```
section .data
 ; Onde ficam dados inicializados. Normalmente o padrão para declarar
variável é:
 ; NOME TIPO (POde ser DB-BYTE,DD-DOUBLE WORD,DW-WORD) VALOR
 p DB "Digite um número: "
 len_p  equ $-p ; Tamanho da variável p. É necessário para passar ao
registrador edx
 pp DB "Digite o segundo número: "
 len_pp equ $-pp

 m DB "Resultado: "
 len_m equ $-m




section .bss
 ; Reservando espaço de 1 byte para posterior uso. No nosso caso,
troca-se apenas o D por RES
 numero RESB 2
 resultado RESB 2
 segundo RESB 2

section .text

global _start

_start:
 ; Qualquer instrução do trecho de programa _start ficará aqui. Neste
caso, _start é um rótulo dado á este trecho específico de programa.
_start é o fluxo principal do programa


; Em instruções que se exibe algo na tela. O EAX possui valor 4, que é
a interrupção para se exibir algo na tela. EDX armazena a quantidade
de caracteres a serem exibidas
 mov edx, len_p ; Tamanho da string que será exibir
 mov ecx, p ; Conteúdo da mensagem
 mov eax, 4 ; Chamada ao núcleo do sistema para exibir algo na tela
 mov ebx, 1 ; 0 = NADA; 1 = STDOUT (Saída); 2 = STDIN (Entrada)
 int 0x80 ; CHamada ao núcleo do sistema, que recebe o valor contido
em eax para executar

 ; Trecho abaixo  trata de receber o número digitado pelo usuário e o
armazena na variável numero
 mov eax, 3 ; Indica ao núcleo que é pra receber um valor preenchido
pelo usuário

 mov ebx, 2 ; STDIN (Preparar o programa para uma entrada de texto)
 mov edx, 1 ; Tamanho do campo recebido
 mov mov ecx, numero ; Armazena a entrada na variável numero
 int 0x80 ; Interrupção


 ; Exibindo na tela a segunda mensagem
 mov edx, len_pp
 mov ecx, pp
 mov eax, 4
 mov ebx, 1
 int 0x80
```

```
 ; Recebendo o segundo parâmetro
 mov eax, 3
 mov ebx, 2
 mov edx, 2
 mov ecx, segundo
 int 0x80

 ; Exibindo terceira mensagem
 mov edx, len_m
 mov ecx, m
 mov eax, 4
 mov ebx, 1
 int 0x80


; Efetuando a soma
 mov bl, [segundo] ; Atribuindo o valor da variável segundo ao
registrador BL
 mov al, [numero] ; Atribuindo o valor da variável numero ao
registrador AL

 sub bl, '0' ; Convertendo valor de ASCII em decimal para fins de
operação


 add al, bl ; Efetuando a soma, sendo que, o resultado fica armazenado
no registrador al

 mov [resultado], al ; Movendo o valor de al na variável resultado


 ; Trecho abaixo exibe o resultado na tela
 mov ecx, resultado
 mov edx, 2
 mov eax, 4
 mov ebx, 1
 int 0x80

 mov eax,1 ; SAIDA
 int 0x80
```

Para compilar o programa acima, copie e cole o código acima em seu editor de texto, e salve com o nome de soma.asm. Todo código-fonte em assembly deve ter a extensão .asm.

Em seguida, execute os seguintes comandos em linha de comando:

```
nasm -f elf64 soma.asm // Para compilar
ld soma.o -o soma // Para gerar o executável
./soma   // Para executar
```