CADT IDT

# ***W1** PRACTICE*

# *From C++ to JS*

## *At the end of this practice, you can*

- ✓ Run JS code
- ✓ Create **variables** and **constants**
- ✓ Call and define **functions**
- ✓ Use JS **loops** and **conditions**
- ✓ Manipulate **arrays**, **objects**, **strings**, **Boolean** and **numbers**

## *Get ready before this practice!*

- ✓ **Read** the following documents to understand JS syntax:

  https://cstart.mines.edu/web/Day2/2-JavaScriptBasicSyntax.pdf
  https://www.integral-domain.org/lwilliams/mis462/JavaScript.pdf

  You can also go further with the following books:
  https://www.gurukultti.org/admin/notice/javascript.pdf
  https://www.w3schools.com/js/default.asp

- ✓ **Complete the quiz** (*you can re-do it until you have 100% score*)

## *How to submit this practice?*

- ✓ **Complete** this document
- ✓ Once finished, join this document to the MS Team assignment and **turn it in**

# *3 WAYS TO **RUN JS CODE***

## *For beginners*

To start with, you can just connect to an **online JavaScript editor**, such as this one:
https://playcode.io/javascript

## *For front-end ninjas*

Chrome or any other **Web Browser** can execute JavaScript code while loading HTML

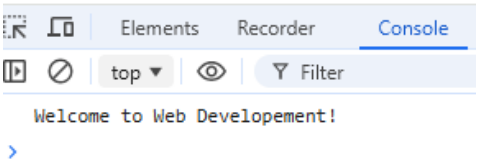
Just create a simple `index.html` file, that links to a `index.js` file:

```
<!DOCTYPE html>
<html>
<head>
    <title>Let s run JS on a Browser</title>
    <script src='index.js'></script>
</head>
<body>
</body>
</html>
```

Then just write some JS code, as example here, we print a message on the Browser console

```
// Example of JS code, printing on console
const courseName = "Web Developement";
console.log("Welcome to " + courseName + "!");
```

Finally open your index.html on a browser and check the console view




## *For back-end gurus*

Node.js is also able to **execute JavaScript** code **outside a web browser.**
You will need first to install Node JS on your computer.

You can then just open a terminal on the folder containing your index.js file and run

```
node ./index.js
```

# PART 1 - UNDERSTAND **JS SYNTAX**

*Note: you can use the [C++ to JS converter] to compare C++ and JS syntax.*


## EXERCISE 1- TYPES, OUTPUTS

Analyze the differences between the provided C++ and JavaScript code.

| C++ | JS |
|---|---|
| `#include <iostream>`<br>`using namespace std;`<br><br>`int main() {`<br>`  const int num = 5;`<br>`  for (int i = 0; i < num; i++) {`<br>`  cout << i << " ";`<br>`  }`<br>`  return 0;`<br>`}` | `const num = 5;`<br>`for (let i = 0; i < num; i++) {`<br>`  console.log(i);`<br>`}` |

**Q1 -** What does the **const** key word mean in JS code?

const define a variable that can not change it value , a constant variable

**Q2 -** Why is it necessary **to specify the type** of variables in C++ but not in JavaScript?

C++ is a static programming language and js is a dynamically programming language

**Q3-** How to **print in the console** in JS?

console.log()

**Q4-** Is there any difference in the **loop syntax** between C++ and JS?

There are some that in fact the same like for loop , while loop , do while and for of (different syntax) but in js there is an extra loop : for in

# EXERCISE 2 - LOOPS, FUNCTIONS

**C++**

```cpp
#include <iostream>
using namespace std;

int calculateSum(int array[], int size) {
    int sum = 0;

    for (int i = 0; i < size; i++) {
      // Add here the calculation logic
       sum += arr[i]
    }
    return sum;
}

int main() {
    int arr[] = {1, 2, 3, 4, 5};
    cout << calculateSum(arr, 5);
    return 0;
}
```

**JS**

```js
function calculateSum(array) {
    let sum = 0;

    for (let i = 0; i < array.length; i++) {
       // Add here the calculation logic
       Sum += arr[i]
    }
    return sum;
}

let arr = [1, 2, 3, 4, 5];
console.log(calculateSum(arr));
```

**Q1 -** Complete the given codes (`see comments`) to compute the sum of all elements in an array

**Q2 –** Why the function calculateSum in JS code **does not have the size** parameter?

Array in js has a build in method .length that will automatically return its size

# EXERCISE 3 - CONDITIONS, EQUALITY

**JS**

```
function myFunction(min, max) {
  var result = "";
  for (let number = min; number <= max; number++) {
    if (number % 2 === 0) {
      result += number + " - ";
    }
  }
  return result;
}
```

**Q1 –** Look at the above code

- Highlight all **variables in blue**
- Underline all **loops in red**
- Highlight all **conditions in green**

**Q2 –** What is the significance of the modulo operator % in these programs?

| |
|---|
| It will return to the remainder of the division |

**Q3 –** What is the difference between === and == in JS? *Highlight the right answer*

| | |
|---|---|
| 4 == 9 | TRUE / FALSE |
| 4 == 4 | TRUE / FALSE |
| 4 == "4" | TRUE / FALSE |
| 4 === "4" | TRUE / FALSE |

**Q4 –** What will this code will print on console?

```
console.log(myFunction(9, 14))
```

| |
|---|
| 10 - 12 - 14 |

**Q5 –** What will this code will print on console?

```
console.log(myFunction(7, 3))
```

| |
|---|
| empty string |

# *EXERCISE 4 – MEMORY ALLOCATION*

Both codes are performing the same job:

**C++**

```cpp
#include <iostream>
using namespace std;

int main() {
    int size = 5;
    int* arr = new int[size];
    for (int i = 0; i < size; i++) {
        arr[i] = i * 2;
    }

    for (int i = 0; i < size; i++) {
        cout << arr[i] << " ";
    }
    delete[] arr;
    return 0;
}
```

**JS**

```js
let size = 5;
let arr = [];
for (let i = 0; i < size; i++) {
    arr[i] = i * 2;
}

for (let i = 0; i < size; i++) {
    console.log(arr[i]);
}
```

**Q1 –** In both codes, are we using a **static** or a **dynamic** array? Explain why…

Dynamic because in C++ we use new to allocate an array at runtime and keyword delete[] to free the allocated array and in js array automatically grow and create at run time

**Q2 –** Explain why JavaScript **does not** need explicit **memory allocation** or **deallocation**, as C++ need it

JS handle memory and free memory automatically , while in C++ we have to manage and handle memory by our self

# *PART 2 - CODE **JS CHALLENGES***


## Good job!

Now you should know the [basic syntax of JavaScript!]
Let's solve some problem now.

Each challenge is structured the same way:

- **Goal** What the function shall do
- **Inputs**: the function parameters
- **Output** the function return

As example, for the challenge 1, you will provide the following function:

```
function challenge1(width, height) {
    let rectangleString = '';
    // Your code
    return rectangleString;
}
```

| CHALLENGE 1 | EASY |
|---|---|
| Draw a rectangle in the console using stars | |

| INPUT | OUTPUT |
|---|---|
| width 3<br>height 4 | ***<br>***<br>***<br>*** |
| width 5<br>height 2 | *****<br>***** |
| width 5<br>height -2 | |

| CHALLENGE 2 | MEDIUM |
|---|---|
| Reverse an array | |

| INPUT | OUTPUT |
|---|---|
| array [14,15,16,20] | [20,16,15,14] |
| array [5,4,3,2,1] | [1,2,3,4,5] |
| array [] | [] |

Any help on arrays with JavaScript? Check here.

| CHALLENGE 3 | MEDIUM |
|---|---|
| Calculate the average grade of a list of students. | |

| INPUT | OUTPUT |
|---|---|
| array [85, 90, 78, 92] | 86.25 |
| array [10,20,30] | 20 |
| array [] | 0 |

| CHALLENGE 4 | MEDIUM |
|---|---|
| Write a function to count how many times a character appears in a string. | |

| INPUT | OUTPUT |
|---|---|
| `text  "hello world"`<br>`char = 'o'` | `2` |
| `text  "aaa bbb a"`<br>`char = 'a'` | `4` |
| `text  "abc"`<br>`char = 'd'` | `0` |

| CHALLENGE 5 | HARD |
|---|---|
| Count the number of words in a sentence | |

| INPUT | OUTPUT |
|---|---|
| `text  "hello world"` | `2` |
| `text  "this is the best day"` | `5` |
| `text  "a bb ccc ddddddd e"` | `5` |

| CHALLENGE 6 | HARD |
|---|---|
| Simulate a voting system for three candidates (A / B/ C).<br>Count votes and declare a winner | |

| INPUT | OUTPUT |
|---|---|
| `votes  ['A', 'B', 'A', 'C', 'A']` | `A is the winner` |
| `votes  ['A', 'B', 'B', 'C', 'A']` | `A and B are both winners` |
| `votes  []` | `There is not vote yet` |