PAI Assignment Lab 3 & 4

Name= SAMARTH VALSANGE

Roll no= 46

Enrollment No= ADT24SOCB0996

SY-06 Batch(B)

Asm codes of Addtion , Subtraction , Multiplication & Divison

Addition.asm

```
section .data
    msg1 db "Enter first number: ",0
    len1 equ $-msg1
    msg2 db "Enter second number: ",0
    len2 equ $-msg2
    resultMsg db "Result = ",0
    lenRes equ $-resultMsg
    newline db 10,0

section .bss
    num1 resb 2
    num2 resb 2
    res  resb 2

section .text
    global _start

_start:
    ; Ask for first number
    mov eax, 4
```

```
mov ebx, 1
mov ecx, msg1
mov edx, len1
int 0x80

; Read input
mov eax, 3
mov ebx, 0
mov ecx, num1
mov edx, 2
int 0x80

; Ask for second number
mov eax, 4
mov ebx, 1
mov ecx, msg2
mov edx, len2
int 0x80

; Read input
mov eax, 3
mov ebx, 0
mov ecx, num2
mov edx, 2
int 0x80

; Convert ASCII to number
mov al, [num1]
sub al, '0'
mov bl, [num2]
```

```
sub bl, '0'

; Add
add al, bl

; Convert back to ASCII
add al, '0'
mov [res], al

; Print "Result = "
mov eax, 4
mov ebx, 1
mov ecx, resultMsg
mov edx, lenRes
int 0x80

; Print result
mov eax, 4
mov ebx, 1
mov ecx, res
mov edx, 1
int 0x80

; Print newline
mov eax, 4
mov ebx, 1
mov ecx, newline
mov edx, 1
int 0x80
```

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

```
kaizen@kaizen:~$
kaizen@kaizen:~$ nano addition.asm
kaizen@kaizen:~$ nasm -f elf32 addition.asm -o addition.o
kaizen@kaizen:~$ ld -m elf_i386 -s -o addition addition.o
kaizen@kaizen:~$
kaizen@kaizen:~$ ./addition
Files first number: 2
second number: 3
Result = 5
```

Subtraction

```
section .data
    msg1 db "Enter first number: ",0
    len1 equ $-msg1
    msg2 db "Enter second number: ",0
    len2 equ $-msg2
    resultMsg db "Result = ",0
    lenRes equ $-resultMsg
    newline db 10,0

section .bss
    num1 resb 2
    num2 resb 2
    res  resb 2

section .text
    global _start
```

```
_start:
    ; Ask for first number
    mov eax, 4
    mov ebx, 1
    mov ecx, msg1
    mov edx, len1
    int 0x80

    ; Read input
    mov eax, 3
    mov ebx, 0
    mov ecx, num1
    mov edx, 2
    int 0x80

    ; Ask for second number
    mov eax, 4
    mov ebx, 1
    mov ecx, msg2
    mov edx, len2
    int 0x80

    ; Read input
    mov eax, 3
    mov ebx, 0
    mov ecx, num2
    mov edx, 2
    int 0x80
```

```
; Convert ASCII to number
mov al, [num1]
sub al, '0'
mov bl, [num2]
sub bl, '0'

; Subtract
sub al, bl

; Convert back to ASCII
add al, '0'
mov [res], al

; Print "Result = "
mov eax, 4
mov ebx, 1
mov ecx, resultMsg
mov edx, lenRes
int 0x80

; Print result
mov eax, 4
mov ebx, 1
mov ecx, res
mov edx, 1
int 0x80

; Print newline
mov eax, 4
mov ebx, 1
```

```
    mov ecx, newline
    mov edx, 1
    int 0x80

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

```
kaizen@kaizen:~$ gedit subtraction.asm
kaizen@kaizen:~$ nasm -f elf32 subtraction.asm -o subtraction.o
kaizen@kaizen:~$ ld -m elf_i386 -s -o subtraction subtraction.o
kaizen@kaizen:~$ ./subtraction
Enter first number: 5
Enter second number: 2
Result = 3
```

Multiplication.asm

```
section .data
    msg1 db "Enter first number: ",0
    len1 equ $-msg1
    msg2 db "Enter second number: ",0
    len2 equ $-msg2
    resultMsg db "Result = ",0
    lenRes equ $-resultMsg
    newline db 10,0

section .bss
    num1 resb 2
    num2 resb 2
    res  resb 2
```

```
section .text
    global _start

_start:
    ; Ask for first number
    mov eax, 4
    mov ebx, 1
    mov ecx, msg1
    mov edx, len1
    int 0x80

    ; Read input
    mov eax, 3
    mov ebx, 0
    mov ecx, num1
    mov edx, 2
    int 0x80

    ; Ask for second number
    mov eax, 4
    mov ebx, 1
    mov ecx, msg2
    mov edx, len2
    int 0x80

    ; Read input
    mov eax, 3
    mov ebx, 0
    mov ecx, num2
```

```
mov edx, 2
int 0x80

; Convert ASCII to number
mov al, [num1]
sub al, '0'
mov bl, [num2]
sub bl, '0'

; Multiply
mul bl      ; AX = AL * BL
add al, '0'  ; Convert back to ASCII
mov [res], al

; Print "Result = "
mov eax, 4
mov ebx, 1
mov ecx, resultMsg
mov edx, lenRes
int 0x80

; Print result
mov eax, 4
mov ebx, 1
mov ecx, res
mov edx, 1
int 0x80

; Print newline
mov eax, 4
```

```
    mov ebx, 1
    mov ecx, newline
    mov edx, 1
    int 0x80

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

```
kaizen@kaizen:~$ gedit mul.asm
kaizen@kaizen:~$ nasm -f elf32 mul.asm -o mul.o
kaizen@kaizen:~$ ld -m elf_i386 -s -o mul mul.o
kaizen@kaizen:~$ ./mul
```

```
kaizen@kaizen:~$ ./mul
Enter first number: 3
Enter second number: 2
Result = 6
```

Division.asm

```
section .data
    msg1 db "Enter dividend: ",0
    len1 equ $-msg1
    msg2 db "Enter divisor: ",0
    len2 equ $-msg2
    resultMsg db "Quotient = ",0
    lenRes equ $-resultMsg
    newline db 10,0
```

```
section .bss
    num1 resb 2
    num2 resb 2
    res  resb 2

section .text
    global _start

_start:
    ; Ask for first number
    mov eax, 4
    mov ebx, 1
    mov ecx, msg1
    mov edx, len1
    int 0x80

    ; Read input
    mov eax, 3
    mov ebx, 0
    mov ecx, num1
    mov edx, 2
    int 0x80

    ; Ask for second number
    mov eax, 4
    mov ebx, 1
    mov ecx, msg2
    mov edx, len2
    int 0x80
```

```
; Read input
mov eax, 3
mov ebx, 0
mov ecx, num2
mov edx, 2
int 0x80

; Convert ASCII to number
mov al, [num1]
sub al, '0'
mov bl, [num2]
sub bl, '0'

; Divide
xor ah, ah    ; clear upper byte
div bl        ; AL = AL / BL, AH = remainder
add al, '0'
mov [res], al

; Print "Quotient = "
mov eax, 4
mov ebx, 1
mov ecx, resultMsg
mov edx, lenRes
int 0x80

; Print result
mov eax, 4
mov ebx, 1
```

```
mov ecx, res

mov edx, 1

int 0x80


; Print newline

mov eax, 4

mov ebx, 1

mov ecx, newline

mov edx, 1

int 0x80

; Exit

mov eax, 1

xor ebx, ebx

int 0x80
```

```
kaizen@kaizen:~$ gedit div.asm
kaizen@kaizen:~$ nasm -f elf32 div.asm -o div.o
kaizen@kaizen:~$ ld -m elf_i386 -s -o div div.o
```

```
kaizen@kaizen:~$ ./div
Enter dividend: 3
Enter divisor: 9
Quotient = 0
```