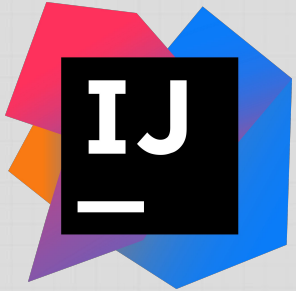


Why Learn JavaScript?

- The language of the web
- Beginner-friendly syntax
- High demand in the job market
- Versatile for both frontend and backend development

Different IDEs for JavaScript

- An integrated development environment (IDE) is a software application that provides comprehensive facilities to computer programmers for software development



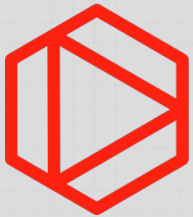
VS Code Extensions

Why do we need extensions in VS Code

- VS Code extensions are needed to enhance the editor's capabilities
- They make coding easier and more efficient by customizing the environment to fit our specific needs











Recommended VS Code Extensions

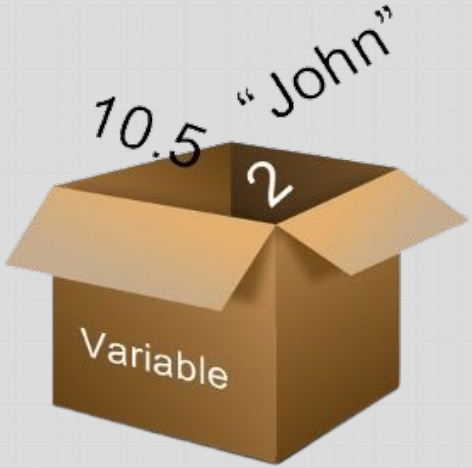
- Code Runner 
- JavaScript (ES6) code snippets 
- Prettier - Code formatter 
- Material Icon Theme 
- NPM 
- npm Intellisense 
- NPM Runner 

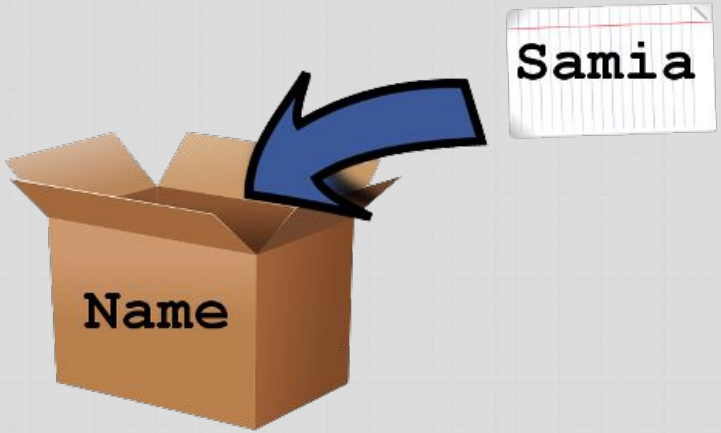
- Playwright Test for VSCode 
- Playwright Snippets 
- Playwright Test Snippets 
- Cucumber Gherkin 
- Better Comments 
- Tabnine 

Variables and Data Types

What Is A Variable?

- A variable is a container for storing a data value





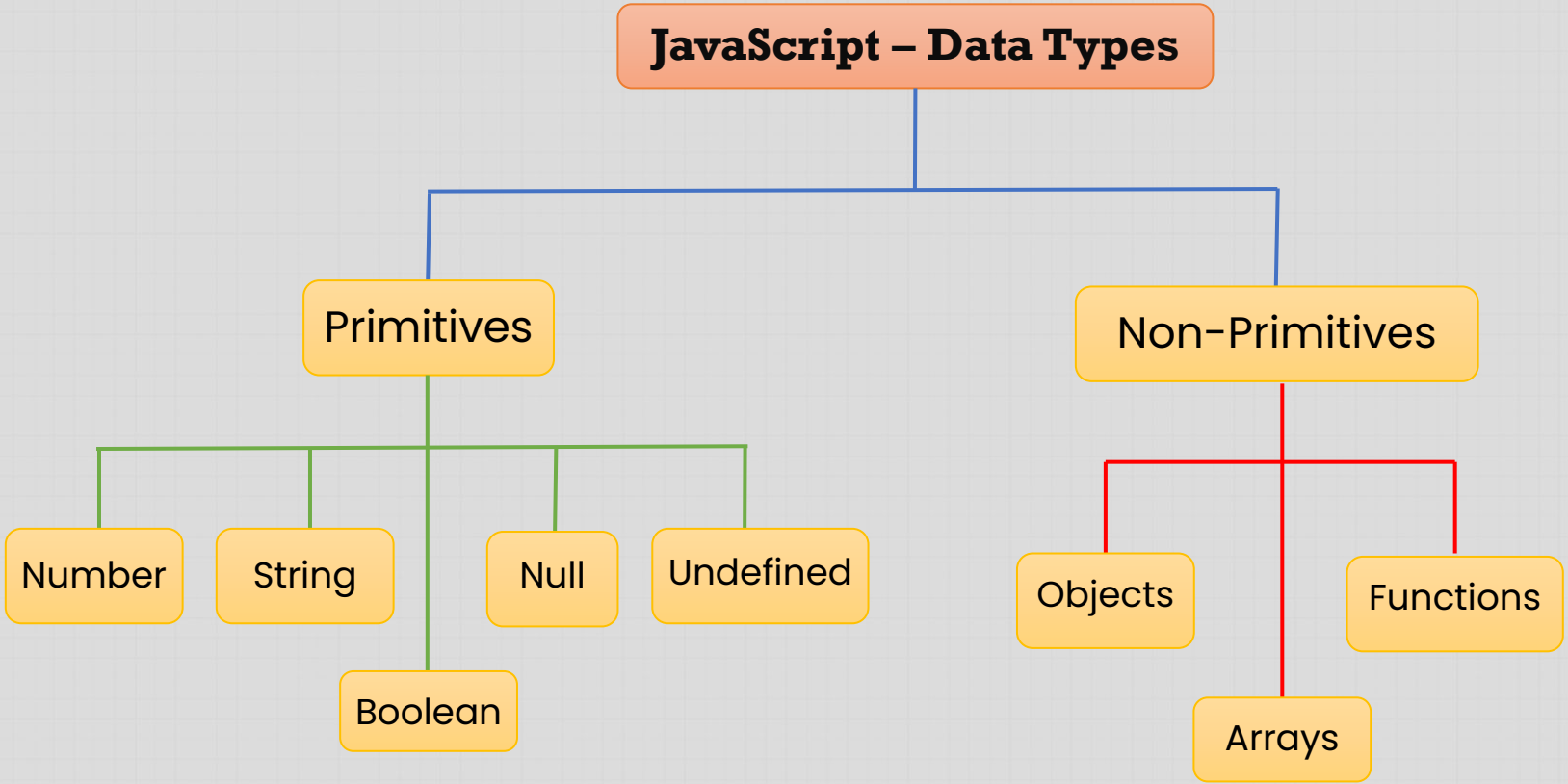
Variable

- Improves the reusability of the data
- Variables must be declared before use

```
var variableName = Data;

let variableName = Data;

const VARIABLE_NAME = Data;
```

```
var magicWord = "Wooden Spoon";

let number = 300;

const MAX_USER = 6;

var isEmployed = true;

let isMarried = false;
```

Data Types In JavaScript



Operators

Arithmetic Operators

NAME	OPERATOR	PURPOSE & NOTES	EXAMPLE	RESULT
ADDITION	+	Adds one value to another	10+5	15
SUBTRACTION	-	Subtracts one value from another	10-5	5
DIVISION	/	Divides two values	10/5	2
MULTIPLICATION	*	Multiplies two values	10*5	50
MODULUS	%	Divides two values and returns the remainder	10%3	1

Shorthand Operators

NAME	SHORTHAND OPERATOR	MEANING
Assignment	x = y	x = y
Addition Assignment	x += y	x = x + y
Subtraction Assignment	x -= y	x = x – y
Multiplication Assignment	x *= y	x = x * y
Division Assignment	x /= y	x = x / y
Remainder Assignment	x %= y	x = x % y

Relational Operators

Operator	Description
>	Greater than
>=	Greater than or equal
<	Less than
<=	Less than or equal
==	Equal
===	Strict Equal
!=	Not equal

All the relational operators will return Boolean (True or False)

Logical Operators

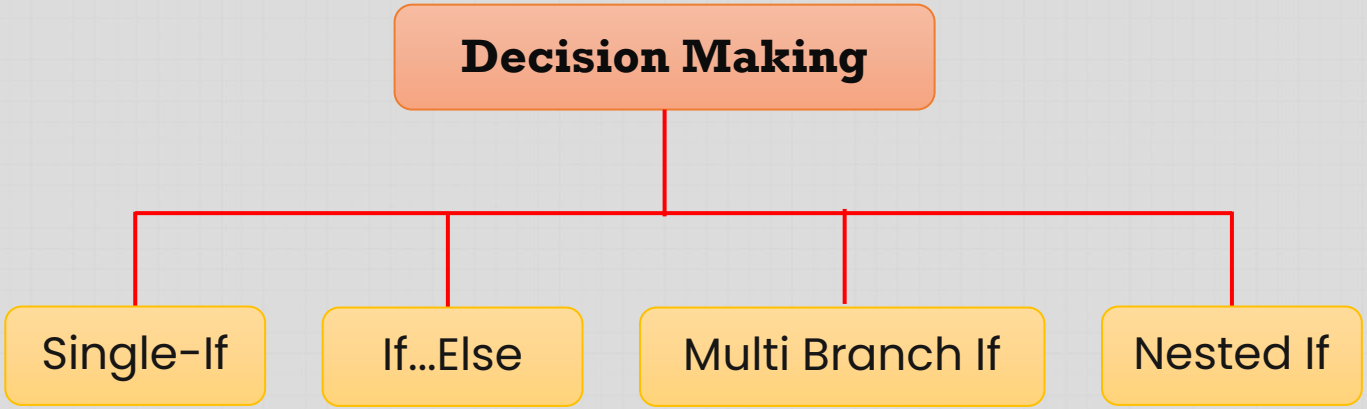
OPERATOR	DESCRIPTION
&&	Logical AND
||	Logical OR
!	Logical NOT

All the logical operators will return Boolean (True or False)

Decision Makings

If Statements

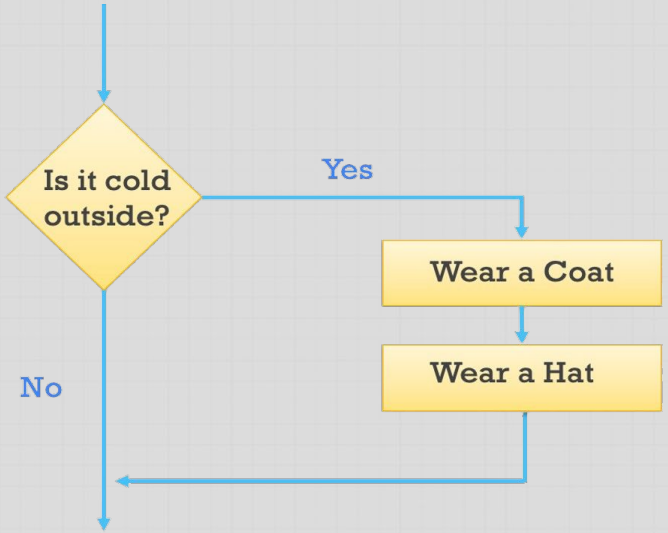
- Used for making decisions based on specified criteria



Single If

- The if statement evaluates a condition
- If the condition evaluates to true, any statements in the subsequent code block are executed

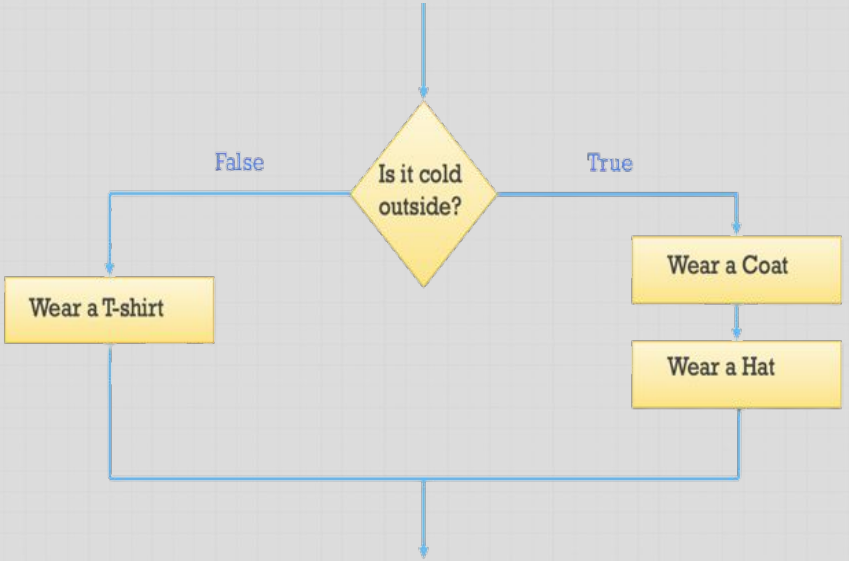
```
if(Condition){
    Statements
}
```



If...Else

- The if...else statement checks a condition
- If it resolves to true the first code block is executed
- If the condition resolves to false, the second code block is run instead

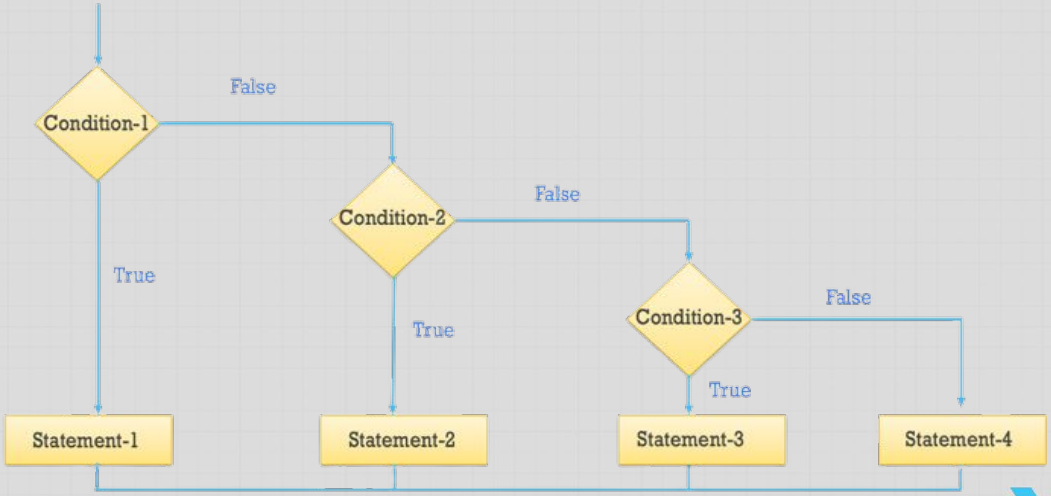
```
if(Condition){
    Statements
}else{
    Statements
}
```



Multi-branch If

- Multi-branch if statement can be used to create an else if clause
- It is used to make decision among several alternatives

```
if(Condition1){
    Statements
}else if(Condition2){
    Statements
}else{
    Statements
}
```



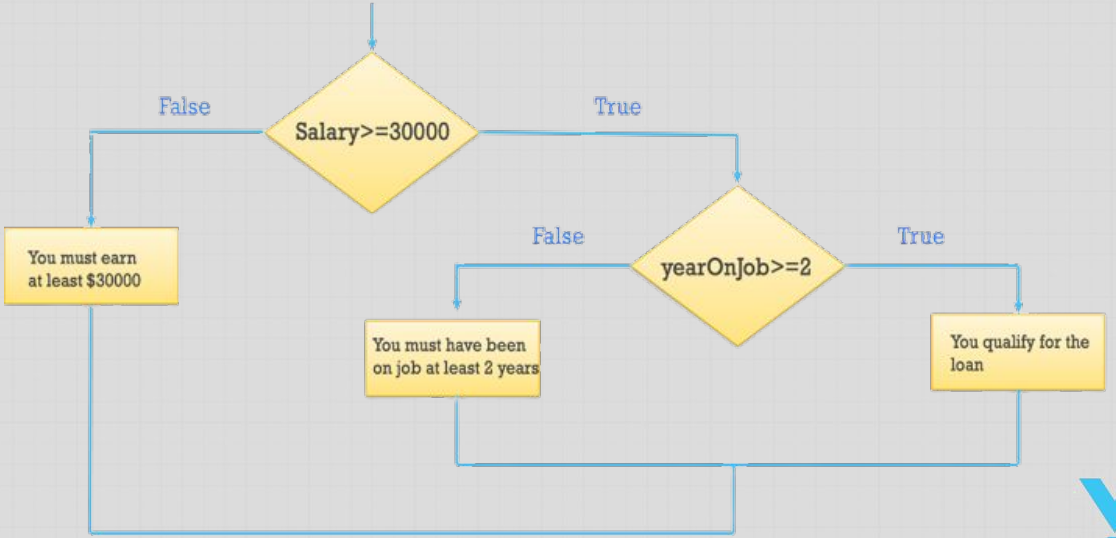
Multiple else if blocks can be given

Nested If

- Nested if statements can be used for creating a pre-condition
- It's used if one condition can be evaluated to several alternatives

```
if(Condition){
    if(Condition){
        Statements
    }
}
```

Outer and Inner If statements can be any type of if statement



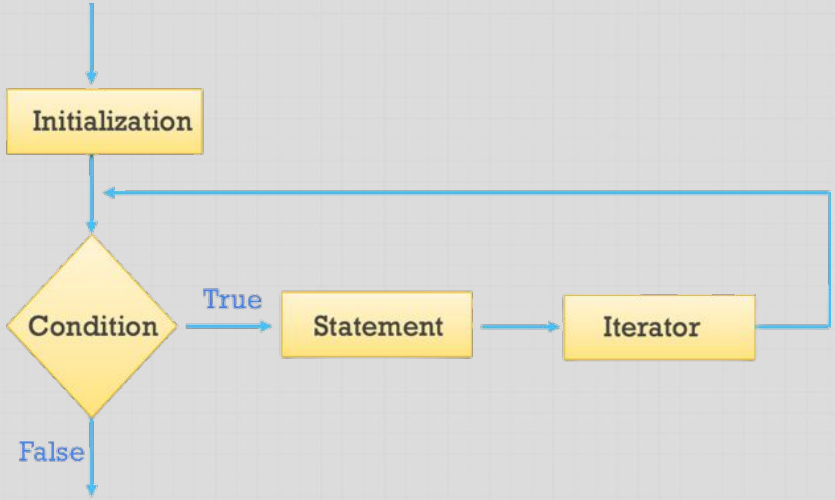
Loops

Loops

- Used for repeating a set of statements
- There are two types of loops:
  - For Loop
  - While Loop
  - Do-While Loop

For Loop

- Runs the given code a specific number of times
- Initialization is the starting point of the loop
- Condition is the ending point of the loop
- Iterator is responsible for making the condition false



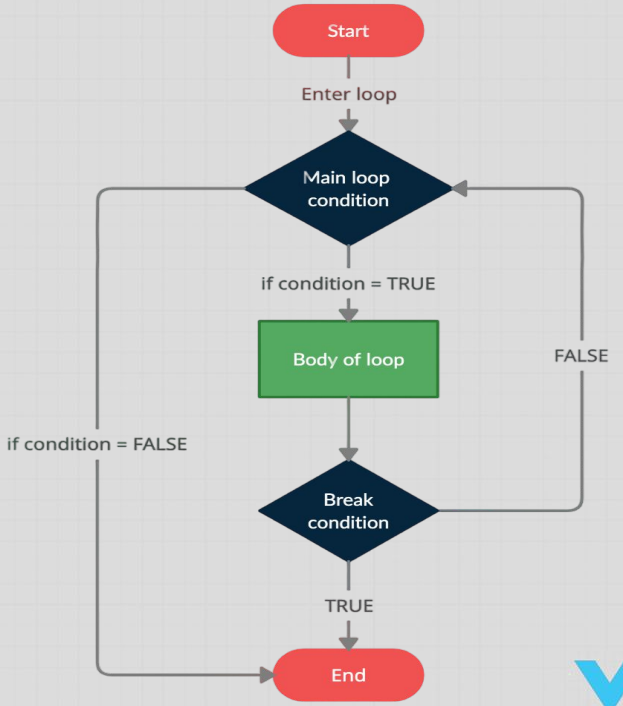
For Loop Syntax

```
for(initialization; condition; iterator){
        Statements
}
```

- The initialization expression initializes the loop
- When the condition expression evaluates to false, loop stops running
- The Iteration gets executed after each iteration through the loop

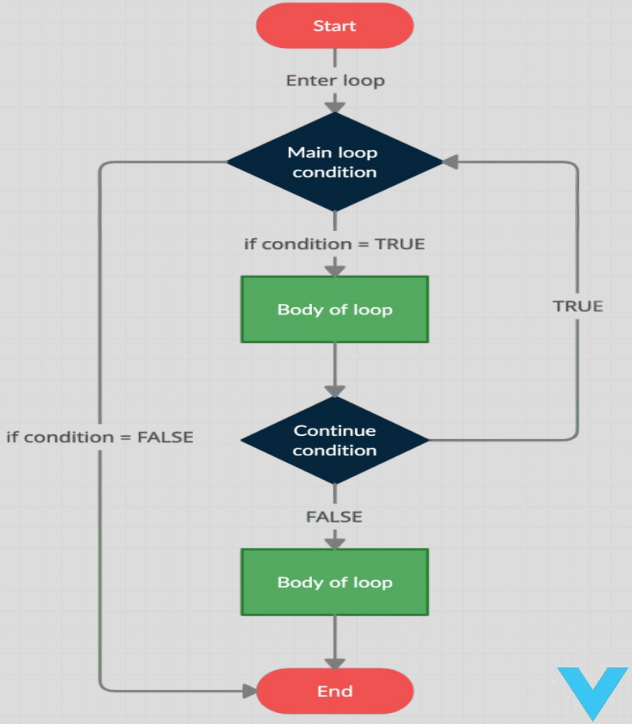
Break Statement

- Causes the termination of the loop
- Tells the interpreter to go on to the next statement of code outside of the loop



Continue Statement

- Skips the current iteration of the loop
- Tells the interpreter to jump to the next iteration



Perplexity AI

Perplexity AI: Empowering Programmers

- Hybrid AI Tool: Combines Search Engine and Prompting Capabilities.
- Enhanced Search:
  - Quick access to coding solutions.
  - Efficient research and fact-checking.
- Troubleshooting Support:
  - Debugging assistance and code snippets.
  - Clarifies complex programming concepts.