**7.7 Summary**

**9:00 – Introduction to JavaScript**

* JavaScript has no connection to Java, despite the similar name.
* ECMAScript is another name for JavaScript. It was more commonly used in the past.
* JavaScript is the most popular language today, especially for web development.
* Stack Overflow has become less popular due to the rise of AI tools.
* JavaScript is popular because it is the only language natively understood by web browsers.
* Node.js transformed JavaScript from a browser-only scripting language into a general-purpose programming language.
* The alert() function works only in browser environments.
* We can run JavaScript in the terminal using:
* node <file-name>
* The Event Loop exists in browsers but works differently (or is absent) in Node.js.
* TypeScript is a superset of JavaScript that adds static typing and additional features.

**10:00 – Variables and Types**

* JavaScript data types include: number, string, boolean, and undefined.
* JavaScript is loosely typed (a.k.a. dynamically typed), meaning:
* let x; // x can be reassigned to any type later
* var (from ES3) vs. let/const (from ES6):
  + var has function scope, while let and const have block scope.
  + var declarations are hoisted to the top of their scope, but let/const are not initialized until their definition.
  + Example:
  + console.log(a); // undefined
  + var a = 5;
  + console.log(b); // ReferenceError
  + let b = 5;
* Recommendation: Use only let or const. Avoid var.
* null is considered an object, but it represents no value.

**11:00 – Objects and Miscellaneous**

* Object declaration example:
* let obj = {
* key1: value1,
* key2: value2,
* key3: value3
* };
* Object values can be: primitives, arrays, functions, or other objects.
* Variable names (identifiers) must start with a letter, underscore \_, or dollar sign $.
* Keep in mind memory efficiency when creating large objects or arrays.

**Review Questions**

* What are identifiers?  
  Variable/function names used to identify values in the code.
* What happens when you concatenate a number and a string?  
  The number is converted to a string:
* console.log(1 + "2"); // "12"
* console.log(1 == "1")  
  → true (loose equality allows type coercion)
* console.log(1 === "1")  
  → false (strict equality compares both value and type)
* What is NaN and when does it occur?  
  NaN (Not-a-Number) occurs when a numeric operation fails:
* console.log("abc" / 2); // NaN
* What is a falsy expression?  
  Values that are treated as false in a boolean context:
  + false, 0, "", null, undefined, NaN
* What is the purpose of startPosition || 0?  
  Ensures a fallback to 0 if startPosition is undefined, null, or falsy.
* What does employees.push(...candidates) do?  
  Adds all elements of the candidates array to the employees array using the spread operator.

Let me know if you want this formatted into a PDF, slide deck, or one-pager for sharing or studying.

| **Method** | **Action** |
| --- | --- |
| push(x) | Adds x to the **end** |
| pop() | Removes the **last** element |
| unshift(x) | Adds x to the **beginning** |
| shift() | Removes the **first** element |
| **13.7- Solving problems** |  |
|  |  |

8.8- Functions

Function overload- same function with different arguments. Not available in JavaScript.

10:20- Advanced functions

Call back function

Function which call other functions. The called function will be an argument of the first function.

|  |  |
| --- | --- |
|  |  |

Ex3- Is it the solution?

Debugging- To see variables I must add them to watchlist?

**23.7**

תרגיל עצים

.hasOwnProperty- סוקר את המפתחות

this- No access. Work with another pointer.

{}- hashmap שהמפתחות מחרוזות

ניתן ליצור מערך אובייקטים

בדיקות jest

התקנה

ייצוא מה שבודקים

ליצור קובץ בדיקה

| Method | Description | Example |

| --------------------------------- | ------------------------------------- | ----------------------------------- |

| `expect(value).toBe(expected)` | Tests strict equality (`===`) | `expect(5).toBe(5);` |

| `expect(value).toEqual(expected)` | Tests deep equality (objects, arrays) | `expect([1,2]).toEqual([1,2]);` |

| `expect(value).toBeDefined()` | Checks value is not `undefined` | `expect(someVar).toBeDefined();` |

| `expect(value).toBeTruthy()` | Checks value coerces to true | `expect(true).toBeTruthy();` |

| `expect(value).toBeFalsy()` | Checks value coerces to false | `expect(0).toBeFalsy();` |

| `expect(value).toContain(item)` | Checks array or string contains item | `expect(['a','b']).toContain('a');` |

| `expect(fn).toThrow()` | Expects function to throw an error | `expect(() => myFunc()).toThrow();` |

git init

git clone

git add .

git commit -m <str>

git push (default: current branch)(avalible: origin master)

git status -> Did you add?

git remote -v -> Does local code connect to GitHub?

git remote add origin <YOUR\_URL> -> adding an URL

branches

git branch

git checkout -b branch-name

git push origin branch-name\_

git checkout branch-name

git merge <branch-name>

git ls -a

git log

git push origin master

git diff

git remote -v (the url's repo)

git checkout -b server-backup (opening a new branch- don't work on master)

git push origin

git pull origin master

set methods

| Method / Operation | Description | Example | Time Complexity |

| -------------------------- | ---------------------- | ------------------------------------ | --------------- |

| `new Set(iterable)` | Create a new Set | `new Set([1, 2, 3])` | O(n) |

| `.add(value)` | Add a value to the set | `mySet.add(5)` | O(1) |

| `.delete(value)` | Remove a value | `mySet.delete(2)` | O(1) |

| `.has(value)` | Check if value exists | `mySet.has(2)` | O(1) |

| `.clear()` | Remove all values | `mySet.clear()` | O(1) |

| `.size` | Number of items | `mySet.size` | O(1) |

| `.forEach(callback)` | Iterate through values | `mySet.forEach(x => console.log(x))` | O(n) |

| `...spread` / `Array.from` | Convert Set to array | `[...mySet]`, `Array.from(mySet)` | O(n) |