**You don’t Know JS - UP & GOING**

**Statement:** a group of words, numbers, and operators that performs a specific task

**Variables:**  holds values

E**xpression:**  is any reference to a variable or value, or a set of variable(s) and value(s) combined with operators.

* literal value expression : 2
* variable expression:b
* arithmetic expression:b\*2
* assignment expression a= b\*2
* call expression alert(a)

**Interpreter**/**compiler**: is used to translate the code you write into commands a computer can understand.

**Interpreting the code**: Translation of commands. It is typically done from top to bottom, line by line, every time the program is run.

**Compiling the code**: translation is done ahead of time

The **JavaScript engine** actually **compiles** the program on the fly and then immediately **runs** the compiled code.

**Operators:**  are how we perform actions on variables and values. = ,\*

**Common operators**:

* Assignment: =
* Math: -+ (addition), - (subtraction), \* (multiplication), and / (division)
* Compound assignment: +=, -=, \*=, and /=
* Increment/decrement: ++, --
* Object property access: console.log()
* Equality: == (loose-equals), === (strict-equals), != (loose not-equals), !== (strict not-equals), as in a == b.
* Comparison: < (less than), > (greater than), <= (less than or loose-equals), >= (greater than or loose-equals), as in a <= b. See “Values & Types” on page 10 and Chapter 2. Logical && (and), || (or), as in a || b that selects either a or b.

**Primitive values**

* When you need to do math, you want a number.
* When you need to print a value on the screen, you need a string (one or more characters, words, or sentences).
* When you need to make a decision in your program, you need a Boolean (true or false).

**Literals** : Values that are included directly in the source code.

**Coercion**: If you have a number but need to print it on the screen, you need to convert the value to a string, and in JavaScript this conversion is called “coercion.” The first console.log(..) command has to implicitly coerce that number value to a string to print it out.

**Static typing (type enforcement):** is typically cited as a benefit for program correctness by preventing unintended value conversions.

**Weak typing(dynamic typing):**  allows a variable to hold any type of value at any time

**Constants** : when you declare a variable with a value and intend for that value to not change throughout the program. Another common usage of variables is for centralizing value setting.

**Block:**  group a series of statements together

**Conditionals** – if , switch(shorthand for a series of if..else statements), loops etc

**Loop:** A loop includes the test condition as well as a block (typically as { .. }). Each time the loop block executes, that’s called an **iteration**

**Function:** is generally a named section of code that can be “called” by name, and the code inside it will be run each time.

**Scope:**  is basically a collection of variables as well as the rules for how those variables are accessed by name. Only code inside that function can access that function’s scoped variables.

**Lexical scope:**  code in one scope can access variables of either that scope or any scope outside of it.