TALLER #5

```
section .data
    num1 db 16          ; Primera variable (ent
    num2 db 1           ; Segunda variable (entr
    result db 0         ; Espacio para almacenar

section .text
    global _start

_start:
    mov al, [num1]      ; Cargar num1 en AL
    add al, [num2]      ; Sumar num2 a AL
    add al, '0'         ; Convertir el resultado
```

```
section .data
    num1 db 9           ; Primera variable (
    num2 db 1           ; Segunda variable (
    result db 0         ; Espacio para almac

section .text
    global _start

_start:
    mov al, [num1]      ; Cargar num1 en AL
    add al, [num2]      ; Sumar num2 a AL
    add al, '0'         ; Convertir el resul

    mov [result], al    ; Guardar el carácte
```

Output:

:

```
section .data
    num1 db 11          ; Primera variable (entr
    num2 db 2           ; Segunda variable (entre
    result db 0         ; Espacio para almacenar

section .text
    global _start

_start:
    mov al, [num1]      ; Cargar num1 en AL
    add al, [num2]      ; Sumar num2 a AL
    add al, '0'         ; Convertir el resultado

    mov [result], al    ; Guardar el carácter ASC
```

Output:

=

```
section .data
    num1 db 11          ; Primera variable (ent
    num2 db 4           ; Segunda variable (entr
    result db 0         ; Espacio para almacenar

section .text
    global _start

_start:
    mov al, [num1]      ; Cargar num1 en AL
    add al, [num2]      ; Sumar num2 a AL
    add al, '0'         ; Convertir el resultado

    mov [result], al    ; Guardar el carácter AS
```

Output:

?

V. _

```
section .data
    num1 db 40          ; Primera variable
    num2 db 5           ; Segunda variable
    result db 0         ; Espacio para alma

section .text
    global _start

_start:
    mov al, [num1]      ; Cargar num1 en AL
    add al, [num2]      ; Sumar num2 a AL
    add al, '2'         ; Convertir el resu
```

Output:

_

```
section .data
    num1 db 20          ; Primera variable (entre 1
    num2 db 4           ; Segunda variable (entre 1 y
    result db 0         ; Espacio para almacenar el r

section .text
    global _start

_start:
    mov al, [num1]      ; Cargar num1 en AL
    Sub al, [num2]      ; restar num2 a AL
    add al, '2'         ; Convertir el resultado a AS

    mov [result], al    ; Guardar el carácter ASCII e
```

Output:

B

```
section .data
    num1 db 74          ; Primera variable (en
    num2 db 4           ; Segunda variable (ent
    result db 0         ; Espacio para almacena

section .text
    global _start

_start:
    mov al, [num1]      ; Cargar num1 en AL
    Sub al, [num2]      ; restar num2 a AL
    add al, '2'         ; Convertir el resultad

    mov [result], al    ; Guardar el carácter A
```

Output:

x

```
section .data
    num1 db 4           ; Primera variable (entre 1 y 3)
    num2 db 1           ; Segunda variable (entre 1 y 3)
    result db 0         ; Espacio para almacenar el resultado

section .text
    global _start

_start:
    mov al, [num1]      ; Cargar num1 en AL
    Sub al, [num2]      ; restar num2 a AL
    add al, '('         ; Convertir el resultado a ASCII

    mov [result], al    ; Guardar el carácter ASCII en 'resul
```

Output:

+

```
section .data
    num1 db 2          ; Primera variable (entre 1 y 3)
    num2 db 1          ; Segunda variable (entre 1 y 3)
    result db 0        ; Espacio para almacenar el resultado

section .text
    global _start

_start:
    mov al, [num1]     ; Cargar num1 en AL
    Sub al, [num2]     ; restar num2 a AL
    add al, '&'        ; Convertir el resultado a ASCII

    mov [result], al   ; Guardar el carácter ASCII en 'result
```

Output:

'

```
section .data
    num1 db 2          ; Primera variable (entre 1 y 3)
    num2 db 1          ; Segunda variable (entre 1 y 3)
    result db 0        ; Espacio para almacenar el resultado

section .text
    global _start

_start:
    mov al, [num1]     ; Cargar num1 en AL
    Sub al, [num2]     ; restar num2 a AL
    add al, 'z'        ; Convertir el resultado a ASCII

```

Output:

{