ALP containing division, XOR, Left shift

```
section .text
global _start
_start:
Division:
 mov ecx, msg_division
 mov edx, len_division
 mov ebx, 1
 mov eax, 4
 int 0x80

  xor edx, edx

  mov eax, [num1]
  mov ebx, [num2]

  div ebx

  xor ecx, ecx
  xor edx, edx
  call print_integer.
logic_XOR:

   mov ecx, msg_XOR
   mov edx, len_XOR
   mov ebx, 1
   mov eax, 4
   int 0x80

   mov eax, [num1]
   mov ebx, [num2]

    xor eax, ebx

    xor ecx, ecx
    xor edx, edx
    call print_integer
```

```
Shift_left:

    mov ecx, msg-left
    mov edx, len- left
    mov ebx, 1
    mov eax, 4
    int 0x80

    mov eax, [num1]
    mov ebx, [num2]

    Shl eax, 3

    xor ecx, ecx
    xor edx, edx

    call print_integer

    jmp Exit

print_integer:

    mov ebx, 10
    div ebx
    add edx, '0'

    push edx
    xor edx, edx

    inc ecx
    cmp eax, 0
    jne print_integer

    xor eax, eax.

reverse:

    pop dword [result + eax]
    inc eax
    dec ecx
    cmp ecx, 0
    jne reverse
    mov edx, eax
    mov ecx, result
    mov ebx, 1
    mov eax, 4
    int 0x80

    xor eax, eax
    xor ebx, ebx
    xor ecx, ecx
    xor edx, edx

    ret
Exit:

    mov ecx, msg
    mov edx, len
    mov ebx, 1
    mov eax, 4
    int 0x80

    mov eax, 1
    int 0x80

Section.data
    num1 dd 200
    num2 dd 100

    msg db 0xA,
    len equ $ - msg

    msg-division db "The division is: "
    len_division equ $ - msg-division

    ~~msg-division~~

    msg_XOR db "The XOR is: "
    len_XOR equ $ - msg-XOR.
    msg-left db " The left shift is "
    len_left equ $ - msg-left

Section .bss

     result resb 8.
```