Código de Assembler

- ***Alumno:*** Jeriel Estrada Candiano
- ***Profesor:*** Santiago Trini
- ***Materia:*** CATE
- ***Fecha de Entrega:*** 14/10/2025
- ***Ciclo Lectivo:*** 2025

Código:

```
        .data
array:  .space 400              // Reservamos espacio para 100 enteros (4 bytes c/u)
newline: .asciiz "\n"           // Salto de línea para imprimir después

        .text
        .globl main


 Función: fill(a, n, s)
//llena el arreglo con valores que empiezan en s

fill:
        li   $t0, 0             // i = 0
fill_loop:
        bge  $t0, $a1, fill_end  // si i >= n, terminar

        sll  $t1, $t0, 2        // t1 = i * 4 (porque cada int ocupa 4 bytes)
        add  $t2, $a0, $t1      // t2 = dirección de a[i]
        sw   $a2, 0($t2)        // a[i] = s

        addi $a2, $a2, 1        // s++
        addi $t0, $t0, 1        // i++
        j    fill_loop          // volver al inicio del bucle

fill_end:
        jr   $ra                // volver al main



 Función: max(a, n)
// busca el número más grande del arreglo
max:
        lui  $t0, 0x8000        // r = INT_MIN (0x80000000)
        move $t1, $t0           // guardamos ese valor en t1 (r)
        li   $t2, 0             // i = 0
max_loop:
        bge  $t2, $a1, max_end   // si i >= n, terminar

        sll  $t3, $t2, 2        // t3 = i * 4
        add  $t4, $a0, $t3      // t4 = dirección de a[i]
        lw   $t5, 0($t4)        // t5 = a[i]
```

```
        ble  $t5, $t1, skip      // si a[i] <= r, saltar
        move $t1, $t5            // si no, r = a[i]
    skip:
        addi $t2, $t2, 1         // i++
        j    max_loop            // volver al inicio del bucle

    max_end:
        move $v0, $t1            // return r
        jr   $ra
```

Función Principal: main

```
    main:
        la   $a0, array          // a0 = dirección del arreglo
        li   $a1, 100            // a1 = tamaño (n)
        li   $a2, 87             // a2 = valor inicial (s)
        jal  fill                // llama a fill(a, 100, 87)

        la   $a0, array          // a0 = dirección del arreglo
        li   $a1, 100            // a1 = tamaño
        jal  max                 // llama a max(a, 100)
                                 // el resultado queda en v0

        move $a0, $v0            // pasamos el valor máximo a imprimir
        li   $v0, 1              // syscall 1 = print_int
        syscall

        la   $a0, newline        // imprime salto de línea
        li   $v0, 4
        syscall

        li   $v0, 10             // syscall 10 = salir del programa
        syscall
```

Instrucciones en Binario:

Instrucción: add $t2, $s0, $t1 / Binario:
00000010000010010101000000100000

Instrucción: addi $t0, $t0, 1/ Binario: 00100001000010000000000000000001

Instrucción: lw $t1, 0($t0) / Binario: 10001101000010010000000000000000