Variables and Keywords in JavaScript

🔷 What are Variables?

- A **variable** is like a container used to store data values.
- Example: You can store a number, text, or object inside a variable.

🔷 Ways to Declare Variables

1. **var** → Variable can be re-declared & updated. A global scope variable.
2. **let** → cannot be re-declared but can be updated. A block scope variable.
3. **const** → Variable cannot be re-declared or updated. A block scope variable

---

🔷 Rules for Naming Variables

Rules for Naming Variables

- Can contain **letters, digits, _ , $** (but not space and other special symbols)
- Must not start with a **digit**.
- Case-sensitive (`Name` and `name` are different)
- Use meaningful names (`studentName` not `x`)
- Reserved words can't be used.

Keywords

Keywords are **reserved words** in JavaScript that have a **special meaning** and **cannot be used as variable names**, function names, or identifiers

var	let	const	if
else	switch	case	default
break	continue	for	while
do	function	return	class
extends	super	try	catch
throw	finally	import	export
this	new	delete	in
instanceof	async	await	typeof
void	yield	true	false
null	undefined	NaN	with
debugger			

Category	Keyword	Description
Variable Declaration	`var`	Declares a variable (function-scoped, old method).
	`let`	Declares a block-scoped variable (modern).
	`const`	Declares a block-scoped constant (cannot be reassigned).
Control Statements	`if`	Executes a block if condition is true.
	`else`	Executes a block if condition is false.
	`switch`	Selects one block of code to execute among many.
	`case`	Defines a case in a `switch` statement.
	`default`	Executes if no `case` matches.
	`break`	Exits from a loop or switch.
	`continue`	Skips current iteration in a loop.
Loops	`for`	Runs a loop for a fixed number of times.
	`while`	Runs while a condition is true.
	`do`	Runs at least once, then checks condition.
Functions & Classes	`function`	Declares a function.
	`return`	Returns a value from a function.
	`class`	Declares a class (object-oriented).
	`extends`	Inherits from another class.
	`super`	Calls the parent class constructor or method.
Error Handling	`try`	Defines a block to test for errors.
	`catch`	Defines a block to handle errors.
	`throw`	Manually throws an error.
	`finally`	Executes code after `try...catch` (always runs).
Modules (ES6+)	`import`	Imports modules, functions, or classes.
	`export`	Exports modules, functions, or classes.
Objects & Properties	`this`	Refers to the current object.
	`new`	Creates an instance of an object or class.
	`delete`	Deletes a property from an object.
	`in`	Checks if a property exists in an object.
	`instanceof`	Checks if an object is an instance of a class.
Asynchronous Programming	`async`	Declares an asynchronous function.
	`await`	Waits for a promise to resolve inside async functions.
Operators & Type Checking	`typeof`	Returns the type of a variable.
	`void`	Evaluates an expression but returns `undefined`.

Generators	`yield`	Pauses and resumes a generator function.
Boolean & Values	`true`	Boolean value representing truth.
	`false`	Boolean value representing falsehood.
	`null`	Represents an empty or non-existent value.
	`undefined`	Represents an uninitialized variable.
	`NaN`	Stands for “Not a Number”.
Deprecated / Rarely Used	`with`	Extends the scope chain (deprecated).
	`debugger`	Stops execution and calls the debugger.

👉 **Mini Assignment (Practice)**:

1. Declare a variable `firstName` and store your name.
2. Declare a variable `birthYear` and store your birth year.
3. Declare a constant `PI` with value `3.14159`.
4. Print all three using `console.log()`.