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

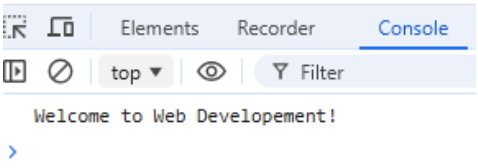
Just create a simple `index.html` file, that links to a `index.js` file:

```
<!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

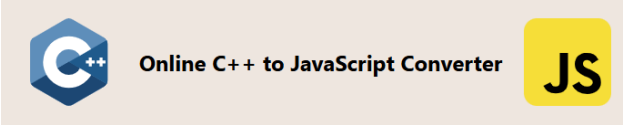
Node.js is also able to **execute JavaScript** code **outside a web browser.**
You will need first to install Node JS on your computer.

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>`
`using namespace std;`
 	
`int main() {`
`  const int num = 5;`
`  for (int i = 0; i < num; i++) {`
`  cout << i << " ";`
`  }`
`  return 0;`
`}`
	`const num = 5;`
	`for (let i = 0; i < num; i++) {`
	`  console.log(i);`
	`}`

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

Const is a key word that we use to declare a variable that can not be reassigned a new value.

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

Because **C++ is statically typed,** this mean that the type of variable is know a compile time. We must declare the type of variable before we use it.
Whereas **JavaScript is dynamically typed,** this mean that the type of variable is check during runtime. We don't need to declare the type of variable.

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 is no different in the **loop syntax** between **C++ and JS**.

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 += array[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 += array[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

`sum += array[i];`

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

Because JavaScript array have a built-in **length** property that can calculate a size of an array.

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?

Modulo operator % in these programs is working to find the EVEN number in a length between min and max number. It's make sure the is even.

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))
```

(an empty line)

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

```javascript
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…

In both code, we using dynamic array. Because we in C++ The use of new int[size] allocate memory on the heap at runtime, making it a dynamic array. The size is used to determine the amount of memory to allocate at runtime. While JS arrays are inherently dynamic. They can grow or shrink as needed, and memory management is handled automatically by the JavaScript engine.

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

The fundamental reason JavaScript doesn't require explicit memory allocation and deallocation like C++ is because it uses automatic garbage collection.

*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 height 4	*** *** *** ***
width 5 height 2	***** *****
width 5 height -2	

Code :

```
function challenge1(width, height) {
    let rectangleString = '';
    for (let i = 0; i < height; i++) {
        for (let j = 0; j < width; j++) {
            rectangleString += '*';
        }
        if (i < height - 1) {
            rectangleString += '\n';
        }
    }
    return rectangleString;
}
// user input
console.log(challenge1(3, 4));
```

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.

Code :

```
function challenge2() {
    array.reverse();
    return array;
}
// user input
array = [14,15,16,20];
// array = [5, 4, 3, 2, 1];
// array = [];
console.log(challenge2());
```

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

Code :

```
function challenge3(array) {
    let sum = 0;
    let average = 0;

    if (array.length === 0) {
        return 0;
    }
    for (let i = 0; i < array.length; i++) {
        sum += array[i];
    }
    average = sum / array.length;
    return average;
}

//user input
array = [85, 90, 78, 92];

// array = [10,20,30];

// array = [];
console.log(challenge3(array));
```

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

INPUT	OUTPUT
text "hello world" char = 'o'	2
text "aaa bbb a" char = 'a'	4
text "abc" char = 'd'	0

Code :

```
function challenge4(text, char) {
    let count = 0;
    for (let i = 0; i < text.length; i++) {
        if (char === text[i]) {
            count = count + 1;
        }
    }
    return count;
}
// text = "hello world";
// char = 'o';

text = "aaa bbb a"
char = 'a'

// text = "abc"
// char = 'd'

console.log(challenge4(text, char));
```

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

Code :

```
function challenge5(text) {
    count = 0;
    for (let i = 0; i < text.length; i++) {
        if (text[i] === ' ') {
            count++;
        }
    }
    return count + 1;
}
// text = "hello world"

// text = "this is the best day"

text = "a bb ccc ddddddd e"


console.log(challenge5(text));
```

CHALLENGE 6	HARD
Simulate a voting system for three candidates (A / B/ C). 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

Code :

```
function challenge6(votes) {
    if (votes.length === 0) {
        return 'There is not vote yet';
    }

    let aCount = 0;
    let bCount = 0;
    let cCount = 0;

    for (let i = 0; i < votes.length; i++) {
        if (votes[i] === 'A') {
```

```
            aCount++;
        } else if (votes[i] === 'B') {
            bCount++;
        } else if (votes[i] === 'C') {
            cCount++;
        } else {
            console.log(`Invalid vote: ${votes[i]}`); // Handle invalid votes
        }
    }

    if (aCount === bCount && aCount > cCount) {
        return 'A and B are both winners.';
    } else if (aCount === cCount && aCount > bCount) {
        return 'A and C are both winners.';
    } else if (bCount === cCount && bCount > aCount) {
        return 'B and C are both winners.';
    } else if (aCount > bCount && aCount > cCount) {
        return 'A is the winner.';
    } else if (bCount > aCount && bCount > cCount) {
        return 'B is the winner.';
    } else if (cCount > bCount && cCount > aCount) {
        return 'C is the winner.';
    } else if (aCount === bCount && aCount === cCount && aCount !== 0) {
        return 'All three candidates are tied.';
    } else if (aCount === 0 && bCount === 0 && cCount === 0) {
        return 'There is not vote yet.';
    }
}

votes1 = ['A', 'B', 'A', 'C', 'A'];

votes2 = ['A', 'B', 'B', 'C', 'A'];

votes3 = [];

console.log(challenge6(votes1));
console.log(challenge6(votes2));
console.log(challenge6(votes3));
```