Programación de algoritmos básicos.

1.- Convertir Celsius a Fahrenheit:

```
function convertCtoF(celsius) {
  let fahrenheit = (celsius * 9/5) + 32; //Crea la variable fahrenheit y le asigna el resultado
  return fahrenheit;                     //de multiplicar celsius por 9/5 y sumarle 32
}

convertCtoF(30);
```

2.- Invertir una cadena:

```
function reverseString(str) {

  let strInvert = "";          //Creamos una variable y la inicializamos como un string vacío

  for(let i = str.length -1 ; i  >= 0 ; i--){  //Recorremos el string de la última letra a la primera
    strInvert += str[i];     //En cada pasada añadimos la letra al string
  }
  return strInvert;        //Devolvemos el string
}
```

3.- Factorizar un número:

```
function factorialize(num) {
  let factorial = 1;    //Creamos una variable factorial y la inicializamos a 1

    for(let i = 2; i <= num; i++){ //Bucle crea una variable i = 2, mientras i sea menor o igual
      factorial *= i;          //al número dado, coge factorial y guarda en ella el resultado de
    }                          //multiplicar factorial por i

  return factorial;          //Devuelve la variable factorial
}
factorialize(5);
```

4.- Encuentra la palabra más larga en una cadena:

```
function findLongestWordLength(str) {
  let words = str.split(" ");  //.split(), divide un string en una cadena de string, el parámetro
                               //indica el carácter de separación de la cadena (espacio blanco).

 let longestWord = 0;     //Creamos una variable donde almacenaremos la longitud máxima

  for(let i = 0; i < words.length; i++){    //Iteramos sobre el array de palabras

    if(words[i].length > longestWord){  //Si la palabra es mayor que la longitud máxima
      longestWord = words[i].length;   //Igualamos la longitud máxima a la de la palabra
    }
```

```
  }
return longestWord;    //Una vez termina el bucle, devolvemos la longitud máxima
}
findLongestWordLength("The quick brown fox jumped over the lazy dog");
```

5.- Devuelve los números mayores de los arreglos:

```
function largestOfFour(arr) {

  let result = [];  //Creamos un arreglo vacío, donde almacenaremos nuestros resultados

  for(let i = 0; i < arr.length; i++){ //Iteramos sobre el arreglo arr

   let longestNumber = arr[i][0];  //Establecemos el primer número más grande

    for(let y = 0; y < arr[i].length; y++){ //Iteramos sobre el sub-arreglo arr[i]

      //Si el número actual es mayor que longestNumber lo actualiza
      if(arr[i][y] > longestNumber){
        longestNumber = arr[i][y];
      }
    }
    result[i] = longestNumber; //Almacena el longestNumber dentro del arreglo result
  }
  return result; //Devuelve el arreglo
}
largestOfFour([[4, 5, 1, 3], [13, 27, 18, 26], [32, 35, 37, 39], [1000, 1001, 857, 1]]);
```

6.- Confirma el final:

```
function confirmEnding(str, target) {
//Creamos una variable y la inicializamos al string que devuelve slice()
//La sección de la cadena comienza en (str.length - target.length) y termina al final de esta

  let start = str.slice(str.length - target.length);
//Luego comprueba que la cadena creada sea igual a target
  if(start === target){
    return true;
  } else {
    return false
  }
}
confirmEnding("Bastian", "n");
```

7.- Repite una cadena por un número de veces:

```
function repeatStringNumTimes(str, num) {
  let result = ""; //Creamos un string vacío donde almacenaremos el resultado
  for(let i = 0; i < num; i++){ //Creamos i=0, mientras i sea menor al número dado, i +1
    result += str; //Añadimos el valor de str en result
  }
 return result;  //devolvemos result
}
repeatStringNumTimes("abc", 4);
```

8.- Recorta una cadena:

```
function truncateString(str, num) {
  // Creamos una variable y la igualamos a la sección de str que comienza en 0
  let recorte = str.slice(0, num);   //Y acaba en num

  if(str.length > num){   //Si num es menor a la longitud total del str
    recorte += "...";          //Añadimos … al final de recorte
  }
  return recorte;      //Devolvemos recorte
}

truncateString("A-tisket a-tasket A green and yellow basket", 8);
```

9.- Busca guardianes:

```
function findElement(arr, func) { //Creamos una función que recibe dos argumentos
                                  //Un arreglo y otra función
let num = 0;                 //Busca y devuelve un elemento del arreglo y le aplica la función

  for(let i = 0; i < arr.length; i++){    //Recorre el arreglo

    num = arr[i];             //asigna el valor actual del arreglo a num

    if(func(num)){        //Prueba la función con num
      return num;         //Si es true, devuelve num
    }
  }
  return undefined;    //Si no pasa la prueba devuelve undefined
}

findElement([1, 2, 3, 4], num => num % 2 === 0);
```

10.- Boo who (booleano quién):

```
function booWho(bool) {  //Creamos una función que recibe un argumento

  if(bool === true || bool === false){ //Si bool es true o false (posibilidades de boolean)
    return true;   //Devuelve true
  }
  return false;  //Si no, devuelve false
}

booWho(null);
```

11.- Haz que la primera letra de una palabra esté en mayúscula:

```
function titleCase(str) {  //Creamos la función que recibirá como argumento un string

  let words = str.split(" "); //Creamos un array que contendrá cada palabra del string por
  let newString = "";         //separado
  let finalString = "";       //Creamos dos strings vacios

  for(let i = 0; i < words.length; i++){  //Iteramos sobre el array con las palabras
     //Pasamos a mayúsculas la primera letra y el resto a minúsculas
    words[i] = words[i][0].toUpperCase() + words[i].slice(1).toLowerCase();
    newString += words[i] + " ";       //Añadimos cada palabra del array + espacio a un string
    finalString = newString.slice(0, -1)         //Quitamos el último espacio en blanco
  }
  return finalString;    //Devolvemos el string
}
titleCase("I'm a little tea pot");
```

12.- Cortar y trocear:

```
function frankenSplice(arr1, arr2, n) { //Función que toma 3 argumentos: 2 array y numero

  let firstHalf = arr2.slice(0, n); //Extraemos la primera mitad del array hasta el número n
  let secondHalf = arr2.slice(n, arr2.length);  //Extraemos la segunda mitad

  for(let i = 0; i  < arr1.length; i++){ //Iterramos arr1, insertandolo en la primera mitad
     firstHalf.push(arr1[i]);
  }
  for(let i = 0; i  < secondHalf.length; i++){ //Iteramos segunda mitad insertando en primera
     firstHalf.push(secondHalf[i]);
  }
return firstHalf; //Devolvemos el array completo almacenado en primera mitad
}
frankenSplice([1, 2, 3], [4, 5, 6], 1);
```

13.-Rebote falsy. Elimina todos los valores falsos de un arreglo:

```
//Dentro del bucle for, se evalúa cada elemento utilizando una condición if. Si el elemento
//es considerado "verdadero" en JavaScript (es decir, si no es null, undefined, 0, NaN,
//false o una cadena vacía ''), entonces se agrega al nuevo arreglo arrayFiltrado utilizando
//el método push.

function bouncer(arr) {
  const arrayFiltrado = [];
  for (let i = 0; i < arr.length; i++) {
    if (arr[i]) {
      arrayFiltrado.push(arr[i])
      }
  }
  return arrayFiltrado;
}
```

14.- ¿Dónde pertenezco?:

```
function getIndexToIns(arr, num) {

//El método sort se llama sobre el array arr y se utiliza una función de comparación
// (a, b) => a - b para ordenar los elementos de forma ascendente.

  arr.sort((a, b) => a - b);

//Dentro del bucle for, se compara cada elemento del array ordenado con el número num.
//Si se encuentra un elemento del array ordenado que es mayor o igual a num, la función
//devuelve la posición actual del bucle for (es decir, la posición en la que se debería
//insertar num en el array ordenado).

  for (let i = 0; i < arr.length; i++) {
    if (arr[i] >= num) return i;
  }

  return arr.length;
}
```

15.- Mutaciones:

```
function mutation(arr) {  //Recibe un argumento con dos arreglos

  let objetivo = arr[0].toLowerCase(); //Coje el primer arreglo y lo pasa a minúsculas
  let test = arr[1].toLowerCase();       //Coje el segundo arreglo y lo pasa a minúsculas

  for(let i = 0; i < test.length; i++){  //Itera sobre cada elemento del segundo arreglo

    //indexOf() comprueba si cada elemento del segundo array está en el primero
    if(objetivo.indexOf(test[i]) < 0) {return false} //Si no lo encuentra devuelve -1, false

  } return true;  //Si el resultado es mayor a 0, devuelve true
```

```
}
mutation(["hello", "hey"]);
```

16.- Monito trocitos:

```
//Pasa los test pero no está bien

function chunkArrayInGroups(arr, size) {

  let save = arr.length + 1; //Asignamos la long de arr+1 a save
  let result = [];   //Creamos un array vacío

  for (let i = 0; i < save; i++) {     //Iteramos sobre save
    result[i] = arr.splice(0, size);  //Vamos añadiendo los size valores desde el índice 0 de
                                      //arr a el índice i en result. Y los eliminamos de arr
    save -= size;                     //restamos size al valor de save
    }

  if (arr.length !== 0){result.push(arr);}  //Si queda algo en arr lo añadimos al final de result

  return result;
}
```