```
section .data

num1 db 5 ; declaracion de variables
num2 db 11
result db 0
msg db 'Resultado: ', 0

section .bss
buffer resb 4 ;reserva memoria

section .text
global _start

_start:
    mov al, [num1]          ; carga el valor de num1 en AL (AL = 5)
    add al, [num2]          ; suma el valor de num2 a AL
    mov [result], al        ; guarda el resultado en result

    ; Convertir el resultado a ASCII
    movzx eax, byte [result] ; Extiende con ceros el byte result a 32 bits en eax
    add eax, 48       ; Convertir el valor numérico en su
                      ; correspondiente ASCII ('0' = 48)
    mov [buffer], al ; Almacenar el carácter ASCII en el
                      ; buffer

    mov eax, 4
    mov ebx, 1
    mov ecx, msg      ; direccion del mensaje
    mov edx, 11       ; longitud del mensaje
    int 0x80           ; interrupcion del sistema

    mov eax, 4
    mov ebx, 1
    mov ecx, buffer ; dirección del buffer que contiene el caracter
    mov edx, 1
    int 0x80        ; interrupcion del sistema

    mov eax, 1
    xor ebx, ebx
    int 0x80
```

\

A

```
section .data

num1 db 6 ; declaracion de variables
num2 db 11
result db 0
msg db 'Resultado: ', 0

section .bss
buffer resb 4 ;reserva memoria

section .text
global _start

_start:
    mov al, [num1]         ; carga el valor de num1 en AL (AL = 5)
    add al, [num2]         ; suma el valor de num2 a AL
    mov [result], al       ; guarda el resultado en result

    ; Convertir el resultado a ASCII
    movzx eax, byte [result] ; Extiende con ceros el byte result a 32 bits en eax
    add eax, 48      ; Convertir el valor numérico en su
                     ; correspondiente ASCII ('0' = 48)
    mov [buffer], al ; Almacenar el carácter ASCII en el
                     ; buffer

    mov eax, 4
    mov ebx, 1
    mov ecx, msg      ; direccion del mensaje
    mov edx, 11       ; longitud del mensaje
    int 0x80           ; interrupcion del sistema

    mov eax, 4
    mov ebx, 1
    mov ecx, buffer ; dirección del buffer que contiene el caracter
    mov edx, 1
    int 0x80        ; interrupcion del sistema

    mov eax, 1
    xor ebx, ebx
    int 0x80
```

\

```
section .data

num1 db 33 ; declaracion de variables
num2 db 11
result db 0
msg db 'Resultado: ', 0

section .bss
buffer resb 4 ;reserva memoria

section .text
global _start

_start:
    mov al, [num1]          ; carga el valor de num1 en AL (AL = 5)
    add al, [num2]          ; suma el valor de num2 a AL
    mov [result], al        ; guarda el resultado en result

    ; Convertir el resultado a ASCII
    movzx eax, byte [result] ; Extiende con ceros el byte result a 32 bits en eax
    add eax, 48       ; Convertir el valor numérico en su
                      ; correspondiente ASCII ('0' = 48)
    mov [buffer], al ; Almacenar el carácter ASCII en el
                      ; buffer

    mov eax, 4
    mov ebx, 1
    mov ecx, msg      ; direccion del mensaje
    mov edx, 11       ; longitud del mensaje
    int 0x80           ; interrupcion del sistema

    mov eax, 4
    mov ebx, 1
    mov ecx, buffer ; dirección del buffer que contiene el caracter
    mov edx, 1
    int 0x80        ; interrupcion del sistema

    mov eax, 1
    xor ebx, ebx
    int 0x80
```

$

```
section .data

num1 db 35 ; declaracion de variables
num2 db 1
result db 0
msg db 'Resultado: ', 0

section .bss
buffer resb 4 ;reserva memoria

section .text
global _start

_start:
    mov al, [num1]          ; carga el valor de num1 en AL (AL = 5)
    add al, [num2]          ; suma el valor de num2 a AL
    mov [result], al        ; guarda el resultado en result

    ; Convertir el resultado a ASCII
    movzx eax, byte [result] ; Extiende con ceros el byte result a 32 bits en eax
    add eax, 0      ; Convertir el valor numérico en su
                    ; correspondiente ASCII ('0' = 48)
    mov [buffer], al ; Almacenar el carácter ASCII en el
                    ; buffer

    mov eax, 4
    mov ebx, 1
    mov ecx, msg      ; direccion del mensaje
    mov edx, 11       ; longitud del mensaje
    int 0x80           ; interrupcion del sistema

    mov eax, 4
    mov ebx, 1
    mov ecx, buffer ; dirección del buffer que contiene el caracter
    mov edx, 1
    int 0x80        ; interrupcion del sistema

    mov eax, 1
    xor ebx, ebx
    int 0x80
```

&

```
section .data

num1 db 35 ; declaracion de variables
num2 db 3
result db 0
msg db 'Resultado: ', 0

section .bss
buffer resb 4 ;reserva memoria

section .text
global _start

_start:
    mov al, [num1]          ; carga el valor de num1 en AL (AL = 5)
    add al, [num2]          ; suma el valor de num2 a AL
    mov [result], al        ; guarda el resultado en result

    ; Convertir el resultado a ASCII
    movzx eax, byte [result] ; Extiende con ceros el byte result a 32 bits en eax
    add eax, 0      ; Convertir el valor numérico en su
                    ; correspondiente ASCII ('0' = 48)
    mov [buffer], al ; Almacenar el carácter ASCII en el
                    ; buffer

    mov eax, 4
    mov ebx, 1
    mov ecx, msg      ; direccion del mensaje
    mov edx, 11       ; longitud del mensaje
    int 0x80           ; interrupcion del sistema

    mov eax, 4
    mov ebx, 1
    mov ecx, buffer ; dirección del buffer que contiene el caracter
    mov edx, 1
    int 0x80        ; interrupcion del sistema

    mov eax, 1
    xor ebx, ebx
    int 0x80
```

```
section .data

num1 db 1 ; declaracion de variables
num2 db 0
result db 0
msg db 'Resultado: ', 0

section .bss
buffer resb 4 ;reserva memoria

section .text
global _start

_start:
    mov al, [num1]          ; carga el valor de num1 en AL (AL = 5)
    add al, [num2]          ; suma el valor de num2 a AL
    mov [result], al        ; guarda el resultado en result

    ; Convertir el resultado a ASCII
    movzx eax, byte [result] ; Extiende con ceros el byte result a 32 bits en eax
    add eax, 48       ; Convertir el valor numérico en su
                      ; correspondiente ASCII ('0' = 48)
    mov [buffer], al ; Almacenar el carácter ASCII en el
                      ; buffer

    mov eax, 4
    mov ebx, 1
    mov ecx, msg      ; direccion del mensaje
    mov edx, 11       ; longitud del mensaje
    int 0x80           ; interrupcion del sistema

    mov eax, 4
    mov ebx, 1
    mov ecx, buffer ; dirección del buffer que contiene el caracter
    mov edx, 1
    int 0x80        ; interrupcion del sistema

    mov eax, 1
    xor ebx, ebx
    int 0x80
```

```
section .data
char db
msg db 'Resultado: ', 0

section .bss
buffer resb 4 ;reserva memoria

section .text
global _start

_start:
    mov al, 64 // inmediato
    mov [char], al

    mov eax, 4
    mov ebx, 1
    mov ecx, msg      ; direccion del mensaje
    mov edx, 11       ; longitud del mensaje
    int 0x80           ; interrupcion del sistema

    mov eax, 4
    mov ebx, 1
    mov ecx, char ; dirección del buffer que contiene el caracter
    mov edx, 1
    int 0x80        ; interrupcion del sistema

    mov eax, 1
    xor ebx, ebx
    int 0x80
```