# Lab Assignment-9

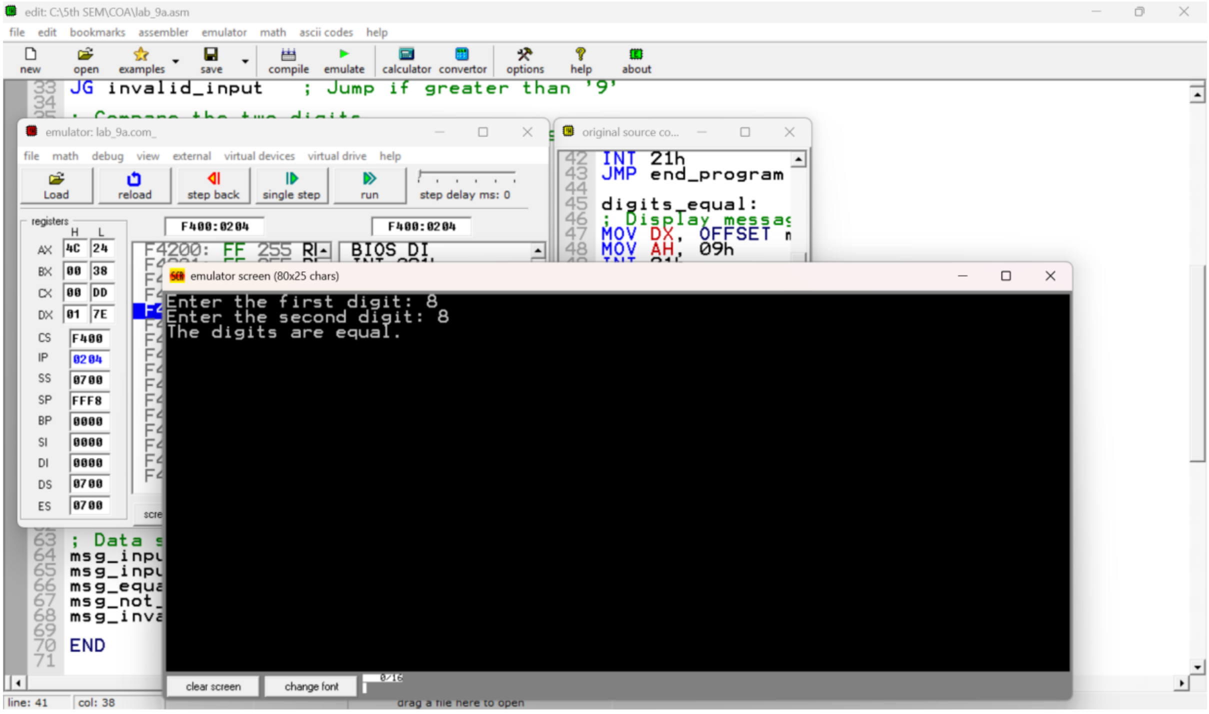
### 1.Write a program in assembly language to take two single-digit numbers as input and display whether they are equal or not.

**Code:**

```
ORG 100h      ;start of the program in memory
; Display the message "Enter the first digit: "
MOV DX, OFFSET msg_input1
MOV AH, 09h
INT 21h
; Read the first digit from the user
MOV AH, 01h
INT 21h
MOV BL, AL         ; Store the first digit in BL
; Check if the first input is a digit
CMP BL, '0'        ; Compare with '0'
JL invalid_input   ; Jump if less than '0'
CMP BL, '9'        ; Compare with '9'
JG invalid_input   ; Jump if greater than '9'
; Display the message "Enter the second digit: "
MOV DX, OFFSET msg_input2
MOV AH, 09h
INT 21h
; Read the second digit from the user
MOV AH, 01h
INT 21h
;MOV CL, AL         ; Store the second digit in CL
; Check if the second input is a digit
CMP AL, '0'        ; Compare with '0'
JL invalid_input   ; Jump if less than '0'
CMP AL, '9'        ; Compare with '9'
```

```
JG invalid_input   ; Jump if greater than '9'
; Compare the two digits
CMP BL, AL         ; Compare the two digits
JE digits_equal     ; Jump if equal
; Display message for not equal
MOV DX, OFFSET msg_not_equal
MOV AH, 09h
INT 21h
JMP end_program
digits_equal:
; Display message for equal
MOV DX, OFFSET msg_equal
MOV AH, 09h
INT 21h
end_program:
; Terminate the program
MOV AH, 4Ch
INT 21h
invalid_input:
; Display message for invalid input
MOV DX, OFFSET msg_invalid
MOV AH, 09h
INT 21h
JMP end_program
; Data section with messages
msg_input1  DB 'Enter the first digit: $'
msg_input2  DB 0Dh, 0Ah, 'Enter the second digit: $'
msg_equal   DB 0Dh, 0Ah, 'The digits are equal.$'
msg_not_equal DB 0Dh, 0Ah, 'The digits are not equal.$'
msg_invalid DB 0Dh, 0Ah, 'Invalid input! Please enter digits only.$'
END
```

# OUTPUT:



edit: C:\5th SEM\COA\lab_9a.asm
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about
emulator: lab_9a.com_
file math debug view external virtual devices virtual drive help
Load reload step back single step run step delay ms: 0
registers
original source co...
emulator screen (80x25 chars)
clear screen change font
line: 41 col: 38
drag a file here to open

```
JG invalid_input    ; Jump if greater than '9'
```

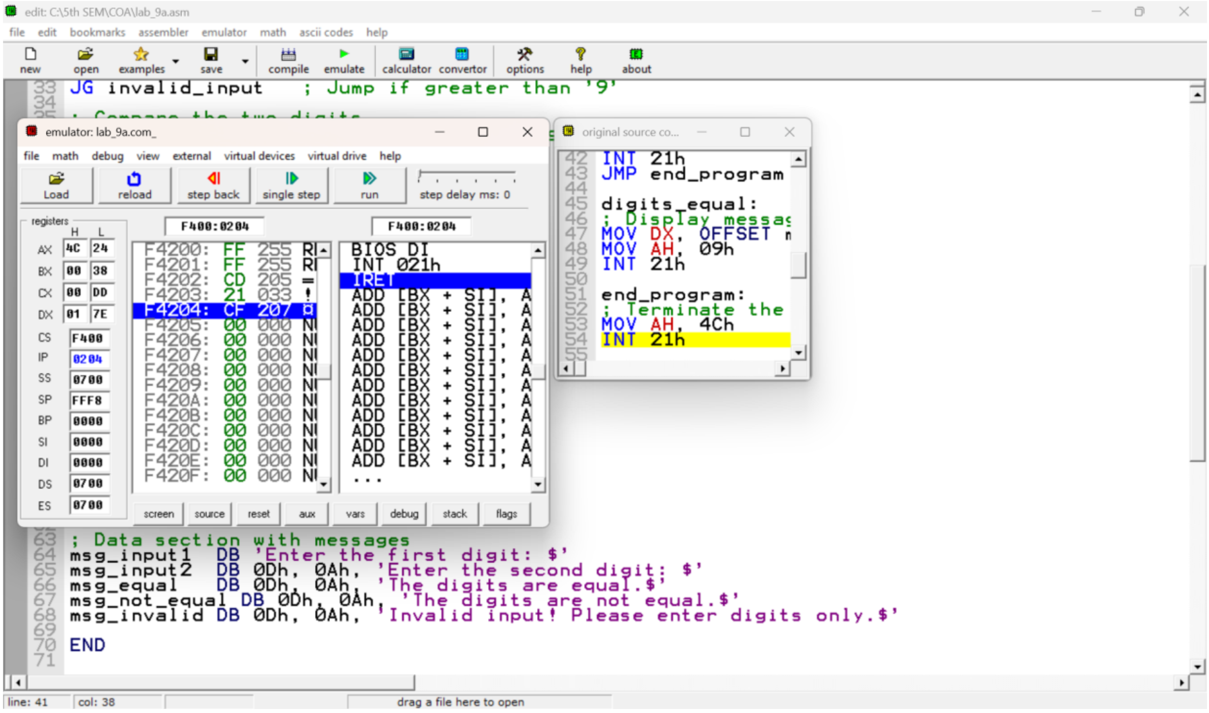
```
INT 21h
JMP end_program

digits_equal:
; Display messag
MOV DX, OFFSET m
MOV AH, 09h
```

| | H | L |
|---|---|---|
| AX | 4C | 24 |
| BX | 00 | 38 |
| CX | 00 | DD |
| DX | 01 | 7E |
| CS | F400 | |
| IP | 0204 | |
| SS | 0700 | |
| SP | FFF8 | |
| BP | 0000 | |
| SI | 0000 | |
| DI | 0000 | |
| DS | 0700 | |
| ES | 0700 | |

```
F400:0204
F4200: FF 255 RI
BIOS DI
```

```
Enter the first digit: 8
Enter the second digit: 8
The digits are equal.
```

```
; Data s
msg_inpu
msg_inpu
msg_equa
msg_not_
msg_inva

END
```



edit: C:\5th SEM\COA\lab_9a.asm
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about
emulator: lab_9a.com_
file math debug view external virtual devices virtual drive help
Load reload step back single step run step delay ms: 0
registers
original source co...
screen source reset aux vars debug stack flags
line: 41 col: 38
drag a file here to open

```
JG invalid_input    ; Jump if greater than '9'
```

| | H | L |
|---|---|---|
| AX | 4C | 24 |
| BX | 00 | 38 |
| CX | 00 | DD |
| DX | 01 | 7E |
| CS | F400 | |
| IP | 0204 | |
| SS | 0700 | |
| SP | FFF8 | |
| BP | 0000 | |
| SI | 0000 | |
| DI | 0000 | |
| DS | 0700 | |
| ES | 0700 | |

```
F400:0204
F4200: FF 255 RI
F4201: FF 255 RI
F4202: CD 205 =
F4203: 21 033 !
F4204: CF 207 ¤
F4205: 00 000 NI
F4206: 00 000 NI
F4207: 00 000 NI
F4208: 00 000 NI
F4209: 00 000 NI
F420A: 00 000 NI
F420B: 00 000 NI
F420C: 00 000 NI
F420D: 00 000 NI
F420E: 00 000 NI
F420F: 00 000 NI
```

```
F400:0204
BIOS DI
INT 021h
IRET
ADD [BX + SI], A
ADD [BX + SI], A
ADD [BX + SI], A
ADD [BX + SI], A
ADD [BX + SI], A
ADD [BX + SI], A
ADD [BX + SI], A
ADD [BX + SI], A
ADD [BX + SI], A
ADD [BX + SI], A
ADD [BX + SI], A
ADD [BX + SI], A
...
```

```
INT 21h
JMP end_program

digits_equal:
; Display messag
MOV DX, OFFSET m
MOV AH, 09h
INT 21h

end_program:
; Terminate the
MOV AH, 4Ch
INT 21h
```

```
; Data section with messages
msg_input1  DB 'Enter the first digit: $'
msg_input2  DB 0Dh, 0Ah, 'Enter the second digit: $'
msg_equal   DB 0Dh, 0Ah, 'The digits are equal.$'
msg_not_equal DB 0Dh, 0Ah, 'The digits are not equal.$'
msg_invalid DB 0Dh, 0Ah, 'Invalid input! Please enter digits only.$'

END
```

## Practice set:

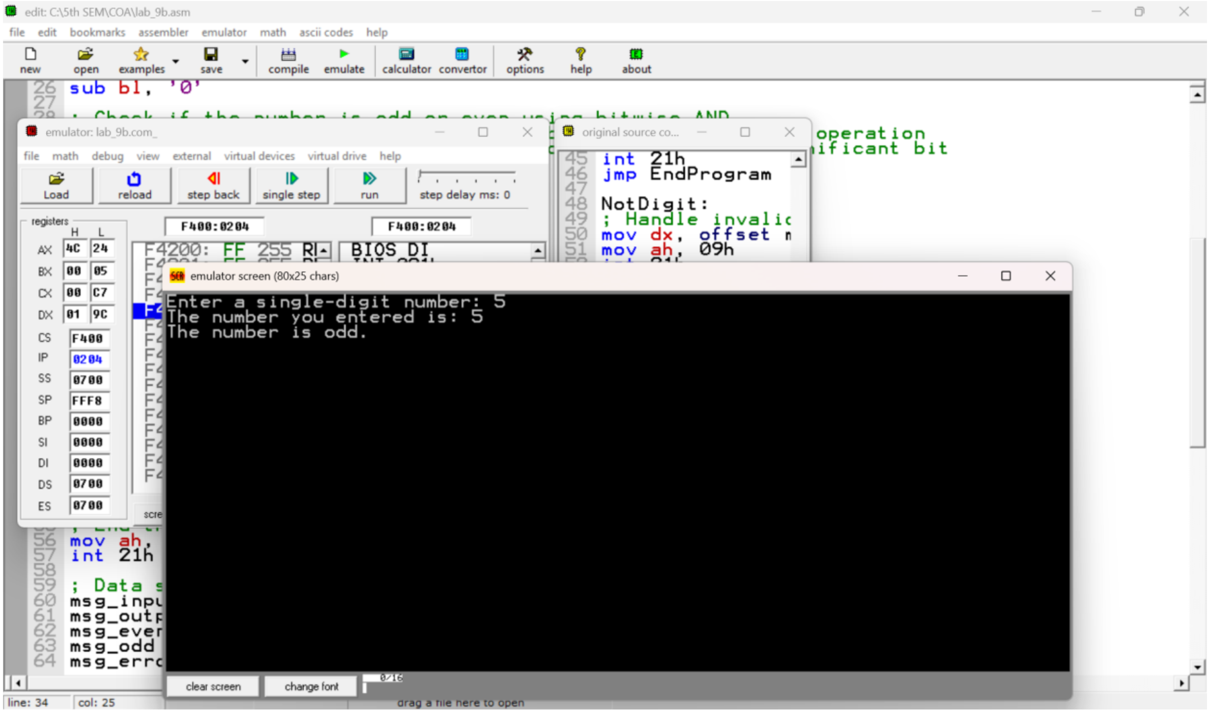
## 2. Write a program in assembly language to check whether a single-digit number is odd or even.

## Code:

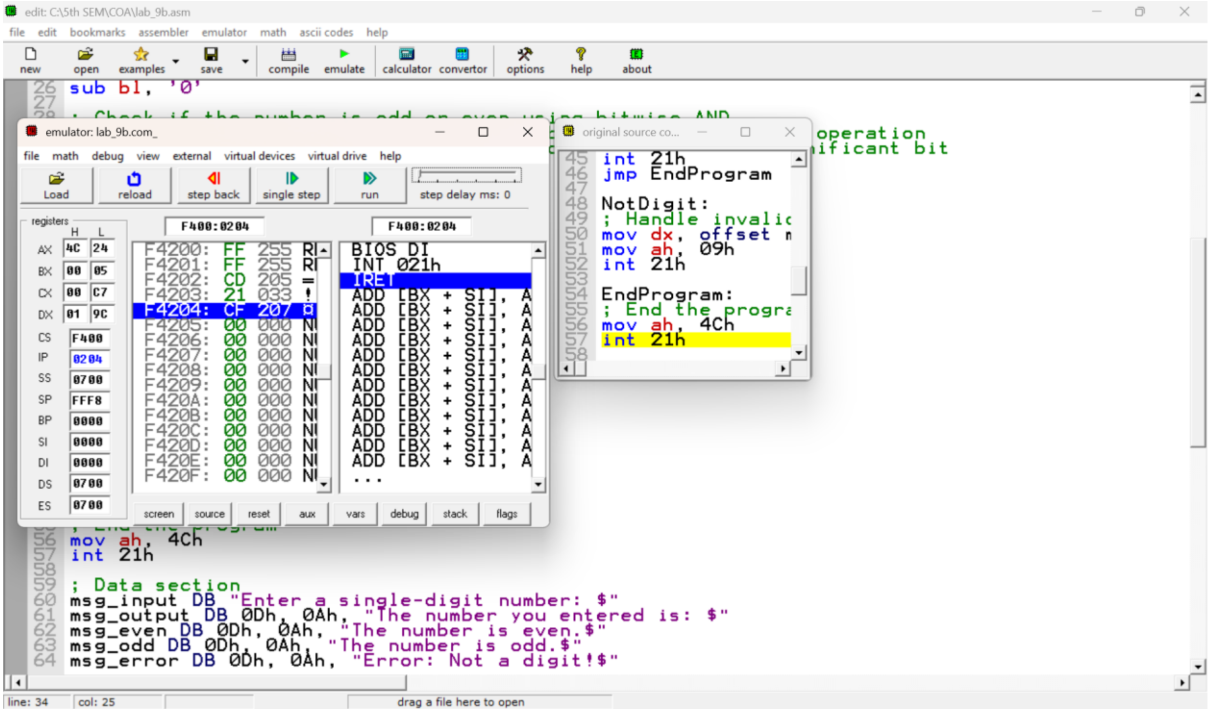
```
ORG 100h

; Prompt for the single-digit number

mov dx, offset msg_input

mov ah, 09h

int 21h

; Get the single digit

mov ah, 01h

int 21h

mov bl, al           ; Store the input in BL

cmp al, '0'          ; Check if it's a valid digit (ASCII '0' = 48)

jl NotDigit          ; If less than '0', it's not a digit

cmp al, '9'          ; Check if it's greater than '9' (ASCII '9' = 57)

jg NotDigit          ; If greater than '9', it's not a digit

; Display the input digit

mov dx, offset msg_output

mov ah, 09h

int 21h

mov dl, bl

mov ah, 02h

int 21h

; Convert the digit from ASCII to numeric value

sub bl, '0'

; Check if the number is odd or even using bitwise AND

mov al, bl           ; Move the number to AL for bitwise operation

and al, 1            ; AND with 1 to check the least significant bit

jz Even              ; If zero, the number is even

jmp Odd              ; If not zero, the number is odd

Even:
```

```
; Display "The number is even"
mov dx, offset msg_even
mov ah, 09h
int 21h
jmp EndProgram
Odd:
; Display "The number is odd"
mov dx, offset msg_odd
mov ah, 09h
int 21h
jmp EndProgram
NotDigit:
; Handle invalid input
mov dx, offset msg_error
mov ah, 09h
int 21h
EndProgram:
; End the program
mov ah, 4Ch
int 21h
; Data section
msg_input DB "Enter a single-digit number: $"
msg_output DB 0Dh, 0Ah, "The number you entered is: $"
msg_even DB 0Dh, 0Ah, "The number is even.$"
msg_odd DB 0Dh, 0Ah, "The number is odd.$"
msg_error DB 0Dh, 0Ah, "Error: Not a digit!$"
```

# Output:



```
Enter a single-digit number: 5
The number you entered is: 5
The number is odd.
```



```
int 21h
jmp EndProgram

NotDigit:
; Handle invalic
mov dx, offset m
mov ah, 09h
int 21h

EndProgram:
; End the progra
mov ah, 4Ch
int 21h
```

```
mov ah, 4Ch
int 21h

; Data section
msg_input DB "Enter a single-digit number: $"
msg_output DB 0Dh, 0Ah, "The number you entered is: $"
msg_even DB 0Dh, 0Ah, "The number is even.$"
msg_odd DB 0Dh, 0Ah, "The number is odd.$"
msg_error DB 0Dh, 0Ah, "Error: Not a digit!$"
```