# Solutions to Lab-06R

# Activity 1

## **Assembly Language Code**

[org 0x0100]

jmp start

message: db 'hello' ; string to be printed

length: dw 5 ; length of the string

message2: db 'my name is fatima' ; string to be printed

length2: dw 17 ; length of the string

message3: db 'thank you' ; string to be printed

length3: dw 9 ; length of the string

; subroutine to clear the screen

clrscr:

push es

push ax

push di

mov ax, 0xb800

mov es, ax ; point es to video base

mov di, 0 ; point di to top left column

nextloc: mov word [es:di], 0x0720 ; clear next char on screen

add di, 2 ; move to next screen location

cmp di, 4000 ; has the whole screen cleared

jne nextloc ; if no clear next position

pop di

pop ax

pop es

ret

; subroutine to print a string at top left of screen

; takes x position, y position, string attribute, address of string

; and its length as parameters

printstr: push bp

mov bp, sp

push es

push ax

push cx

push si

push di

mov ax, 0xb800

mov es, ax ; point es to video base

mov al, 80 ; load al with columns per row

mul byte [bp+10] ; multiply with y position

add ax, [bp+12] ; add x position

shl ax, 1 ; turn into byte offset

mov di, ax ; point di to required location

mov si, [bp+6] ; point si to string

mov cx, [bp+4] ; load length of string in cx

mov ah, [bp+8] ; load attribute in ah

nextchar: mov al, [si] ; load next char of string

mov [es:di], ax ; show this char on screen

add di, 2 ; move to next screen location

add si, 1 ; move to next char in string

loop nextchar ; repeat the operation cx times

pop di

pop si

pop cx

pop ax

pop es

pop bp

ret 10

start:

call clrscr ; call the clrscr subroutine

mov ax, 5

push ax ; push x position

mov ax, 5

push ax ; push y position

mov ax, 0x0067 ; blue on black attribute

push ax ; push attribute

mov ax, message

push ax ; push address of message

push word [length] ; push message length

call printstr ; call the printstr subroutine

start2:

mov ax, 10

push ax ; push x position

mov ax, 10

push ax ; push y position

mov ax, 0x0059 ; blue on black attribute

push ax ; push attribute

mov ax, message2

push ax ; push address of message

push word [length2] ; push message length

call printstr ; call the printstr subroutine

start3:

mov ax, 20

push ax ; push x position

mov ax, 20

push ax ; push y position

mov ax, 0x0042 ; blue on black attribute

push ax ; push attribute

mov ax, message3

push ax ; push address of message

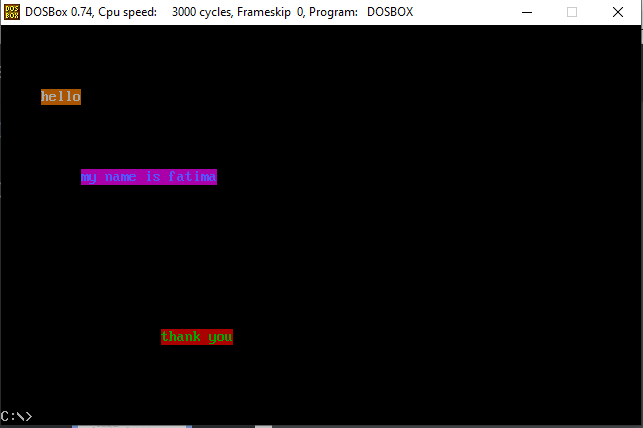
push word [length3] ; push message length

call printstr ; call the printstr subroutine

mov ax, 0x4c00 ; terminate program

int 0x21

## **Debugging Screenshots**



# Activity 2

## **Assembly Language Code**

[org 0x0100]

jmp start

top: dw 10 ;Starting Row

bottom: dw 20 ;Ending Row

left: dw 30 ;Starting Column

right: dw 60 ;Ending Column

start: call clrscr

push word [top]

push word [bottom]

push word [left]

push word [right]

call drawrect

end: mov ax, 0x4c00

int 21h

clrscr: mov ax, 0xb800

mov es, ax ;Loading the video memory

xor di,di

mov ax,0x0720

mov cx,2000

cld

rep stosw

ret

drawrect: push bp

mov bp, sp

pusha

; bp + 4 = right

; bp + 6 = left

; bp + 8 = bottom

; bp + 10 = top

;Calculating the top left position of the rectangle

mov al, 80

mul byte [bp + 10]

add ax, [bp + 6]

shl ax, 1

mov di, ax

push di ;Saving for later use

mov ah, 0x07 ;Storing the attribute

;Calculating the width of the rectangle

mov cx, [bp + 4]

sub cx, [bp + 6]

push cx ;Saving for later use

mov al, '+'

loop1: rep stosw

pop bx

pop di

push bx

dec bx

shl bx, 1

add di, 160

;Calculating the height of the rectangle

mov cx, [bp + 8]

sub cx, [bp + 10]

sub cx, 2 ;Excluding the top and bottom row

mov al, '|'

loop2: mov si, di

mov word [es:si], ax

add si, bx

mov word [es:si], ax

sub si, bx

add di, 160

loop loop2

pop cx

mov al, '-'

loop3: rep stosw

return: popa

pop bp

ret 8

## **Debugging Screenshots**

