

# Algoritmos y Estructuras de Datos II

TALLER - 15 de marzo 2022

## Laboratorio 0: Repaso de C

- Original 2020: Leonardo Rodríguez
- Revisión 2022: Marco Rocchietti

### Objetivos

1. Repaso de arreglos y estructuras en C.
2. Repaso de uso de las librerías **assert.h**, **stdio.h**, **stdbool.h**
3. Lectura y comprensión de código.

### Ejercicio 1

En el archivo `max_min.c` implementar la función:

```
struct max_min_result compute_max_min(int array[], unsigned int length);
```

que calcula el valor máximo y el valor mínimo del arreglo dado. La estructura que devuelve la función tiene la siguiente definición:

```
struct max_min_result {  
    int max_value;  
    int min_value;  
    unsigned int max_position;  
    unsigned int min_position;  
};
```

Los cuatro campos de la estructura son los siguientes: valor máximo, valor mínimo, posición del máximo y posición del mínimo. Por ejemplo

```
int array[] = {0, 9, -1, 4};  
result = compute_max_min(array, 4);  
printf("d", result.max_value)    // Imprime 9  
printf("%d", result.min_value)   // Imprime -1  
printf("%u", result.max_position) // Imprime 1  
printf("%u", result.min_position) // Imprime 2
```

La función debe implementarse usando un único ciclo (`for` o `while`). En la función `main()` se le debe solicitar al usuario que ingrese uno por uno los elementos del arreglo. Para ello utilizar la función `scanf()` de la librería **stdio.h**.

## Ejercicio 2

En el archivo `tictactoe.c` se encuentra una implementación incompleta del clásico juego [tres en línea](#) (conocido como *tatetí* o *tictactoe*). El tablero 3x3 se representa con una matriz en C, declarada de la siguiente manera.

```
#define BOARD_SIZE 3
char board[BOARD_SIZE][BOARD_SIZE] = {
    { '-', '-', '-' },
    { '-', '-', '-' },
    { '-', '-', '-' }
};
```

Inicialmente todas las “celdas” del tablero se encuentran vacías, lo que se representa con el carácter '-'. Los caracteres 'X' y 'O' se utilizan para representar la cruz y el círculo, respectivamente. El juego permite al usuario llenar una celda del casillero con 'X' y 'O' de acuerdo a su turno, teniendo la posibilidad de elegir con un número entero en qué celda desea marcar el tablero. Las nueve celdas del tablero se encuentran numeradas de la siguiente forma:

```
.....
| 0: -      | 1: -      | 2: -      |
.....
| 3: -      | 4: -      | 5: -      |
.....
| 6: -      | 7: -      | 8: -      |
.....
```

El juego está incompleto puesto que no detecta cuándo hubo un ganador, o si hubo empate. Para ello se deben implementar correctamente las funciones:

```
bool has_free_cell(char board[BOARD_SIZE][BOARD_SIZE])
```

que devuelve verdadero si hay una celda libre (marcada con '-') en el tablero `board`, y devuelve `false` en caso contrario;

```
char get_winner(char board[BOARD_SIZE][BOARD_SIZE])
```

que devuelve el jugador ganador ('X' o 'O') si lo hubo, o '-' si todavía no hay ganador. Para ello se debe recorrer la matriz para verificar si alguna columna, fila o diagonal tiene 3 veces consecutivas el mismo carácter.

Se pide leer el código y comprender cómo se logra el funcionamiento del juego.