DIGITAL ASSIGNMENT : PALINDROME AND BIOS INPUT/OUTPUT

CSE2006 - MICROPROCESSOR AND INTERFACING(L39-40)[MRS SHOBHA REKH]

OCTOBER 29, 2022

ANIRUDH VADERA
20BCE2940

QUESTION 1:

Use the BIOS interrupts and read a character from the keyboard and display the same on the computer screen.

Explanation:

BIOS Interrupts used:

To Read a Character that is entered from Keyboard:

INT 21h / **AH=1** - read character from standard input, with echo, result is stored in **AL**.
if there is no character in the keyboard buffer, the function waits until any key is pressed.

Example:

```
mov ah, 1
int 21h
```

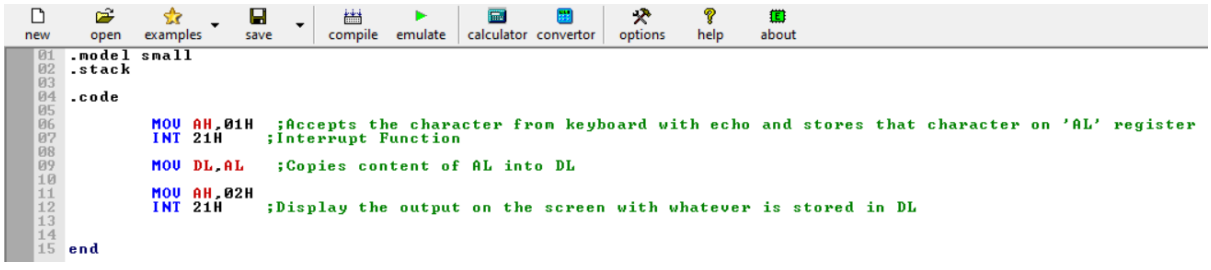
To Display a character that was entered:

INT 21h / **AH=2** - write character to standard output.
entry: **DL** = character to write, after execution **AL = DL**.

Example:

```
mov ah, 2
mov dl, 'a'
int 21h
```

Code Screenshot:



Assembly Language Code:

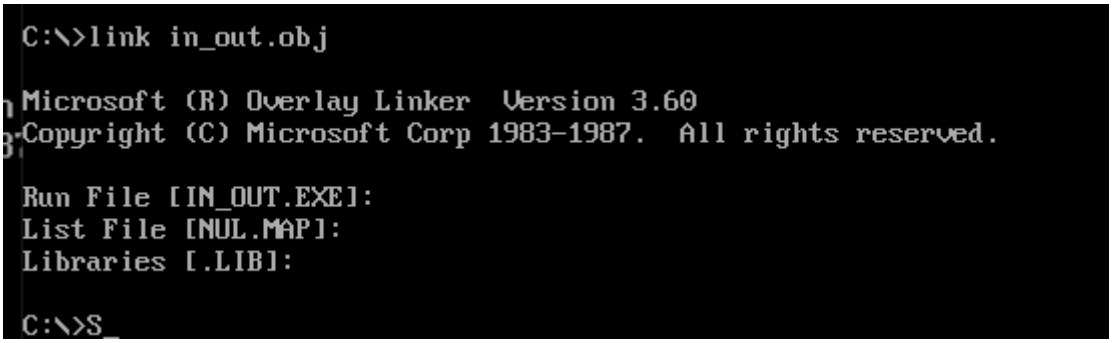
```
.model small

.stack
```

```
.code

    MOV AH,01H  ;Accepts the character from keyboard with echo and stores that
character on 'AL' register

    INT 21H    ;Interrupt Function


    MOV DL,AL   ;Copies content of AL into DL


    MOV AH,02H

    INT 21H    ;Display the output on the screen with whatever is stored in DL



end
```

Execution Proof:

Command: masm in_out.asm

Command: link in_out.obj



Command: debug in_out.exe

```
C:\>debug in_out.exe
-S
```

Giving Inputs:

Input Given as: **h(A string literal)**

```
C:\>debug in_out.exe
-t

AX=01FF  BX=0000  CX=000A  DX=0000  SP=0400  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=076B  CS=076A  IP=0002   NV UP EI PL NZ NA PO NC
076A:0002 CD21          INT     21
-t

AX=01FF  BX=0000  CX=000A  DX=0000  SP=03FA  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=076B  CS=F000  IP=14A0   NV UP DI PL NZ NA PO NC
F000:14A0 FB            STI
-t

AX=01FF  BX=0000  CX=000A  DX=0000  SP=03FA  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=076B  CS=F000  IP=14A1   NV UP EI PL NZ NA PO NC
F000:14A1 FE38          ???     [BX+SI]                         DS:0000=CD
-t
h
```

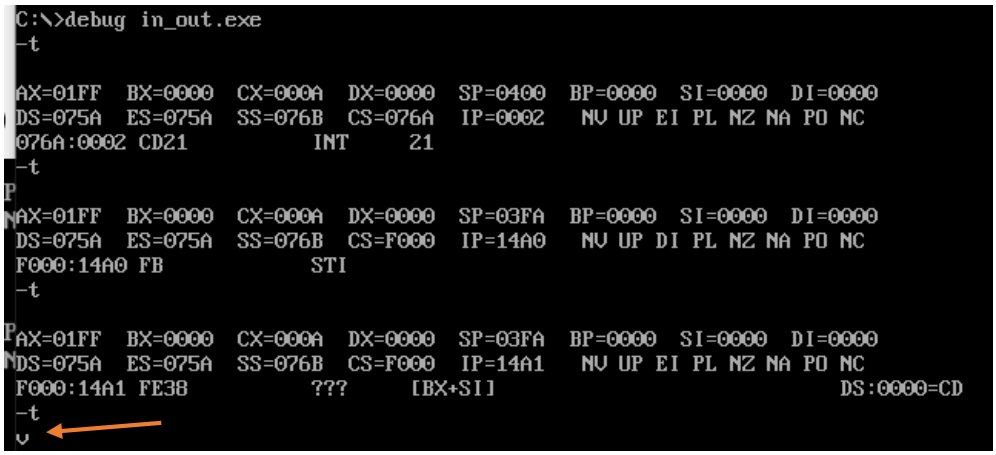
The Output is:

We get the output as: **h** (Hence Code is working fine)

```
AX=0268  BX=0000  CX=000A  DX=0068  SP=03FA  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=076B  CS=F000  IP=14A0   NV UP DI PL NZ NA PO NC
F000:14A0 FB            STI
-t

AX=0268  BX=0000  CX=000A  DX=0068  SP=03FA  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=076B  CS=F000  IP=14A1   NV UP EI PL NZ NA PO NC
F000:14A1 FE38          ???     [BX+SI]                         DS:0000=CD
-t
h
AX=0268  BX=0000  CX=000A  DX=0068  SP=03FA  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=076B  CS=F000  IP=14A5   NV UP EI PL NZ NA PO NC
F000:14A5 CF            IRET
-S_
```

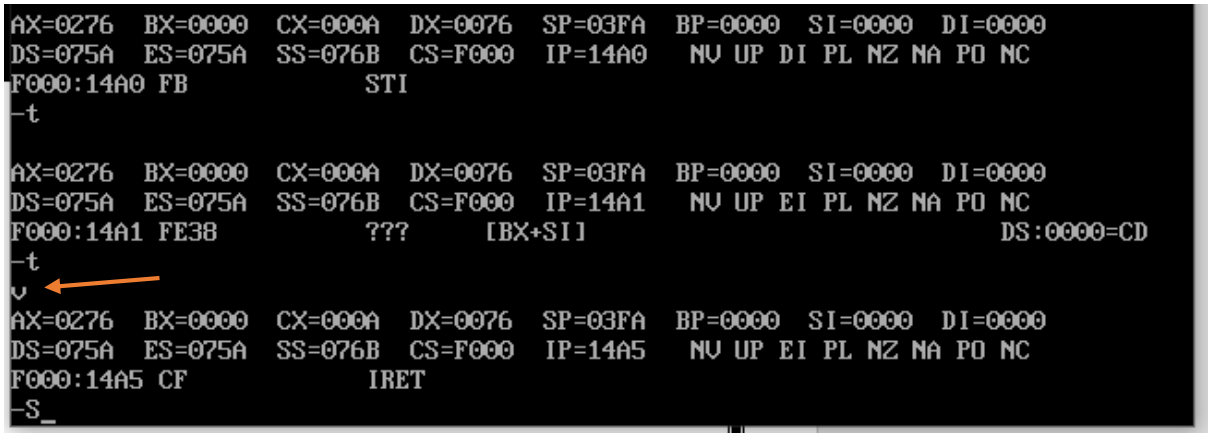
Giving Inputs:

Input Given as: **v**



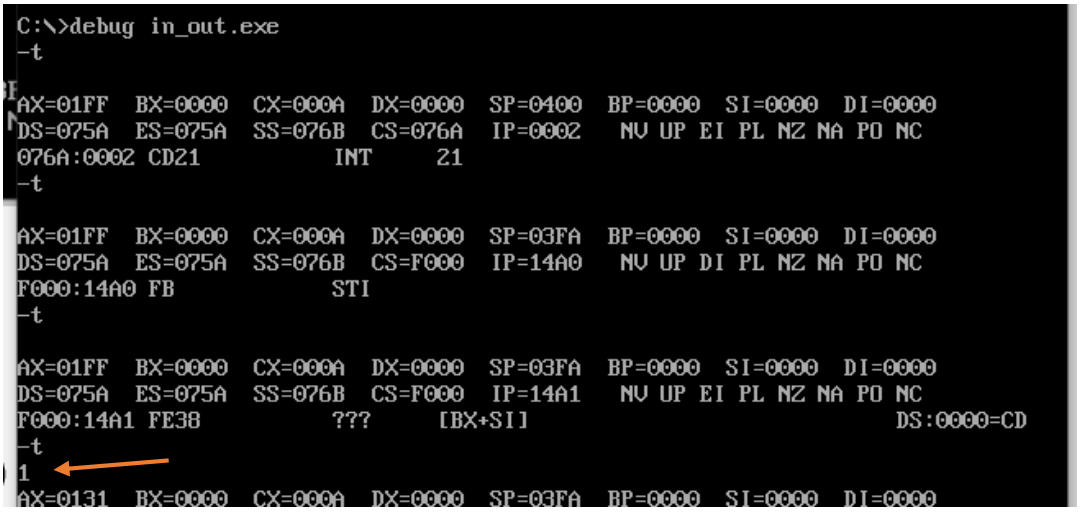
The Output is:

We get the output as: **v** (Hence Code is working fine)



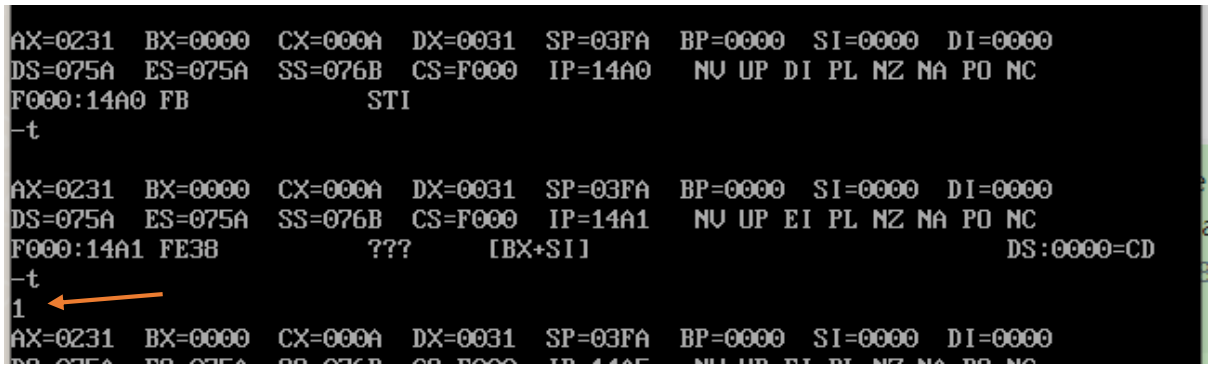
Giving Inputs:

Input Given as: **1(A Number)**



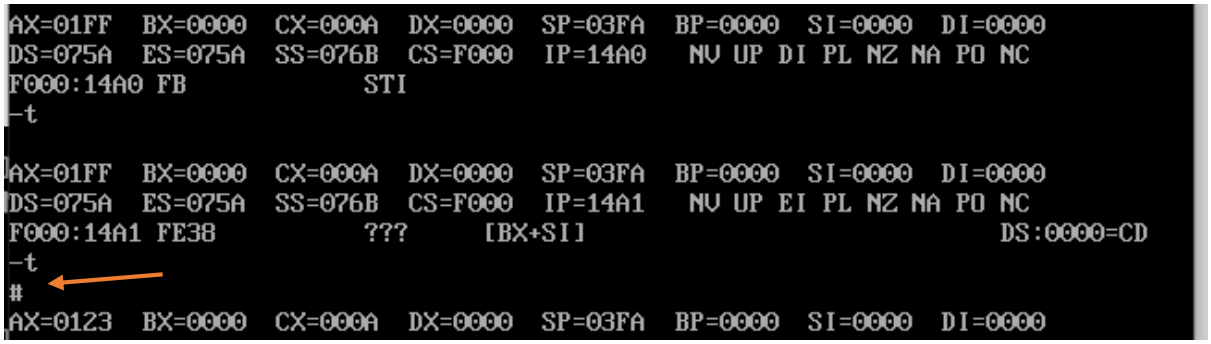
The Output is:

We get the output as: **1** (Hence Code is working fine)



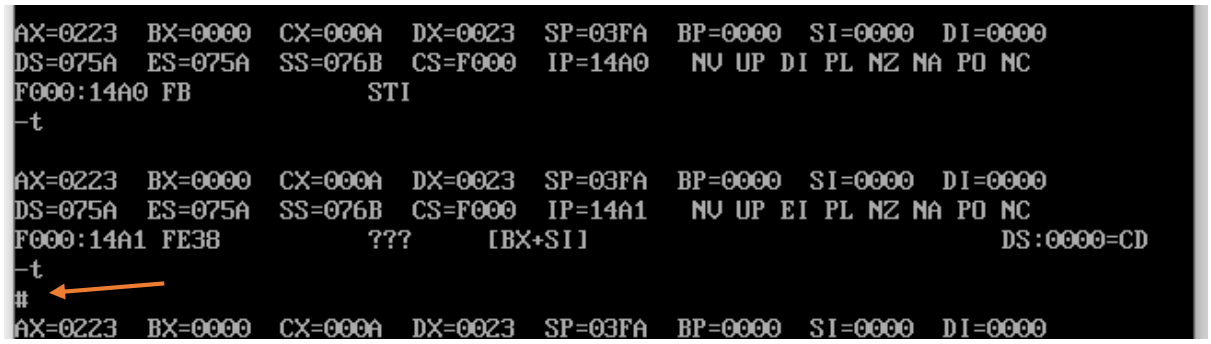
Giving Inputs:

Input Given as: **#(A special Character)**



The Output is:

We get the output as: **#** (Hence Code is working fine)



QUESTION 2:

Store a string in the program. Find if the stored string is a Palindrome or not.

Display "It is a Palindrome "or display "It is not a Palindrome". Use BIOS interrupts to display string on the monitor.

Explanation:

To Enter a string by user:

INT 21h / **AH=0Ah** - input of a string to **DS:DX**, fist byte is buffer size, second byte is number of chars actually read. this function does **not** add '$' in the end of string.

Example:

```
org 100h
mov dx, offset buffer
mov ah, 0ah
int 21h
jmp next
buffer db 10,?, 10 dup(' ')
```

To Display a string that was entered:

INT 21h / **AH=9** - output of a string at **DS:DX**. String must be terminated by '**$**'.

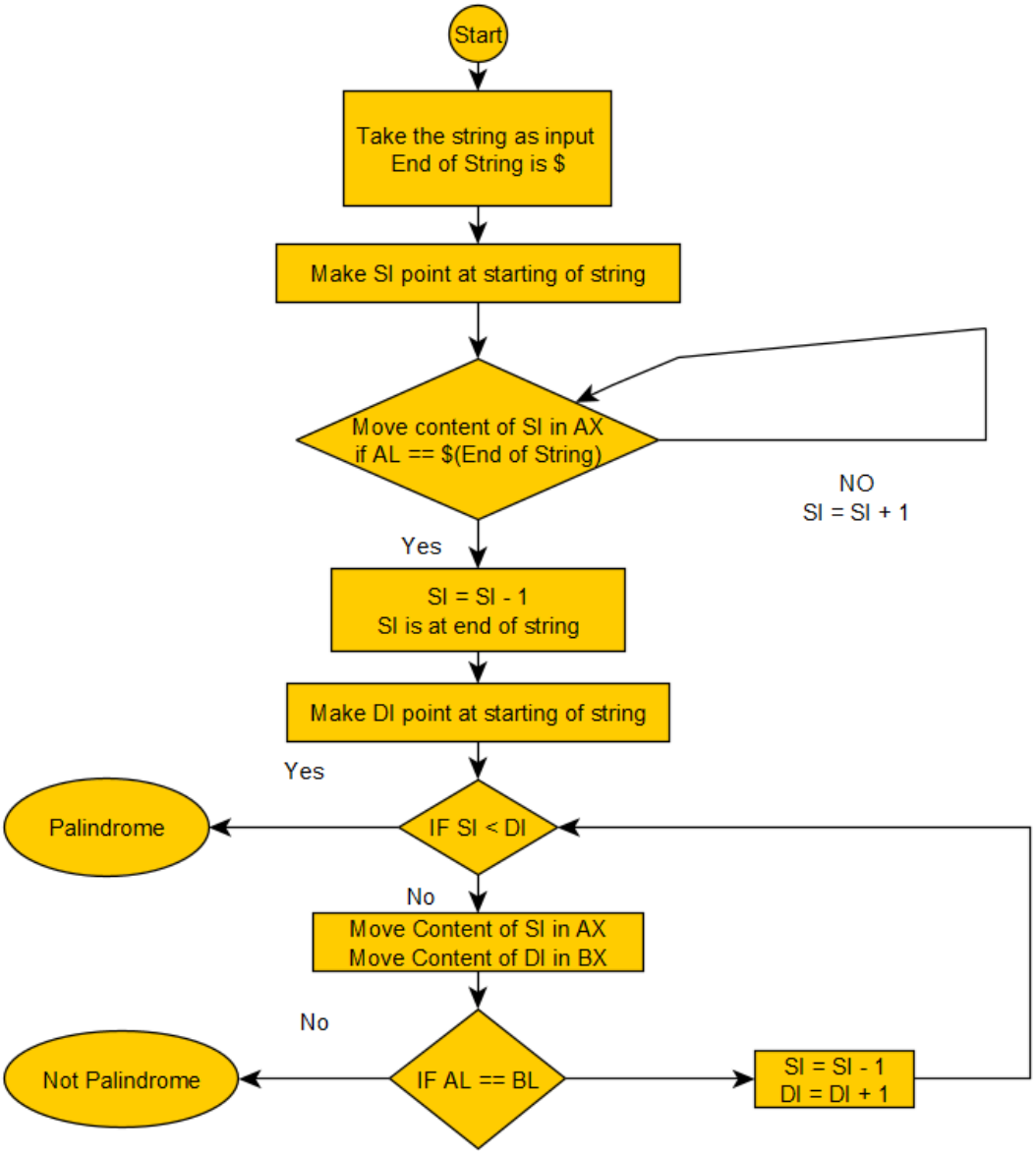
Example:

```
org 100h
mov dx, offset msg
mov ah, 9
int 21h
ret
msg db "hello world $"
```

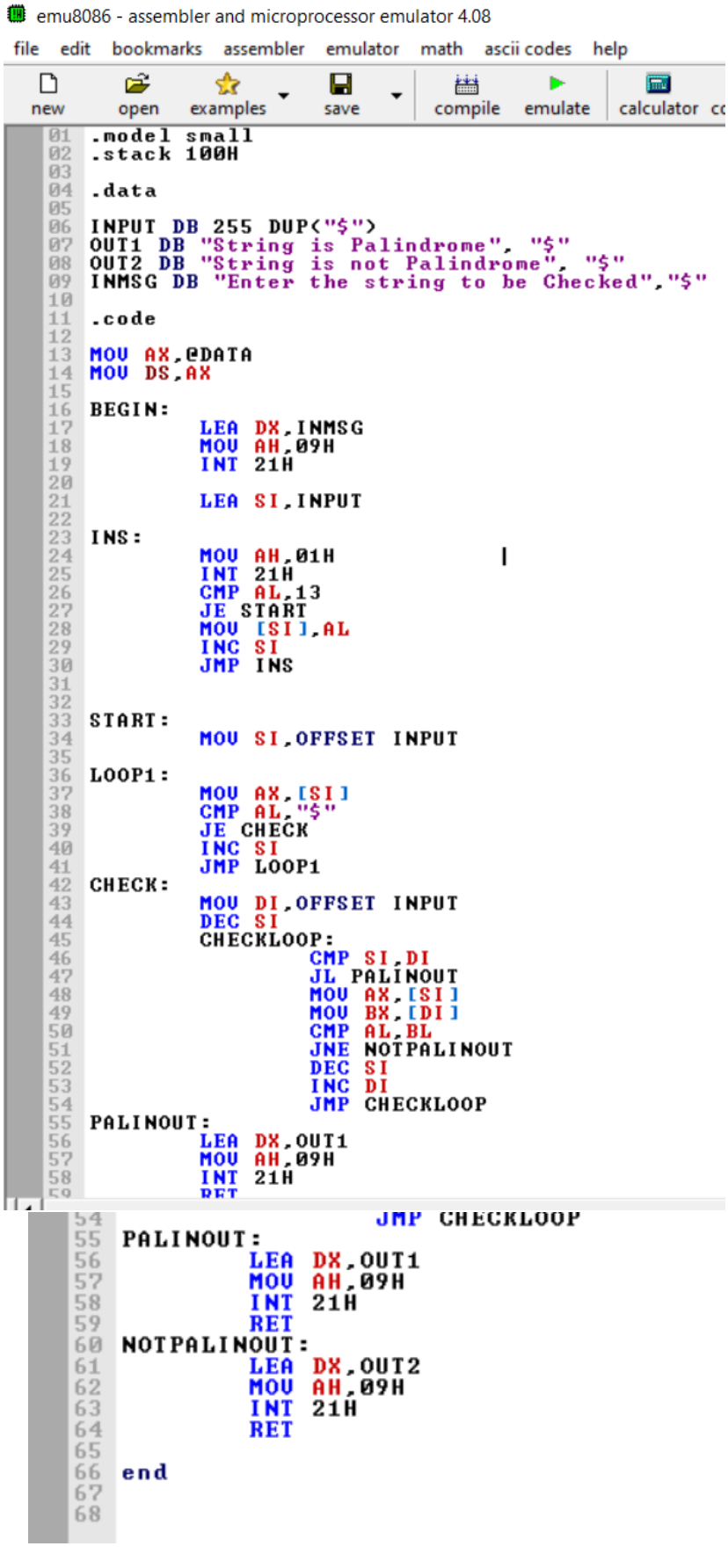
In Assembly Language:

1. Create a string
2. Traverse to the end of the string
3. Get the address of the end of the string, SI
4. Load the starting address of the string, DI
5. Compare the value stored at the address
6. Increment the pointer, DI
7. Decrements the pointer, SI
8. Compare again the value stored at SI and DI
9. Repeat the steps until SI>=DI
10. If all the Characters match print string is palindrome else print not palindrome

FlowChart:



Code Screenshot:



Assembly Language Code:

```
.model small
.stack 100H

.data

INPUT DB 255 DUP("$")
OUT1 DB "String is Palindrome", "$"
OUT2 DB "String is not Palindrome", "$"
INMSG DB "Enter the string to be Checked","$"

.code

MOV AX,@DATA
MOV DS,AX

BEGIN:
    LEA DX,INMSG
    MOV AH,09H
    INT 21H

    LEA SI,INPUT

INS:
    MOV AH,01H
    INT 21H
    CMP AL,13
    JE START
    MOV [SI],AL
```

```
        INC SI
        JMP INS



START:
        MOV SI,OFFSET INPUT


LOOP1:
        MOV AX,[SI]
        CMP AL,"$"
        JE CHECK
        INC SI
        JMP LOOP1
CHECK:
        MOV DI,OFFSET INPUT
        DEC SI
        CHECKLOOP:
            CMP SI,DI
            JL PALINOUT
            MOV AX,[SI]
            MOV BX,[DI]
            CMP AL,BL
            JNE NOTPALINOUT
            DEC SI
            INC DI
            JMP CHECKLOOP
PALINOUT:
        LEA DX,OUT1
        MOV AH,09H
```

```
    INT 21H
    RET
NOTPALINOUT:
    LEA DX,OUT2
    MOV AH,09H
    INT 21H
    RET

end
```

Execution Proof:

Command: masm palindrome.asm

```
C:\>masm palindrome.asm
Microsoft (R) Macro Assembler Version 5.00
Copyright (C) Microsoft Corp 1981-1985, 1987.  All rights reserved.

Object filename [palindrome.OBJ]:
Source listing  [NUL.LST]:
Cross-reference [NUL.CRF]:

  51640 + 464904 Bytes symbol space free

      0 Warning Errors
      0 Severe  Errors

C:\>S
```

Command: link palindrome.obj

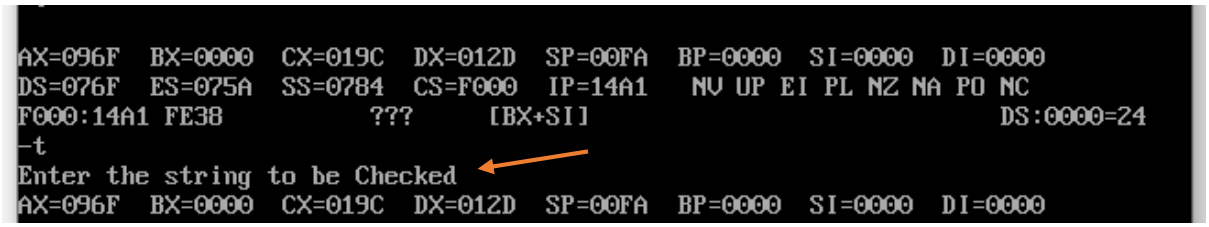
```
C:\>link palindrome.obj

Microsoft (R) Overlay Linker  Version 3.60
Copyright (C) Microsoft Corp 1983-1987.  All rights reserved.

Run File [PALINDROME.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:

C:\>S
```

Command: debug palindrome.exe

```
C:\>debug palindrome.exe
-S_
```

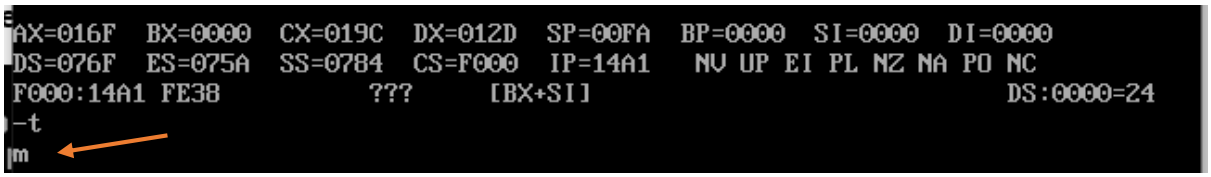
We give Input character by character as we are running in debugging mode.

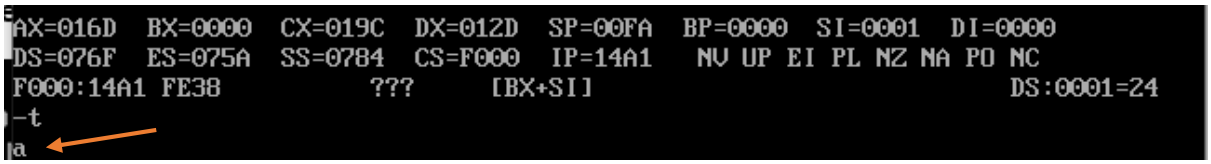
So to give a input of length 5 we need to give 5 characters line by line.

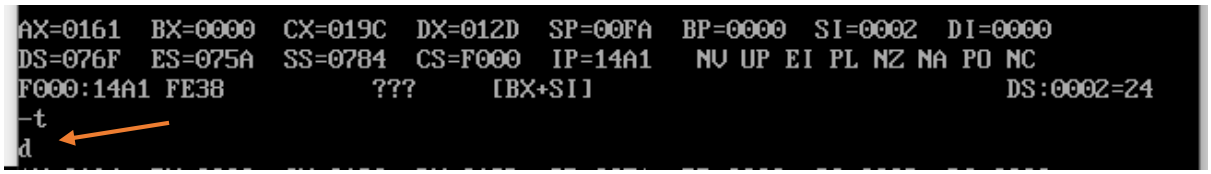


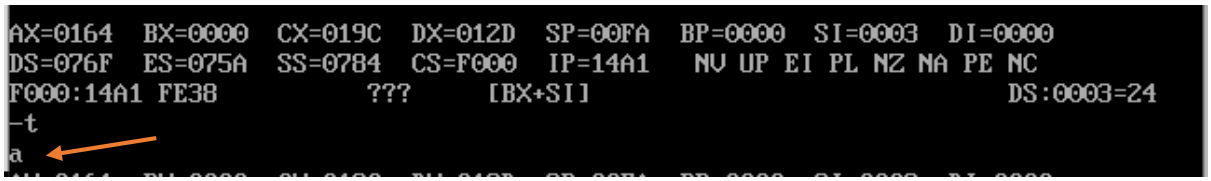
Giving Inputs:

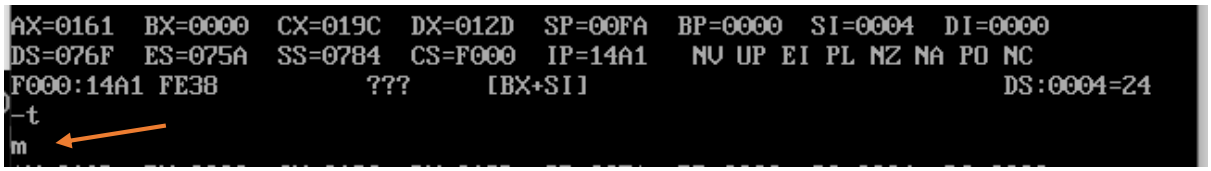
Given Input : **madam**





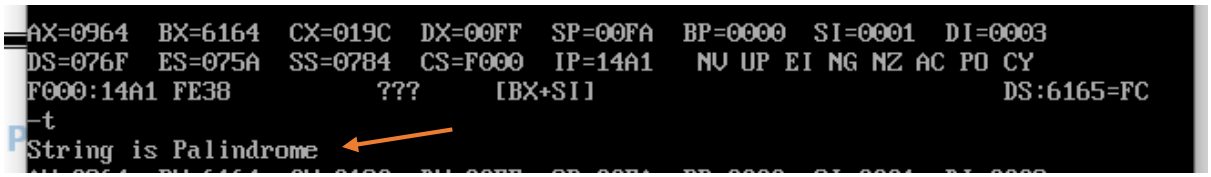






Press Enter to End Entering the String

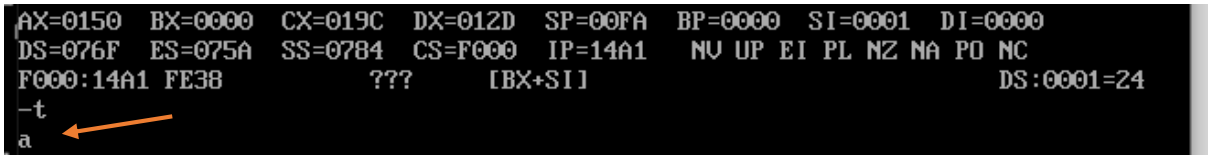
The Output is:

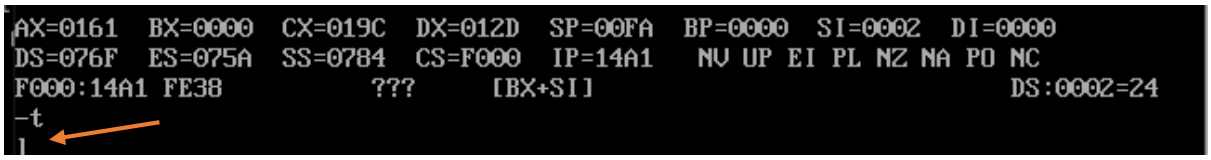


Hence the String is **Palindrome**

Giving Inputs:

Given Input : **Pal**





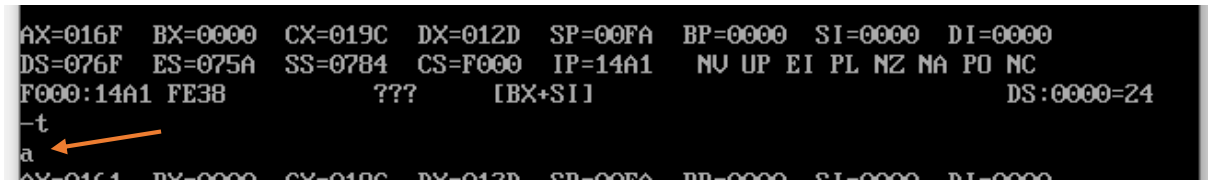
Press Enter to End Entering the String

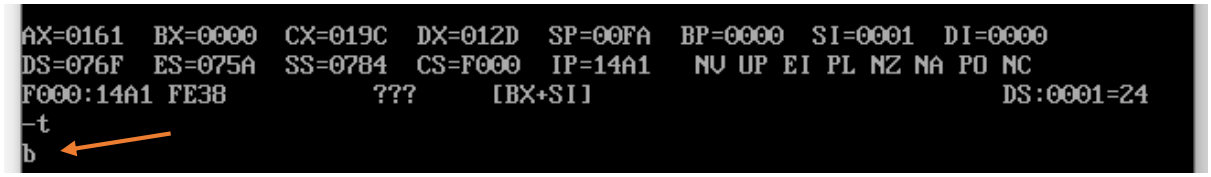
The Output is:

Hence the String is **Not Palindrome**

Giving Inputs:

Given Input : **abba**





```
AX=0162  BX=0000  CX=019C  DX=012D  SP=00FA  BP=0000  SI=0003  DI=0000
DS=076F  ES=075A  SS=0784  CS=F000  IP=14A1   NV UP EI PL NZ NA PE NC
F000:14A1 FE38          ???     [BX+SI]                             DS:0003=24
-t
a
```

Press Enter to End Entering the String

The Output is:

```
AX=0962  BX=6262  CX=019C  DX=00FF  SP=00FA  BP=0000  SI=0001  DI=0002
DS=076F  ES=075A  SS=0784  CS=F000  IP=14A1   NV UP EI NG NZ AC PE CY
F000:14A1 FE38          ???     [BX+SI]                             DS:6263=46
-t
String is Palindrome
```

Hence the String is **Palindrome**