**Programming Test - 4Geeks Academy**

Hint: You can use “mod” or “%” for calculating if a number is divisible (a multiple) by 3 or 5.

For example: Is 15 a multiple of 3?

| if(15 % 3 == 0) *//will be true if its multiple of 3* |
| --- |

### 

### Exercise #1

| Write a program that sums all the numbers divisible by 3 or 5 below **1000**. For example: All the numbers divisible by 3 and 5 below **10** are: 3,5,6,9 and their sum is 23. | Si listamos todos los números por debajo del **10** que son múltiplos de 3 o 5 obtenemos: 3, 5, 6 y 9. La suma de estos múltiplos es 23. Realice un algoritmo para encontrar la suma de todos los múltiplos de 3 y 5 por debajo de **1000**. |
| --- | --- |

### 

| 💻 Your solution here (googling, we don’t care about syntax errors, just logic):  // code |
| --- |

### 

### 

### Exercise #2

| Write a program that prints the numbers from 1 to 100 and for multiples of '3' print "Tic" instead of the number and for the multiples of '5' print "Toc", if the number is multiple of 3 and 5 then print “TicToc”. | Escriba una rutina que imprima los números del 1 al 100 pero: cuando el número sea **múltiplo de 3**, imprima “**Tic**”, en lugar del número. Cuando el número sea **múltiplo de 5**, imprima “**Toc**”, en lugar del número. Cuando el número sea **múltiplo tanto de 3 como de 5**, imprima “**TicToc**”, en lugar del número. |
| --- | --- |

**Example Output:**

|  | for (let i = 0; i<=100; i++){  if(i % 3 === 0){  console.log("tic")  }else if (i % 5 === 0){  console.log("Toc")  }else if ( (i % 5 === 0) && (i % 3 === 0){  console.log(“TicToc”)  }  }else  Console (i)  for (let i = 0; i<=100; i++){  if(i % 3 === 0){  console.log("tic")  }else if (i % 5 === 0){  console.log("Toc")  }else if (){  console.log(“TicToc”)  }  console.log(`Numero ${i} ES PAR `) |
| --- | --- |

### 

| 💻 Your solution here (no googling, we don’t care about syntax errors, just logic):  // code |
| --- |

### 

### Exercise #3

| **const** storage = [  { id: 1, name: 'Miami' },  { id: 2, name: 'Tampa' },  { id: 3, name: 'Orlando' }  ];  **const** products = [  { id: 1, SerialNumber: '86620855', name: 'Monitor'},  { id: 2, SerialNumber: '73178559', name: 'MotherBoard'},  { id: 3, SerialNumber: '73826497', name: 'Notebook'},  { id: 4, SerialNumber: '88587715', name: 'Consoles'},  { id: 5, SerialNumber: '94020190', name: 'SmartPhones'},  { id: 6, SerialNumber: '99804238', name: 'HeadSets' }  ];  **const** items= [  { productId: 6, storageId: 1, balance: 150 },  { productId: 1, storageId: 3, balance: 180 },  { productId: 5, storageId: 3, balance: 1350 },  { productId: 2, storageId: 2, balance: 56 },  { productId: 3, storageId: 1, balance: 230 },  { productId: 5, storageId: 2, balance: 150 },  { productId: 3, storageId: 3, balance: 459 },  { productId: 2, storageId: 3, balance: 190 },  { productId: 4, storageId: 3, balance: 510 },  { productId: 5, storageId: 1, balance: 890 },  { productId: 1, storageId: 2, balance: 16 },  { productId: 5, storageId: 3, balance: 375 },  { productId: 6, storageId: 1, balance: 192 },  { productId: 2, storageId: 3, balance: 100 },  { productId: 3, storageId: 2, balance: 54 },  { productId: 3, storageId: 1, balance: 90 },  { productId: 4, storageId: 3, balance: 135 },  { productId: 2, storageId: 1, balance: 382 },  { productId: 5, storageId: 2, balance: 170 },  { productId: 1, storageId: 3, balance: 10 },  { productId: 5, storageId: 2, balance: 6 },  { productId: 6, storageId: 1, balance: 162 },  { productId: 2, storageId: 2, balance: 100 }  ] |
| --- |

| Write the code to generate an object whose keys are the names of the storages and the values are an array with only the SerialNumber of its products ordered alphabetically. | Escribe el código para generar un objeto en que las keys (claves) sean los nombres de los depósitos y los valores un arreglo con los números de serie de los productos ordenados alfabéticamente por nombre. |
| --- | --- |

### 

| 💻 Your solution here (no googling, we don’t care about syntax errors, just logic):  // code |
| --- |

### Exercise #4

| Make a program that filters a list of strings and returns a list with only your friends name in it. If a name has exactly 4 letters in it, you can be sure that it has to be a friend of yours! Otherwise, you can be sure he’s not…  **Friend** = [“Ryan”, “Kieran”, “Mark”, “Miguel”] **shouldBe** [“Ryan”, “Mark”] | Haga un programa que filtre el arreglo y devuelva un arreglo con solo el nombre de sus amigos. Si un nombre tiene exactamente 4 letras, ¡puedes estar seguro que es amigo tuyo! De lo contrario, puede estar seguro de que no…  **Amigo** = [“Ryan”, “Kieran”, “Mark”, “Miguel”] **Deberia** [“Ryan”, “Mark”] |
| --- | --- |

### 

| 💻 Your solution here (no googling, we don’t care about syntax errors, just logic):  // code |
| --- |

### 

### Exercise #5

| Write a function called ***sumDigits*** that returns the sum of all the digits in a given number.  For example:  5646 => 5+6+4+6 => 21 | Escribe una función llamada ***sumaDigitos*** que retorne la suma de todos los dígitos de un número dado, por ejemplo:  5646 => 5+6+4+6 => 21 |
| --- | --- |

### 

| 💻 Your solution here (no googling, we don’t care about syntax errors, just logic):  // code |
| --- |

### 

### 