**Language Map for JavaScript**

|  |  |
| --- | --- |
| **Variable Declaration**  *Is this language strongly typed or dynamically typed? Provide at least three examples (with different data types or keywords) of how variables are declared in this language.* | **JavaScript is a dynamically-typed language. You can declare variables, functions, and objects without specifying a type first.**   1. **var x = 5;** 2. **let x =5;** 3. **x = 5;** |
| **Data Types**  *List all of the data types (and ranges) supported by this language.* | 1. **number – JS only has one type of number. Can be written with OR without decimals.** 2. **bigint – can safely represent large numbers above 9007199254740991, CANNOT have decimals, arithmetic between BigInt and Number is not allowed, add *n*  to end of a number to turn into a BigInt.** 3. **string – you can use with single or double quotes.** 4. **boolean -true or false.** 5. **null – an object within JS that has no value – considered a bug NOT a feature.** 6. **undefined – variable without a value.** 7. **symbol – object whose constructor returns a symbol primitive that is guaranteed to be unique. Allows you to associate a unique, hidden symbol within the object data to aid with debugging.** 8. **object – written with curly braces {}, properties are written as *name:value pairs*, separated by commas.** |
| **Selection Structures**  *Provide examples of all selection structures supported by this language (if, if else, etc.)* ***Don’t just list them, show code samples of how each would look in a real program.*** | |  |  | | --- | --- | | **if** | **var number = 44;**  **if ((number % 2) != 0 {**  **document.write(number + “ is an odd number”);**  **}** | | **if else** | **var number = 44;**  **if ((number % 2) !=0) {**  **document.write(number + “ is an odd number”);**  **}**  **else {**  **document.write(number + “ is an even number”);**  **}** | | **switch** | **var letter = “I”;**  **switch(letter) {**  **default: document.write(“consonant”);**  **break;**  **case “A”: document.write(“A is a   vowel”);**  **break;**  **case “E”: document.write(“E is a   vowel”);**  **break;**  **case “I”: document.write(“I is a   vowel”);**  **break;**  **case “O”: document.write(“O is a   vowel”);**  **break;**  **case “U”: document.write(“U is a   vowel”);**  **break;**  **}** | |
| **Repetition Structures**  *Provide examples of all repetition structures supported by this language (loops, etc.)* ***Don’t just list them, show code samples of how each would look in a real program.*** | |  |  | | --- | --- | | **for** | **for (let i = 0; i < 5; i++) {**  **text += “The number is “ + i +  “<br>”;**  **}** | | **for/in** | **const person = {fname: “John”, lname: “Doe, age: 25};**  **let text = “”;**  **for (let x in person) {  text += person[x];**  **}** | | **for/of** | **const cars = [“BMW”, “Volvo”, “Mini”];  let text = “”;**  **for (let x of cars) {  text += x; }** | | **while** | **while (i < 10) {  text += “The number is “ + i;  i++;**  **}** | | **do/while** | **do {  text += “The number is “ + i;  i++;**  **}**  **while (i <10);** | |
| **Arrays**  *If this language supports arrays, provide at least two examples of creating an array with a primitive or String data types (e.g. float, int, String, etc.)* | **let arr = new Array();**  **let arr = [];** |
| **Data Structures**  *If this language provides a standard set of data structures, provide a list of the data structures and their Big-Oh complexity.* | 1. **Arrays – collection of items stored at contiguous memory locations.** 2. **Objects (hash tables) – collection of key-value pairs.** 3. **Stacks – store information in list form (LIFO).** 4. **Queues – stores information similarly to a stack but instead follows FIFO.** 5. **Linked lists – stores information in a list but every value is linked to another.**  * **Singly linked lists – only contains pointer to next node.** * **Doubly linked lists – contains pointer to next node AND previous.**  1. **Trees – stores information by linking nodes in a parent/child relationship.**  * **Binary trees – each node has a maximum of two children.**  1. **Heaps – stores information similarly to a tree but varies on the two following types.**  * **MaxHeaps – parent nodes are always greater than its children.** * **MinHeaps – parent nodes are always smaller than its children.**  1. **Graphs – stores information by grouping nodes together and placing certain connections between them. Do not have roots, leaf nodes, head, or tail. No implicit parent-child relationship between nodes.**  * **Undirected graphs – no implicit direction in the connections between nodes (bidirectional). A 🡨🡪 B 🡨🡪 C 🡨🡪 D** * **Directed graphs – implied direction between node connections (unidirectional). A 🡪 B 🡪 C 🡪 D** * **Weighted graph – connections between nodes have an assigned value (weight). It is information about the connection itself, NOT the nodes.** * **Unweighted graphs – connections between nodes have NO assigned value (weight).** |
| **Objects**  *If this language support object-orientation, provide an example of how you would write a simple object with a default constructor and then how you would instantiate it.* | **const person = {firstName:“John”, lastName:“Doe”, age:50; eyeColor:”blue”};** |
| **Runtime Environment**  *What runtime environment does this language compile to? For example, Java compiles to the Java Virtual Machine.*  *Do other languages also compile to this runtime?* | 1. **Browser Runtime Environment – where your JS application is executed within a browser, and it uses the methods built in to the browser to perform its desired actions.** 2. **Node Runtime Environment – allows your JS application to be executed without a browser.**  * **CoffeeScript, Dart, TypeScript, Clojure Script** |
| **Libraries/Frameworks**  *What are the popular libraries or frameworks used by programmers for this language? List at least three (3) and describe what they are used for..* | 1. **React JS – most popular framework (over 40% of devs use it), used to build highly- responsive user interfaces. Declarative, component-based (can reuse components to create complex UIs).** 2. **jQuery – simplifies interaction with the DOM (Document Object Model) tree and helps with tree navigation.** 3. **Express – typically used for backend development. Can be used with Node.js runtime. Provides an easy way to manage routing, middleware packages, and integrate plugins on server-side code.** |
| **Domains**  *What industries or domains use this programming language? Provide specific examples of companies that use this language and what they use it for****. E.g. Company X uses C# for its line of business applications.*** | **JS is most often used for client-side and server-side web development, mobile development, game development, front-end development, and back-end development.**   * **eBay – uses JS for front-end and back-end development of their e-commerce website.** * **Microsoft – front-end & back-end development for their web browser, Edge.** * **Netflix - front-end, back-end, and web-based development of their media streaming apps.** |