Jahangirnagar University (JU)

Institute of Information Technology

Lab Report-4

Assembly Language

Name: Md. Shakil Hossain **Roll:** 2023

Experiment 1:

Case conversion

i. upper case to lower case and vice versa

Algorithm:

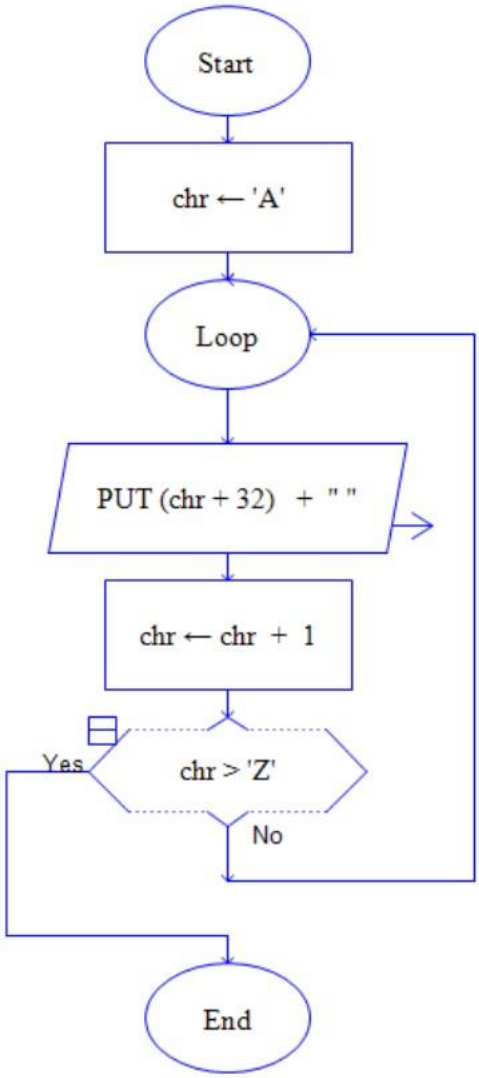
Upper case to lower case

1. Load the input character into a register.
2. Compare the input character with the ASCII value of 'A' (41h).
3. If the input character is less than 'A', jump to step 7.
4. Compare the input character with the ASCII value of 'Z' (5Ah).
5. If the input character is greater than 'Z', jump to step 7.
6. Add 32 (20h) to the input character to convert it to lower case.
7. Store the converted character in a memory location or register.

Lower case to Upper case

1. Load the input character into a register.
2. Compare the input character with the ASCII value of 'a' (61h).
3. If the input character is less than 'a', jump to step 7.
4. Compare the input character with the ASCII value of 'z' (7Ah).
5. If the input character is greater than 'z', jump to step 7.
6. Subtract 32 (20h) from the input character to convert it to upper case.
7. Store the converted character in a memory location or register.

Flow Chart



Program Source Code:

Upper case to lower case

```
org 100h
.model small
.data
    msg1 db 13,10, "Enter upper case: $"
    msg2 db 13,10, "lower case: $"
.code
main proc

   mov ax,@data
   mov dx,ax

   mov dx,offset msg1

   mov ah,9
   int 21h

   mov ah,1
   int 21h

   mov bl,al

   add bl,32

   mov ax,@data
   mov dx,ax

   mov dx,offset msg2

   mov ah,9
   int 21h

   mov dl,bl

   mov ah,2
   int 21h

   mov ah,4ch
```

```
    int 21h


  main endp
end main

ret
```

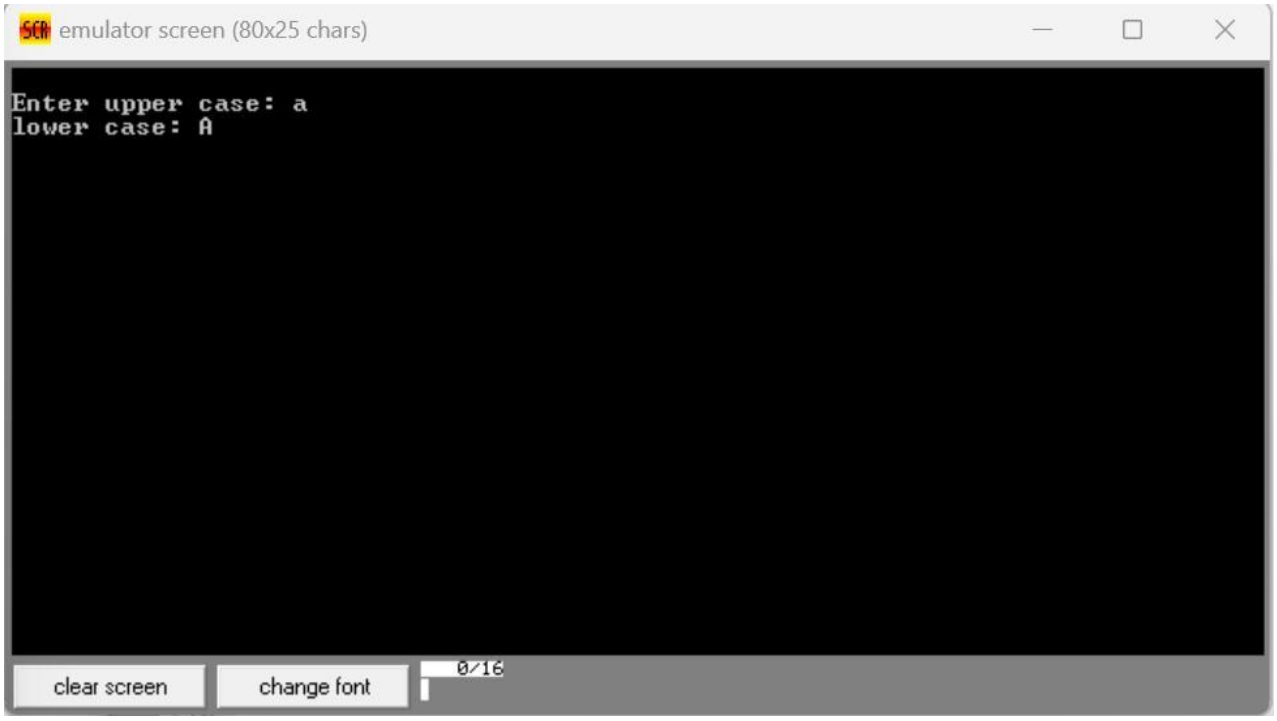
Lower case to Upper case

```
org 100h
.model small
.data
     msg1 db 13,10, "Enter lower case: $"
     msg2 db 13,10, "Upper case: $"
.code
main proc

    mov ax,@data
    mov dx,ax

    mov dx,offset msg1

    mov ah,9
    int 21h

    mov ah,1
    int 21h

    mov bl,al

    sub bl,32

    mov ax,@data
    mov dx,ax

    mov dx,offset msg2
```

```
    mov ah,9
    int 21h

    mov dl,bl

    mov ah,2
    int 21h

    mov ah,4ch
    int 21h


  main endp
end main

ret
```

Sample Input: **A**
Sample Output: **a**

Sample Input: **a**
Sample Output: **A**



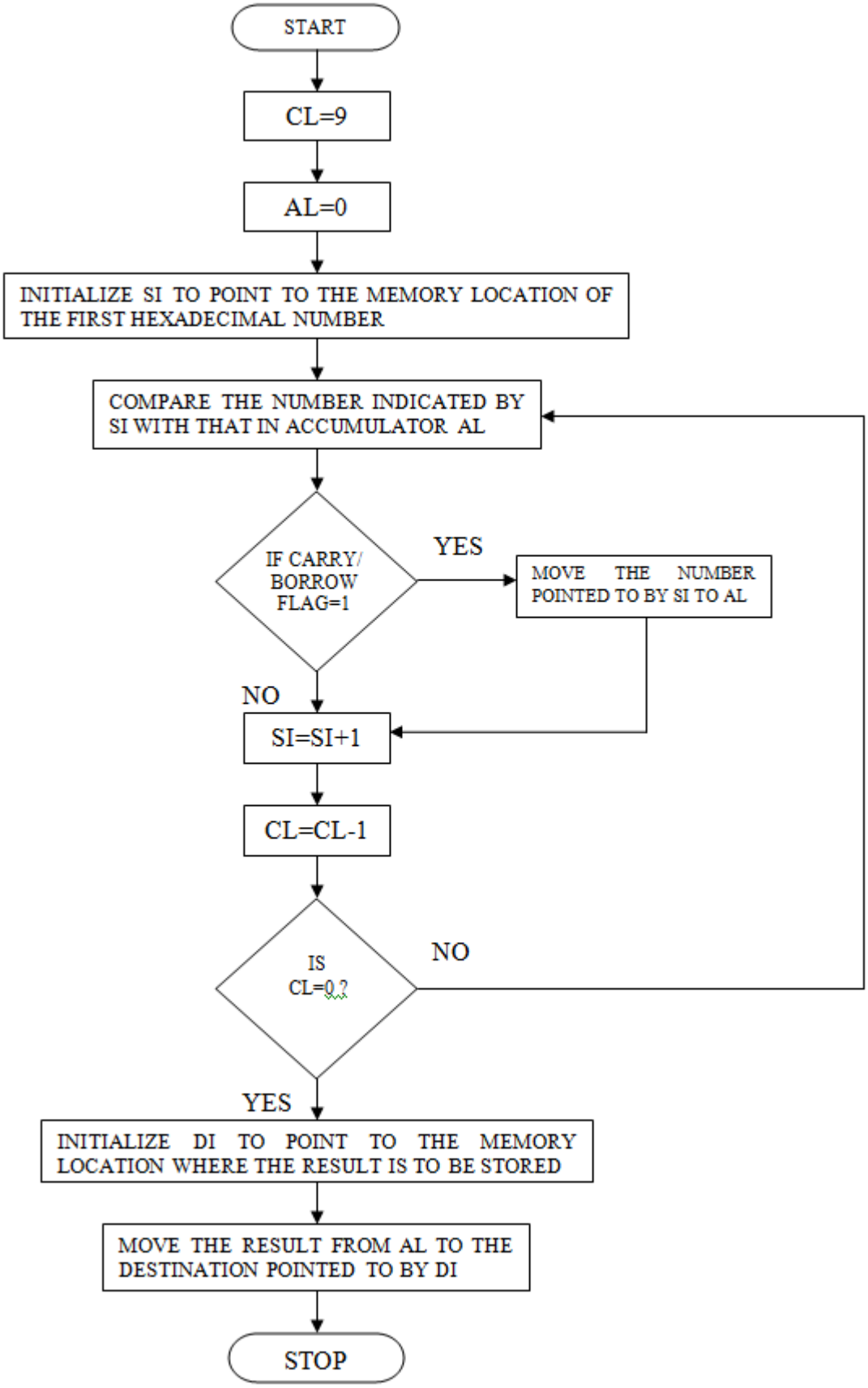
Experiment 2:

Compare three digits and find the biggest number

Algorithm:

1. Load the first digit into a register.
2. Load the second digit into another register.
3. Compare the first and second digits.
4. If the first digit is less than the second digit, store the second digit in the first digit's register.
5. Load the third digit into another register.
6. Compare the first digit with the third digit.
7. If the first digit is less than the third digit, store the third digit in the first digit's register.
8. The first digit's register now contains the largest of the three digits.

Flow Chart:

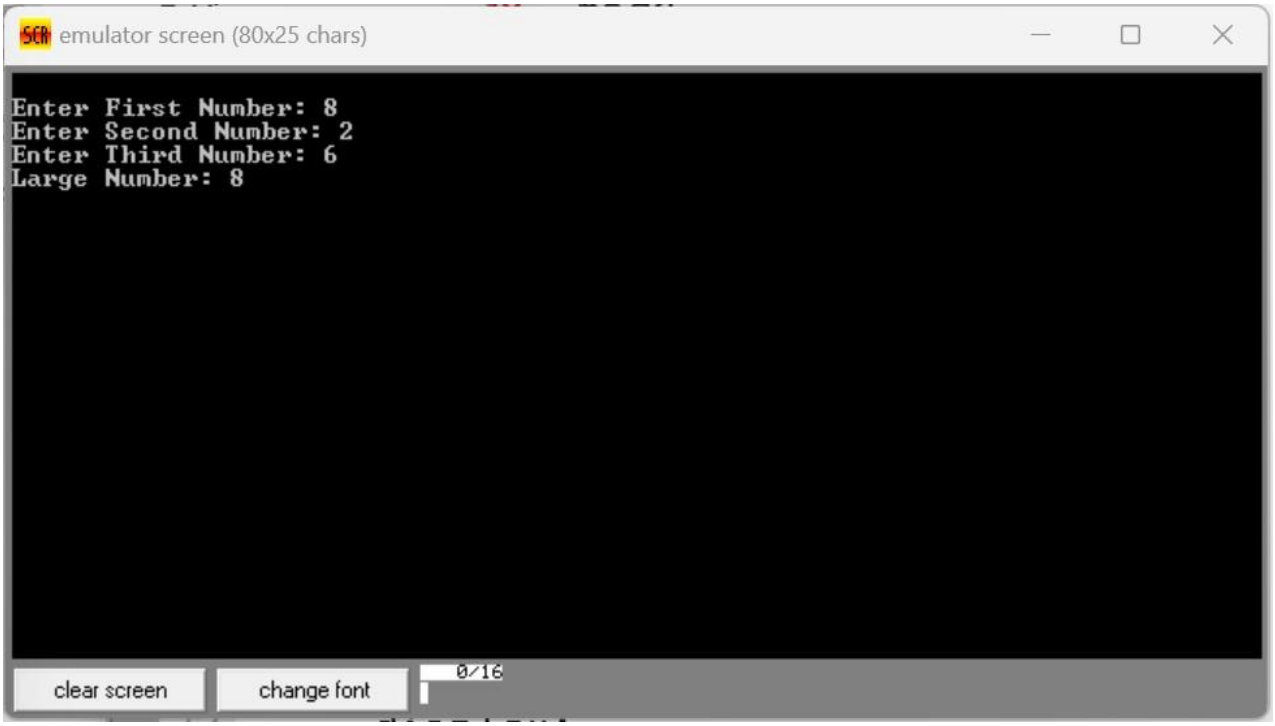

START
CL=9
AL=0
INITIALIZE SI TO POINT TO THE MEMORY LOCATION OF THE FIRST HEXADECIMAL NUMBER
COMPARE THE NUMBER INDICATED BY SI WITH THAT IN ACCUMULATOR AL
IF CARRY/ BORROW FLAG=1
YES
MOVE THE NUMBER POINTED TO BY SI TO AL
NO
SI=SI+1
CL=CL-1
IS CL=0 ?
NO
YES
INITIALIZE DI TO POINT TO THE MEMORY LOCATION WHERE THE RESULT IS TO BE STORED
MOVE THE RESULT FROM AL TO THE DESTINATION POINTED TO BY DI
STOP

Program Source Code:

```
org 100h
.model small
.data

    msg1 db 10,13,"Enter First Number : $"
    msg2 db 10,13,"Enter Second Number : $"
    msg3 db 10,13,"Enter Third Number : $"
    msg4 db 10,13,"Large Number : $"

    num1 db ?
    num2 db ?
    num3 db ?
.code
main proc

    mov ax, @data
    mov ds, ax

    lea dx, msg1
    mov ah, 9
    int 21h

    mov ah, 1
    int 21h

    mov num1, al

    lea dx, msg2
    mov ah, 9
    int 21h

    mov ah, 1
    int 21h


    mov num2, al

    lea dx, msg3
    mov ah, 9
    int 21h

    mov ah, 1
    int 21h
```

```
    mov num3, al

    lea dx, msg4
    mov ah, 9
    int 21h

    mov bl, num1
    cmp bl, num2
    jng number2

    cmp bl, num3
    jng number3

    mov dl, num1
    jmp display

    number2:
    mov bl, num2
    cmp bl, num3
    jng number3


    number3:
    mov dl, num3

    display:
    mov ah, 2
    int 21h

main endp
ret
```

Sample Input: 8 , 2 , 6
Sample Output: 8



THE END