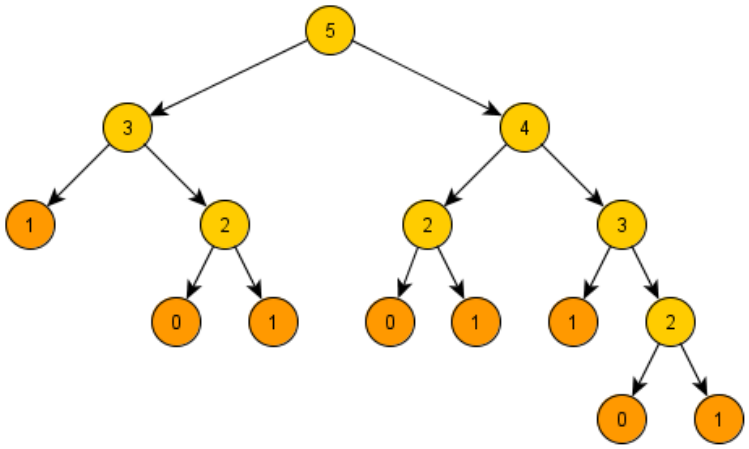
C/ Literat Azorín, 1 46702 – GANDIA
Tfno. 962829430 – Fax. 962829431
Codi: 46004221 – CIF: Q-9655627-I
46004221@edu.gva.es
http://iesmariaenriquez.com

- **ACTIVITY 1 – Even and odd function**
  - In this activity we will create a very simple function that returns the word "**even**" if the value we send is an even number. Otherwise, the word "**odd**" returns.
  - Write on a web page 500 random numbers from 1 to 10,000 and next to each number write if it is even or odd using the previous function.
- **ACTIVITY 2 – Function that draws a table**
  - Create a function in JavaScript that allows us to draw a table on a web page.
  - You will pass 2 parameters to the function, indicating the number of rows and columns. By default, the function will take 10 rows and 4 columns.
  - The table is created with a black border of 1 pixel between each cell, but a third parameter allows you to indicate the color (by default it will be black). The outer edge will measure 3 pixels and will always be the same color as the edge of the cells.
  - The table will occupy the entire width of the page.
  - Use the function to create a table with a black border of 10 rows and 4 columns
  - Use it again to create a table of 20 rows and 10 columns, with a black border.
  - Finally, draw 10 tables of 5 rows and 4 columns that have a green border.
- **ACTIVITY 3 – Function that allows you to know if a number is prime**
  - Create a function that lets you know if a number is prime or not.
  - A number is prime if it cannot be divided by another number (not counting the one or the number itself), giving a remainder of zero.
  - Take advantage of the function created to create a web page that writes the prime numbers from 1 to 1000.
- **ACTIVITY 4 – Sort words in reverse order**
  - Create an application that prompts the user for words continuously until a textless box is accepted or canceled.
  - Repeated words will be eliminated. The words entered will be shown in order in Spanish, but in reverse order (from Z to A)
- **ACTIVITY 5 – Create map with array repeats**
  - Create a function that receives an array of words.
  - The function will return a map that contains each word as a key and the value is the number of times that word appears in the array.
  - We will make a web page that reads words until the user cancels or leaves the box empty and we will show the number of repetitions of the words.
- **ACTIVITY 6 – Fibonacci recursive function**
  - The Fibonacci function is a classic of recursive programming. It is a function related to a succession of elements where the first 2 are 0 and 1, and the rest are the sum of their previous 2.
  - That is, the succession is 0,1,1,2,3,5,8,13,21,34,55,89,etc.
  - We would send a number to the function of which we want to know the value of Fibonacci and it would return that value to us. So if we pass the number 10, it would return 55.
  - The recursive solution is much easier, but it would be interesting to also do the non-recursive solution.

C/ Literat Azorín, 1 46702 – GANDIA
Tfno. 962829430 – Fax. 962829431
Codi: 46004221 – CIF: Q-9655627-I
46004221@edu.gva.es
http://iesmariaenriquez.com



- ❑ **ACTIVITY 7 – Function to detect palindromes**
  - Create a function to solve what was requested in practice 7 of UNIT 4.
  - The function receives a text and will return true if it is a palindrome and false if it is not.
  - Keep in mind that in order for palindromes to be considered well, punctuation marks (spaces, questions, commas, periods, etc.) are also ignored, tildes and umlauts are also ignored (the character **a** is considered the same as the character **á**) and is not distinguished between uppercase and lowercase letters.
- ❑ **ACTIVITY 8 – Function to detect anagrams**
  - Create a function to which a series of texts are passed (minimum 2) and detect if they are anagrams or not.
  - An anagram is a word that results from transposing the letters of another: for example STUDY and DUSTY.
  - The function returns true if all past words are anagrams of the same letters.
- ❑ **ACTIVITY 9 – Tribonacci function**
  - Based on the Fibonacci sequence, we have a sequence known as Tribonacci. In it each element is the sum of the previous 3.
  - The succession is 1,1,3,4,7,13,24,44,81,149,274,etc.
  - Resolves the function recursively and iteratively
- ❑ **ACTIVITY 10 – Create "Filter" function**
  - The filter method of the arrays seen in this topic, allows to indicate a **callback** function, to apply a filter to the elements of an array.
  - It is good practice to learn how to implement callback functions to try to create our own functions.
  - Creates a function called a filter that receives an array and a callback function. The callback function will be understood to have a single parameter and that it returns true or false depending on the criterion we establish.
  - Our filter function will return a new array with the elements that meet the criteria established in the callback function.
  - Test it with various arrays and filter functions that you set to measure.

C/ Literat Azorín, 1 46702 – GANDIA
Tfno. 962829430 – Fax. 962829431
Codi: 46004221 – CIF: Q-9655627-I
46004221@edu.gva.es
http://iesmariaenriquez.com

OPTIONAL

- ❑ **ACTIVITY 11 – Minesweeper Map**
  - Create a web application that shows a map of the popular minesweeper game in which the drawn mines appear and also is indicated in the boxes without mines, the mines that are around.
  - Do it in a modular way so that we divide the application into a series of functions. Specifically, we recommend:
    - A function to which we send the minesweeper board (it would be a 2-dimensional array) and randomly place mines on it. This could be divided into 2, one that receives the array of the board and a position on it and returns the mines around that position. The main function simply invokes the first one by going through each square.
    - Another function that runs through the board by marking in each box the mines that are around. Finally a function that draws the dashboard on a web page.
  - The user will be asked for the size of the board and the mines to be placed.
  - Example of final result (9x9 board and 16 mines)

MINE	3	3	MINE	2	MINE	MINE	2	
2	MINE	MINE	2	2	4	MINE	3	
1	3	3	3	2	4	MINE	3	1
	1	MINE	2	MINE	MINE	3	MINE	1
	1	2	4	4	3	2	1	1
		1	MINE	MINE	2	2	2	1
		1	2	2	2	MINE	MINE	1
					1	2	2	1