**National University of Computer and Emerging Sciences**

**Lab Manual**

**Computer Organization and Assembly Language**



**Lab 08**

|  |  |
| --- | --- |
| **Instructor** | Hazoor Ahmad |
| **Class** | A |
| **Semester** | Fall 2022 |

**Name Umamah Hussain  
 Roll no.** 21L-1858

Fast School of Computing

FAST-NU, Lahore, Pakistan

# Objectives

* String Processing
* STOS Example – Clearing the Screen
* LODS Example – String Printing
* SCAS Example – String Length
* MOVS Example – Screen Scrolling

**Contents**

|  |  |  |
| --- | --- | --- |
| Objectives |  | 2 |
| ACTIVITY 1: | [20 Marks] | 2 |
| ACTIVITY 2: | [20 Marks] | 2 |
| ACTIVITY 3: | [40+20 Marks] | 2 |

REFERENCES 3

**Note for all questions**: You can make as many memory variables, subroutines as you need. Must read all the manual before starting.

## ACTIVITY 1: [20 Marks]

Write a subroutine which uses nested loops for creating a delay of around 3-5 secs**.**

**Note**: Do not use any special commands or interrupts for creating any delay.

**Solution:**

[org 0x0100]

jmp start

message: db "Umamahhh"

length: dw 8

delay:

push cx

mov cx,50

d1:

push cx

mov cx,0xFFFF

d2:

loop d2

pop cx

loop d1

pop cx

ret

clrscr:

push es

push ax

push di

mov ax,0xb800

mov es,ax

mov di,0

nextloc:

mov word[es:di],0x720

add di,2

cmp di,4000

jne nextloc

pop di

pop ax

pop es

ret

printscr:

push bp

mov bp,sp

push es

push ax

push cx

push si

push di

mov ax,0xb800

mov es,ax

mov al,80

mul byte [bp+10]

add ax,[bp+12]

shl ax,1

mov di,ax

mov si,[bp+6]

mov cx,[bp+4]

mov ah,[bp+8]

nextchar:

mov al,[si]

mov [es:di],ax

add di,2

add si,1

loop nextchar

pop di

pop si

pop cx

pop ax

pop es

pop bp

ret 10

start :

call delay

call clrscr

mov ax,10

push ax

mov ax,10

push ax

mov ax,3

push ax

mov ax,message

push ax

push word[length]

call printscr

mov ax,0x4c00

int 0x21

**Screenshot:**

## ACTIVITY 2: [20 Marks]

Practice the following examples from **Chapter 7** and display their outputs in report:

**STOS Example – Clearing the Screen**

**Solution:**

[org 0x0100]

jmp start

clrscr:

push es

push ax

push cx

push di

mov ax, 0xb800

mov es, ax

xor di, di

mov ax, 0x0720

mov cx, 2000

cld

rep stosw

pop di

pop cx

pop ax

pop es

ret

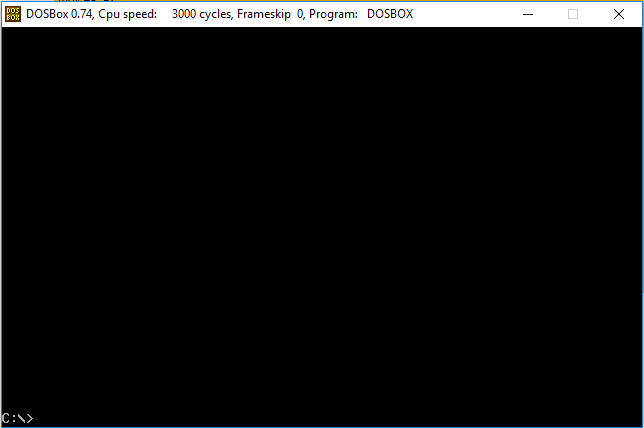
start:

call clrscr

mov ax, 0x4c00

int 0x21

**Screenshot:**



**LODS Example – String Printing**

**Solution:**

[org 0x0100]

jmp start

message: db 'hello world'

length: dw 11

clrscr:

push es

push ax

push cx

push di

mov ax, 0xb800

mov es, ax

xor di, di

mov ax, 0x0720

mov cx, 2000

cld

rep stosw

pop di

pop cx

pop ax

pop es

ret

printstr: push bp

mov bp, sp

push es

push ax

push cx

push si

push di

mov ax, 0xb800

mov es, ax

mov al, 80

mul byte [bp+10]

add ax, [bp+12]

shl ax, 1

mov di,ax

mov si, [bp+6]

mov cx, [bp+4]

mov ah, [bp+8]

cld

nextchar:

lodsb

stosw

loop nextchar

pop di

pop si

pop cx

pop ax

pop es

pop bp

ret 10

start:

call clrscr

mov ax, 30

push ax

mov ax, 20

push ax

mov ax, 1

push ax

mov ax, message

push ax

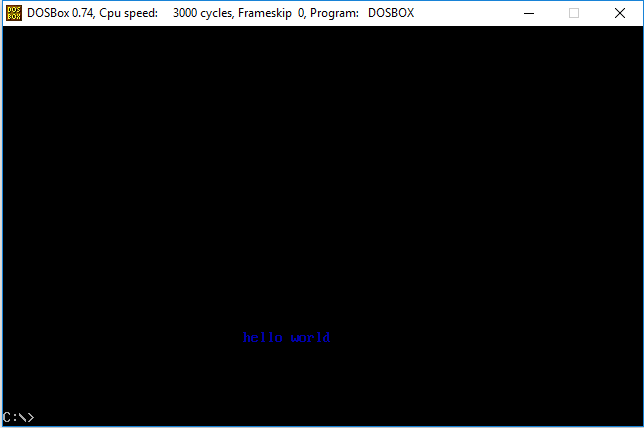
push word [length]

call printstr

mov ax, 0x4c00

int 0x21

**Screenshot:**



**SCAS Example – String Length**

**Solution:**

[org 0x0100]

jmp start

message: db 'hello world', 0

clrscr:

push es

push ax

push cx

push di

mov ax, 0xb800

mov es, ax

xor di, di

mov ax, 0x0720

mov cx, 2000

cld

rep stosw

pop di

pop cx

pop ax

pop es

ret

printstr: push bp

mov bp, sp

push es

push ax

push cx

push si

push di

push ds

pop es

mov di, [bp+4]

mov cx, 0xffff

xor al, al

repne scasb

mov ax, 0xffff

sub ax, cx

dec ax

jz exit

mov cx, ax

mov ax, 0xb800

mov es, ax

mov al, 80

mul byte [bp+8]

add ax, [bp+10]

shl ax, 1

mov di,ax

mov si, [bp+4]

mov ah, [bp+6]

cld

nextchar: lodsb

stosw

loop nextchar

exit: pop di

pop si

pop cx

pop ax

pop es

pop bp

ret 8

start: call clrscr

mov ax, 30

push ax

mov ax, 20

push ax

mov ax, 1

push ax

mov ax, message

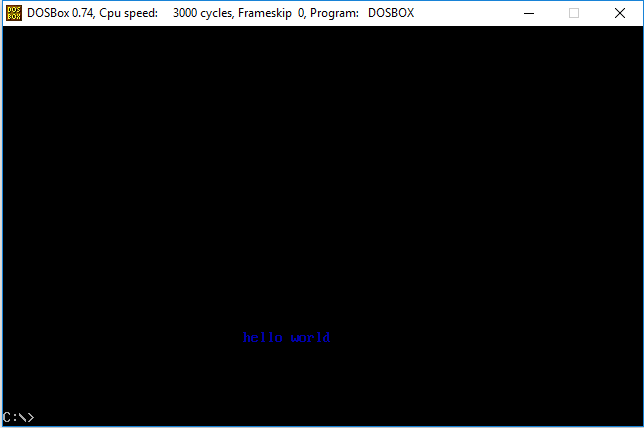
push ax

call printstr

mov ax, 0x4c00

int 0x21

**Screenshot:**



**MOVS Example – Screen Scrolling**

**Solution:**

[org 0x0100]

jmp start

scrollup:

push bp

mov bp,sp

push ax

push cx

push si

push di

push es

push ds

mov ax, 80

mul byte [bp+4]

mov si, ax

push si

shl si, 1

mov cx, 2000

sub cx, ax

mov ax, 0xb800

mov es, ax

mov ds, ax

xor di, di

cld

rep movsw

mov ax, 0x0720

pop cx

rep stosw

pop ds

pop es

pop di

pop si

pop cx

pop ax

pop bp

ret 2

start:

mov ax,5

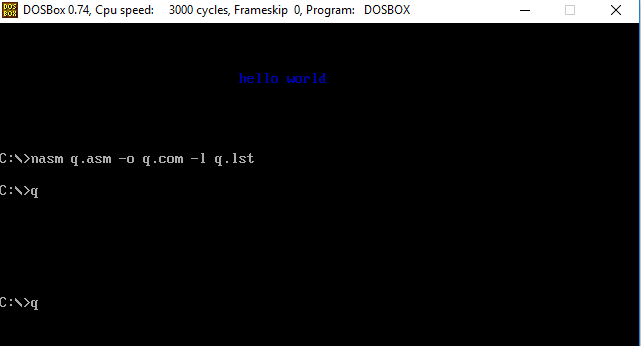
push ax

call scrollup

mov ax, 0x4c00

int 0x21

**Screenshot:**

****

## ACTIVITY 3: [40+20 Marks]

1. Use the subroutines developed in **Activity 1** and **Activity 2** to create an animation (of Start or End Screen) which shows the following information about you (every message with delay) as shown on the following figure and in the [YouTube link [1].](https://youtu.be/Anphs9zoP_c) **(40 Marks)**

**Note**: Do not pass string length parameter calculate string length automatically.

1. Record a video of your animation through Mobile or any screen recorder and submit along with the report. **(20 Marks)**

**Solution:**

[org 0x0100]

jmp start

m1: db 'Name: Umamah Hussain', 0

m2: db 'Institution: FAST School of Computing', 0

m3: db 'Batch: 2021', 0

m4: db 'Roll: 1858', 0

m5: db 'Email ID: l21858@lhr.nu.edu.pk', 0

m6: db 'Address: Lahore, Pakistan', 0

m7: db 'Features: Honest, Punctual', 0

delay:

push cx

mov cx,50

d1:

push cx

mov cx,0xFFFF

d2:

loop d2

pop cx

loop d1

pop cx

ret

clrscr:

push es

push ax

push cx

push di

mov ax, 0xb800

mov es, ax

xor di, di

mov ax, 0x0720

mov cx, 2000

cld

rep stosw

pop di

pop cx

pop ax

pop es

ret

printstr: push bp

mov bp, sp

push es

push ax

push cx

push si

push di

push ds

pop es

mov di, [bp+4]

mov cx, 0xffff

xor al, al

repne scasb

mov ax, 0xffff

sub ax, cx

dec ax

jz exit

mov cx, ax

mov ax, 0xb800

mov es, ax

mov al, 80

mul byte [bp+8]

add ax, [bp+10]

shl ax, 1

mov di,ax

mov si, [bp+4]

mov ah, [bp+6]

cld

nextchar: lodsb

stosw

loop nextchar

exit: pop di

pop si

pop cx

pop ax

pop es

pop bp

ret 8

start:

call clrscr

mov ax, 20

push ax

mov ax, 10

push ax

mov ax, 4

push ax

mov ax, m1

push ax

call printstr

call delay

mov ax, 20

push ax

mov ax, 11

push ax

mov ax, 4

push ax

mov ax, m2

push ax

call printstr

call delay

mov ax, 20

push ax

mov ax, 12

push ax

mov ax, 4

push ax

mov ax, m3

push ax

call printstr

call delay

mov ax, 20

push ax

mov ax, 13

push ax

mov ax, 4

push ax

mov ax, m4

push ax

call printstr

call delay

mov ax, 20

push ax

mov ax, 14

push ax

mov ax, 4

push ax

mov ax, m5

push ax

call printstr

call delay

mov ax, 20

push ax

mov ax, 15

push ax

mov ax, 4

push ax

mov ax, m6

push ax

call printstr

call delay

mov ax, 20

push ax

mov ax, 16

push ax

mov ax, 4

push ax

mov ax, m7

push ax

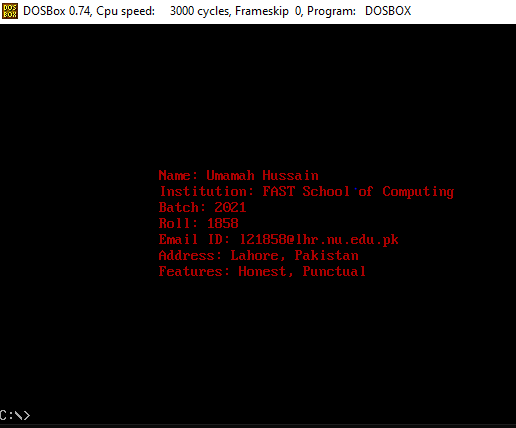
call printstr

call delay

mov ax, 0x4c00

int 0x21

**Screenshot:**



## REFERENCES

1. <https://youtu.be/Anphs9zoP_c>
2. http://www.dosbox.com/download.php?main=1
3. <http://sourceforge.net/projects/nasm>
4. <http://www.nasm.us/>
5. [http://www.programmersheaven.com/download/21643/download.aspx (AFD)](http://www.programmersheaven.com/download/21643/download.aspx)