Keene Keannu Kurt C. De Jesus

BSCS3-1

```
%include 'functions.asm'

SECTION .data
    msg1 db 'Enter first number: ', 0
    msg2 db 'Enter second number: ', 0
    res db 'Sum: ', 0

SECTION .bss
    buf1 resb 10
    buf2 resb 10

SECTION .text
global _start

_start:
    mov eax, 4
    mov ebx, 1
    mov ecx, msg1
    mov edx, 21
    int 0x80

    mov eax, 3
    mov ebx, 0
    mov ecx, buf1
    mov edx, 10
    int 0x80

    mov ebx, 0
    call convert_to_int
    mov edi, ebx

    mov eax, 4
    mov ebx, 1
    mov ecx, msg2
    mov edx, 22
    int 0x80

    mov eax, 3
    mov ebx, 0
    mov ecx, buf2
    mov edx, 10
    int 0x80

    mov ebx, 0
    call convert_to_int

    add edi, ebx

    mov eax, 4
    mov ebx, 1
    mov ecx, res
    mov edx, 5
    int 0x80
```

```
    int 0x80

    mov eax, edi
    call iprintLF

    call quit

convert_to_int:
    xor ebx, ebx
convert_loop:
    mov al, byte [ecx]
    cmp al, 10
    je done_conversion
    sub al, '0'
    imul ebx, ebx, 10
    add ebx, eax
    inc ecx
    jmp convert_loop

done_conversion:
    ret
```

```
xenta@xenta-virtualbox
Enter first number: 5
Enter second number: 5
Sum: 10
```

```
iprint:
    push    eax
    push    ecx
    push    edx
    push    esi
    mov     ecx, 0

divideLoop:
    inc     ecx
    mov     edx, 0
    mov     esi, 10
    idiv    esi
    add     edx, 48
    push    edx
    cmp     eax, 0
    jnz     divideLoop

printLoop:
    dec     ecx
    mov     eax, esp
    call    sprint
    pop     eax
    cmp     ecx, 0
    jnz     printLoop

    pop     esi
    pop     edx
    pop     ecx
    pop     eax
    ret


iprintLF:
    call    iprint
 |
    push    eax
    mov     eax, 0Ah
    push    eax
    mov     eax, esp
    call    sprint
    pop     eax
    pop     eax
    ret


slen:
    push    ebx
    mov     ebx, eax

nextchar:
    cmp     byte [eax], 0
    jz      finished
    inc     eax
    jmp     nextchar

finished:
    sub     eax, ebx
    pop     ebx
    ret


sprint:
    push    edx
    push    ecx
    push    ebx
    push    eax
    call    slen

    mov     edx, eax
    pop     eax

    mov     ecx, eax
    mov     ebx, 1
    mov     eax, 4
    int     80h

    pop     ebx
    pop     ecx
    pop     edx
    ret


sprintLF:
    call    sprint

    push    eax
    mov     eax, 0AH
    push    eax
    mov     eax, esp
    call    sprint
    pop     eax
    pop     eax
    ret


quit:
    mov     ebx, 0
    mov     eax, 1
    int     80h
    ret
```