

Tratamiento de cadenas

Un dato de tipo texto puede ser visto como un arreglo de caracteres (letras, números, etc.) y por lo tanto se puede manejar de manera parcial.

Ejemplo:

```
let nombreApellido : string = 'Juan Perez';  
let inicialNombre : string = nombreApellido[0];  
let inicialApellido : string = nombreApellido[5];  
console.log('Nombre y Apellido: ', nombreApellido, ' Iniciales: ', inicialNombre, inicialApellido);
```

```
1 let nombreApellido: string = 'Juan Perez';  
2 let inicialNombre: string = nombreApellido[0];  
3 let inicialApellido: string = nombreApellido[5];  
4 console.log('Nombre y Apellido: ', nombreApellido, ' Iniciales: ', inicialNombre, inicialApellido);  
5
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

C:\cursos\cfs\arreglos-funciones>ts-node cadenas
Nombre y Apellido: Juan Perez Iniciales: J P

Tratamiento de cadenas

En TS existen diversas funciones pre-definidas para manejar esto:

- length: método que, al igual que con arreglos, retorna la longitud de una cadena.

```
let cadena : string = "ABCDE";  
console.log(cadena.length); //retorna 5
```

- indexOf(textoBuscado): método que retorna el índice de (o sea la posición de) la primer ocurrencia de textoBuscado en una cadena. (retorna -1 si no está, admite un 2do parámetro entero para iniciar la búsqueda desde ahí)

```
let cadena : string = "Tengo que hacer muchos ejercicios";  
console.log(cadena.indexOf("hacer")); //retorna 10
```

- lastIndexOf(textoBuscado): método que retorna el índice de (o sea la posición de) la última ocurrencia de textoBuscado en una cadena. (retorna -1 si no está, admite también un 2do parámetro entero para iniciar la búsqueda desde ahí)

```
let cadena : string = "Tengo que hacer y hacer y hacer muchos ejercicios";  
console.log(cadena.lastIndexOf("hacer")); //retorna 26
```

Tratamiento de cadenas

- substring(inicio, final) método que retorna la porción de la cadena entre las posiciones inicio y final.

```
let cadena : string = "Tengo que hacer y hacer y hacer muchos ejercicios";  
console.log(cadena.substring(10, 15)); //retorna "hacer"
```

- substr(inicio, largo) método que retorna la porción de la cadena de tamaño largo a partir de la posición inicio.

```
let cadena : string = "Tengo que hacer y hacer y hacer muchos ejercicios";  
console.log(cadena.substr(10, 21)); //retorna "hacer y hacer y hacer"
```

- toUpperCase() método que retorna una cadena con todos sus caracteres pasados a mayúsculas.

```
let cadena : string = "Tengo que hacer y hacer y hacer muchos ejercicios";  
console.log(cadena.toUpperCase()); //retorna "TENGO QUE HACER Y HACER Y HACER MUCHOS EJERCICIOS"
```

- toLowerCase() método que retorna una cadena con todos sus caracteres pasados a minúsculas.

```
let cadena : string = "Tengo que hacer y hacer y hacer muchos ejercicios";  
console.log(cadena.toLowerCase()); //retorna "tengo que hacer y hacer y hacer muchos ejercicios"
```

Tratamiento de cadenas

Conversión de tipos

- toString() método que (aplicado a un número) retorna una cadena que representa los caracteres de cada dígito.

```
let numero : number = 2021;  
console.log(numero.toString()); //retorna "2021"
```

- parseInt() método que convierte una cadena con caracteres numéricos sin símbolo decimal en un número entero.

```
let cadena : string = "2021";  
console.log(parseInt(cadena)); //retorna 2021
```

- parseFloat() método que convierte una cadena con caracteres numéricos con símbolo decimal en un número decimal.

```
let cadena : string = "3.14159";  
console.log(parseFloat(cadena)); //retorna 3.14159
```

Tratamiento de cadenas

Ejercicio: Invertir una palabra ingresada por el usuario.

ESPEJO



OJEPSE

Tratamiento de cadenas

Ejercicio: Invertir una palabra ingresada por el usuario.

```
import * as rls from 'readline-sync';  
  
let palabra : string = rls.question("Ingrese la palabra a invertir:");  
  
let cantidadLetras : number = palabra.length;  
  
// console.log("Cargando vector");  
// cargarVector(palabra, cantidadLetras);  
  
console.log("Mostrando invertido");  
mostrarVectorInvertido(palabra, cantidadLetras);  
  
console.log("Invierto los valores en el vector");  
invertirVector(palabra, cantidadLetras);  
  
console.log("Mostrando vector");  
mostrarVector(palabra, cantidadLetras);
```

Re-usamos el algoritmo escrito para invertir un vector con una ligera modificación.

No es necesario cargar el vector.

Tratamiento de cadenas

Ejercicio: Convertir una palabra ingresada por el usuario en clave, según las reglas siguientes:

- si el caracter es una vocal reemplazar aeiou por . , ; : ! respectivamente.
- si el caracter es un número o un operador matemático (+ - * /) queda igual.
- si el caracter es una consonante minúscula pasar a mayúscula y viceversa.

MarcelO3980  ***m.RC,Lo3980***

Tratamiento de cadenas

Ejercicio: Convertir una palabra ingresada por el usuario en clave, según reglas vistas.

```
import * as rls from "readline-sync";
let palabra : string = rls.question("Indique la palabra a codificar: ");
console.log(`La palabra ingresada: ${palabra} se convierte en: ${convertirEnClave(palabra)}`);

function convertirEnClave(palabra : string) : string {
    let vocales : string = "aeiou";
    let signos : string = ",.!:\"";
    let matematicos : string = "0123456789+-*/\";
    let clave : string = "";
    for (let index = 0; index < palabra.length; index++) {
        if (matematicos.indexOf(palabra[index]) >= 0) {
            clave += palabra[index];
        } else {
            if (vocales.indexOf(palabra[index]) >= 0) {
                clave += signos[vocales.indexOf(palabra[index])];
            } else {
                if (palabra[index] == palabra[index].toUpperCase())
                    clave += palabra[index].toLowerCase();
                else
                    clave += palabra[index].toUpperCase();
            }
        }
    }
    return clave;
}
```

```
C:\cursos\cfs\arreglos-funciones>ts-node encriptar
Indique la palabra a codificar: Marcelo03980
La palabra ingresada: Marcelo03980 se convierte en: m.RC,Lo3980
```


Técnicas de Programación

Programador full-stack

Estructuras de Datos y Métodos (Ejercicios)

Estructuras de Datos y Métodos

Cadenas

- Solicite al usuario que ingrese un texto y retórnalo convertido en un nombre de variable/función con las reglas camelCase
- Por ejemplo, si el usuario ingresa:

convertir texto segun camel case

el programa lo debe convertir en:

convertirTextoSegunCamelCase

Estructuras de Datos y Métodos

Producto Escalar

- Cargue dos arreglos de dimensión N números (la cantidad es ingresada por el usuario)
- Calcule el producto escalar entre los dos arreglos:

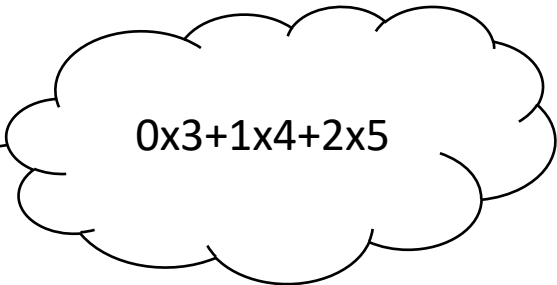
Si $A < a_1, b_1, c_1 >$ y $B < a_2, b_2, c_2 >$

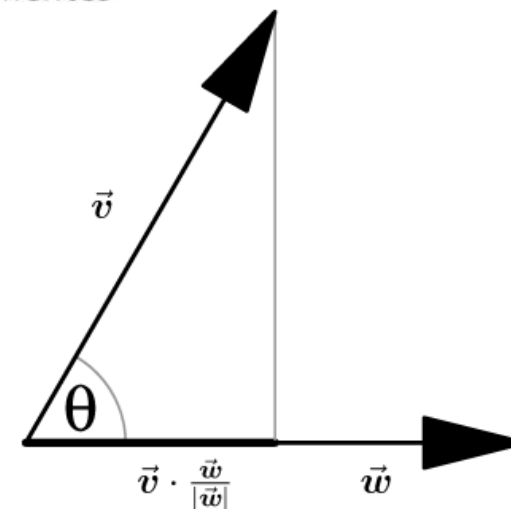
El producto escalar entre A y B en función de sus componentes está dado por:

$$A \cdot B = a_1a_2 + b_1b_2 + c_1c_2$$

Ejemplo:

n = 3
v1 = 0, 1, 2
v2 = 3, 4, 5
producto = 14

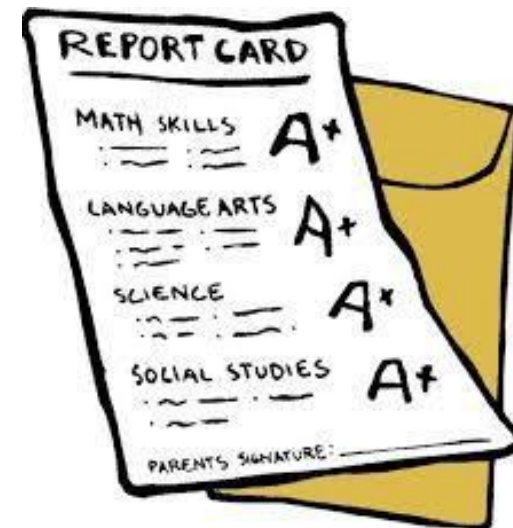

$$0 \times 3 + 1 \times 4 + 2 \times 5$$



Estructuras de Datos y Métodos

Promedio Escolar

- Desarrolle un algoritmo que permita cargar alumnos y sus notas en los tres trimestres
- Se debe permitir obtener el promedio anual (es decir, de sus tres notas) de un alumno (ingresado por el usuario)
- Luego de resolverlo, pensar en aprovechar métodos y discutir cómo representar la información



Estructuras de Datos y Métodos

Cine

- Diseñar un algoritmo que recorra las butacas de una sala de cine y determine cuántas butacas desocupadas hay
- Suponga que para modelar este problema, se utiliza un arreglo con valores lógicos
 - La presencia de un valor verdadero (true) en el arreglo indica que la butaca está ocupada
 - La presencia de un valor falso (false) en el arreglo indica que la butaca está desocupada



Estructuras de Datos y Métodos

Multiplicación

- Implemente un método llamado “multiplicarArreglo” que recibe como parámetros tres arreglos de Enteros del mismo tamaño
- Los dos primeros arreglos contienen los números que se quieren multiplicar
- El tercer arreglo almacena el cálculo de la multiplicación de cada posición de los dos arreglos
- Utilice este método para multiplicar los siguientes cuatro arreglos de tres elementos

v1: [1, 2, 3]

v2: [4, 5, 6]

v3: [2, 1, 2]

v4: [1, 2, 1]

vResultado (v1*v2*v3*v4): [8, 20, 36]

aproveche las ventajas de métodos
para resolver el ejercicio



Estructuras de Datos y Métodos

Jardín

- El jardín infantil necesita un programa para poder asignar cursos a las aulas
- Se cuentan con tres aulas: **Azul**, **Verde** y **Amarilla**
- Cada aula cuenta con una capacidad diferente (es decir, un número de bancos)
- La aula **Azul** tiene 40 bancos, la **Verde** 35 y la **Amarilla** 30
- Dado un número de infantes ingresado por el usuario, el programa deberá determinar el aula que minimice la cantidad de bancos vacíos
- La salida del algoritmo es el color que identifica al aula asignada

Ejemplo: si la cantidad de personas de un curso es 34, entonces el aula a asignar será la Verde. El aula Amarilla no puede ser asignada porque la cantidad de personas es menor a la cantidad de bancos disponibles. El aula Azul es descartada porque quedan más bancos inutilizados que en el aula Verde (6 versus 1).

