const:

Static Constant: these are recommended to write in capital letter

Ex: const PI = 3.14;

Dynamic Constant: these values can be changed but not directly, these are recommended to write in camel case

Ex: let x= 10; let y=20;

const z = x+y; (the value can be changed but not directly)

<script type="text/javascript">

        let x= parseInt(prompt("Enter first number"));

        let y= parseInt(prompt("Enter second number"));

        const z = x+y;

        document.write(z);

</script>

Note: the js engine in chrome and Microsoft edge are called v8 engines. This js v8 engine has the components they are:

Note: Java script is a single thread engine.

1. Parser: Converts raw JavaScript source code into tokens (lexical analysis). Checks syntax and builds an Abstract Syntax Tree (AST)
2. Semantic Analyzer: Performs semantic checks on the AST:

Scope Resolution

Variable/function declaration checks

Detects semantic errors (e.g., illegal reassignments)

Prepares data structures for execution (scope chains, environment records).

1. Byte Code Generator (Interpreter Compiler): Converts the AST into bytecode, an intermediate low-level instruction set optimized for interpretation.
2. Interpreter (e.g., Ignition in V8):

Executes the bytecode by running machine code routines that implement the bytecode instructions.

The interpreter itself is compiled to native machine code and runs on the CPU.

Enables fast startup and immediate execution of code.

1. Profiler / Hot Code Detector:Monitors the running code to identify “hot” or frequently executed sections of bytecode suitable for optimization.
2. JIT Compiler (e.g., Turbofan in V8): Takes hot bytecode sections and compiles them into optimized native machine code for the CPU. Produces highly efficient machine code that runs faster than bytecode interpretation.
3. Runtime / Built-in-APIs: Provides implementations for JavaScript standard library features (like Math, Date, Promise). Includes the event loop integration and interaction with the environment (browser or Node.js)
4. Garbage Collector (GC): Automatically manages memory allocation and reclamation. Frees unused objects to avoid memory leaks.
5. Optimizer / Deoptimizer: Optimizes compiled machine code based on runtime information. Can deoptimize and revert to bytecode interpretation if assumptions made during optimization are invalidated.

typeof(): typeof lets you what the variable is holding

Variable hoisting: variable hoisting in let cannot be accessed but this is not meant that hoisting is not occurred. Variable hoisting puts the let variable in dead time zone

Confirm dialog box: This dialog box takes only Boolean input from the user.

**Conditional Statement:**

1. If else statement:

if(condition/Boolean expression)

Statement

else

Statement

Note: When there are multiple statements to be executed we need to mention the block with curly brackets (mandatory)

1. If else if statement:

if(condition)

Statement

else if(condition)

Statement

else

statement

**Looping Statement:**

1. while: as long as the given condition is evaluated to true the statements inside the while loop will be executed repeatedly. Infinite loops will execute infinite times as the condition is not becomes false

<script>

        document.write("<br>Hello");

        let i = 0;

        while(i<4){

            document.write("start");

            i++;

        }

        document.write("end");

</script>