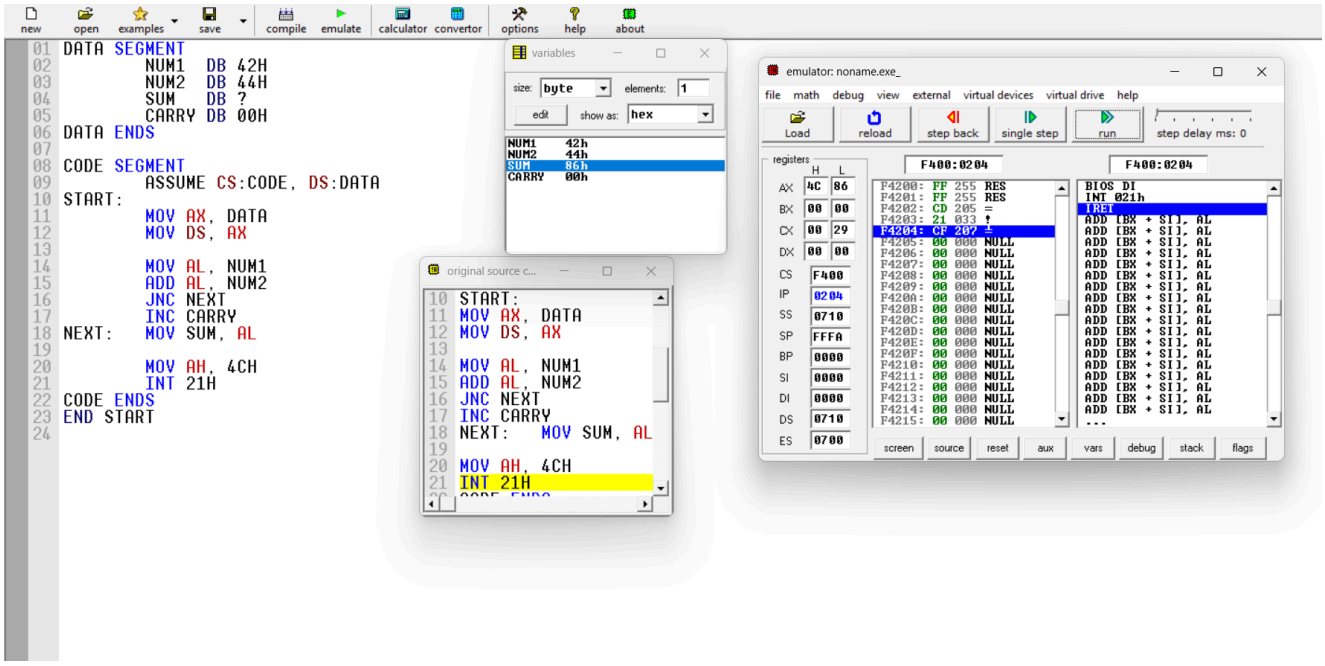
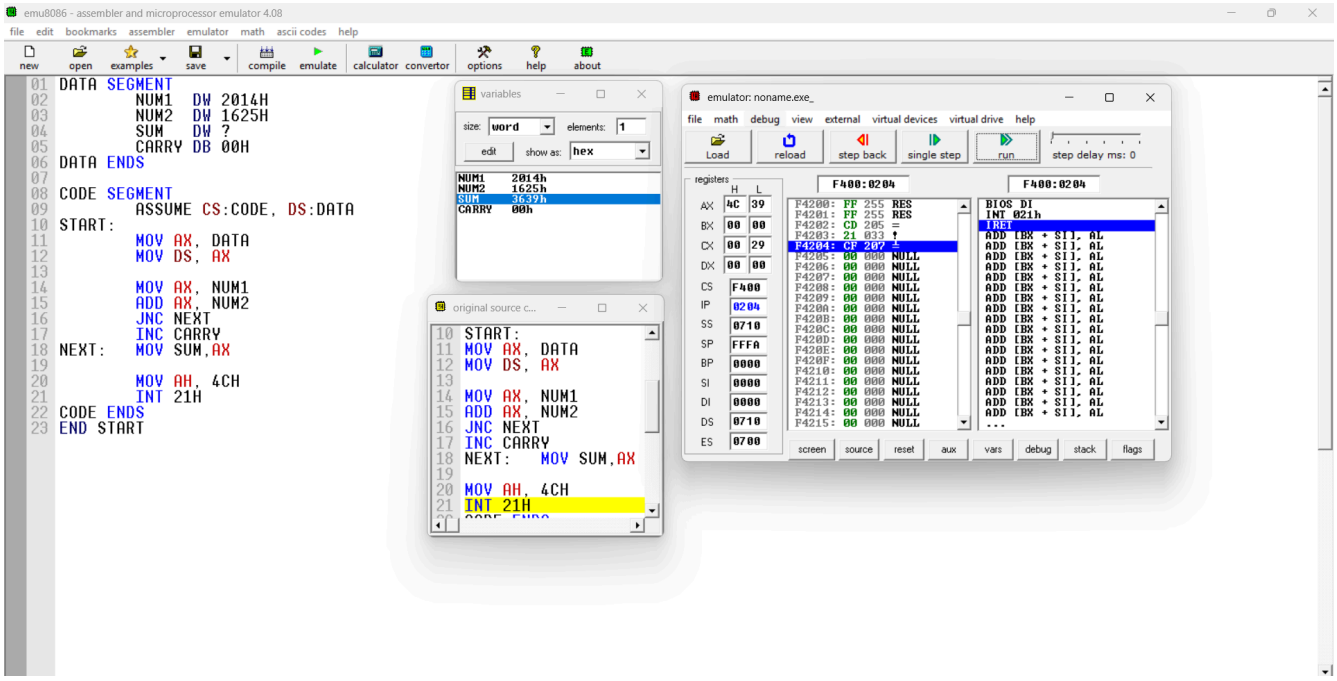
MIC - Practical - QB - Solutions - By - Th3_

1. Write ALP to perform addition of two 8 bit numbers



```
DATA SEGMENT
        NUM1  DB 42H
        NUM2  DB 44H
        SUM   DB ?
        CARRY DB 00H
DATA ENDS

CODE SEGMENT
        ASSUME CS:CODE, DS:DATA
START:
        MOV AX, DATA
        MOV DS, AX

        MOV AL, NUM1
        ADD AL, NUM2
        JNC NEXT
        INC CARRY
NEXT:   MOV SUM, AL

        MOV AH, 4CH
        INT 21H
CODE ENDS
END START
```

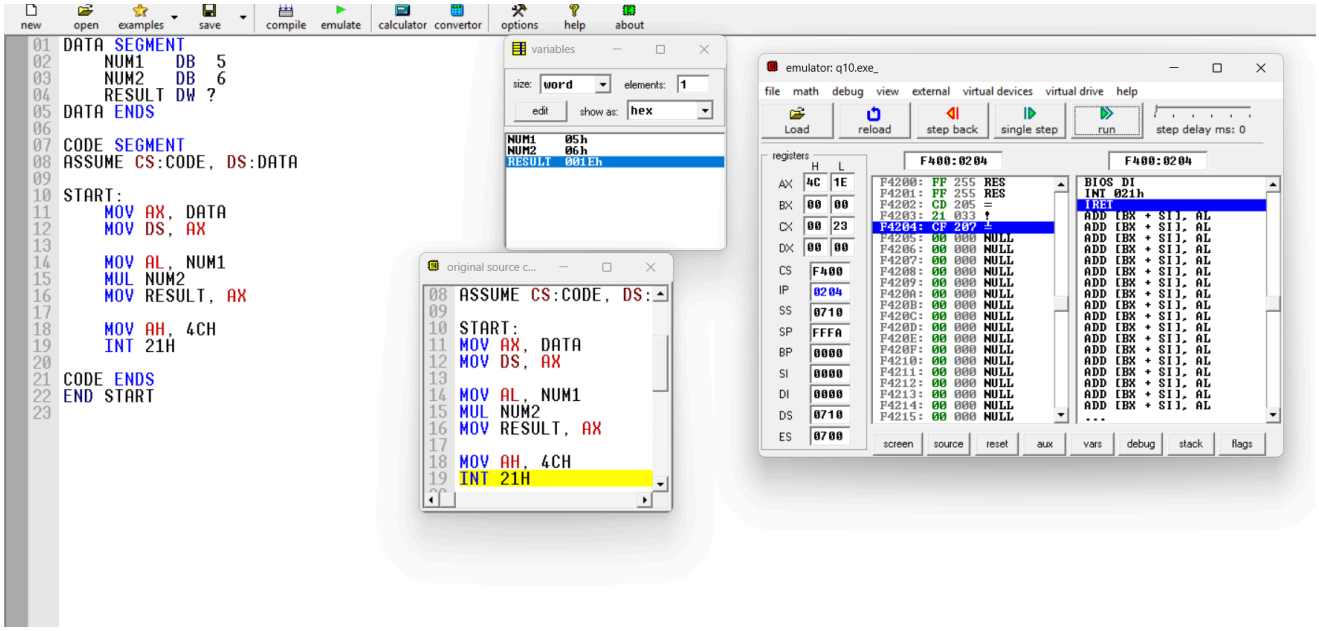
2. Write ALP to perform addition of two 16 bit numbers



```
DATA SEGMENT
        NUM1  DW 2014H
        NUM2  DW 1625H
        SUM   DW ?
        CARRY DB 00H
DATA ENDS

CODE SEGMENT
        ASSUME CS:CODE, DS:DATA
START:
        MOV AX, DATA
        MOV DS, AX

        MOV AX, NUM1
        ADD AX, NUM2
        JNC NEXT
        INC CARRY
NEXT:   MOV SUM,AX

        MOV AH, 4CH
        INT 21H
CODE ENDS
END START
```

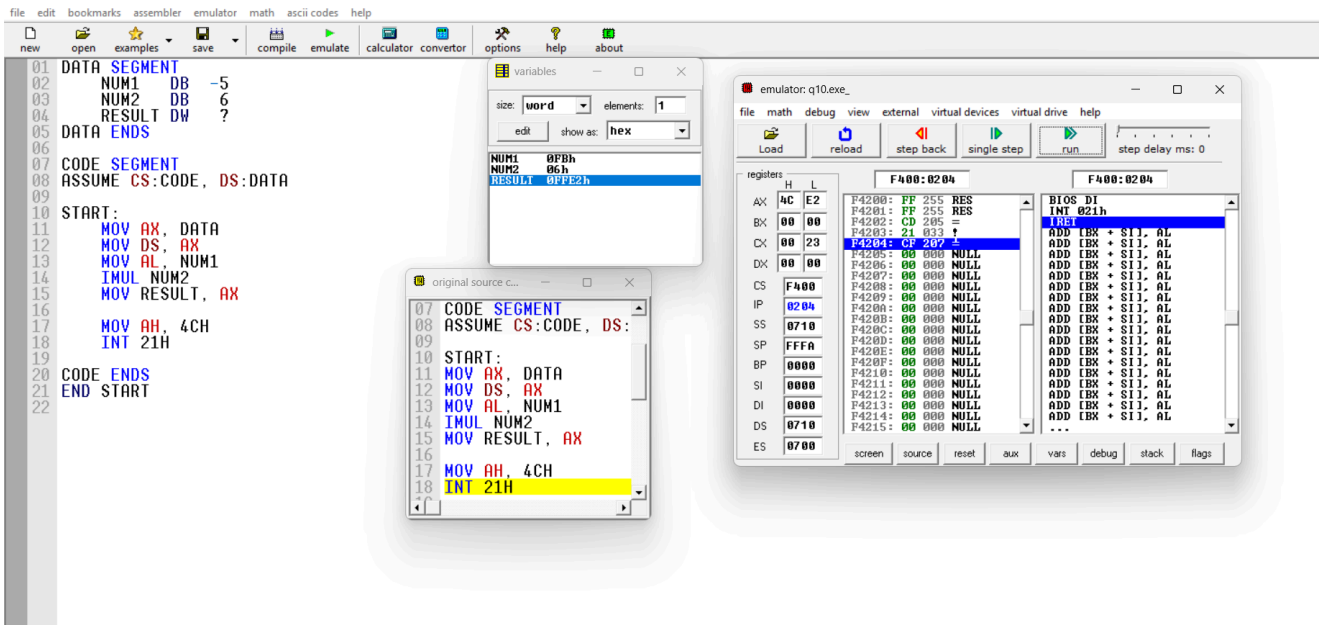
MIC - Practical - QB - Solutions - By - Th3_

3. Write ALP to perform multiplication of two unsigned 8 bit numbers



```
DATA SEGMENT
    NUM1   DB  5
    NUM2   DB  6
    RESULT DW ?
DATA ENDS

CODE SEGMENT
ASSUME CS:CODE, DS:DATA

START:
    MOV AX, DATA
    MOV DS, AX

    MOV AL, NUM1
    MUL NUM2
    MOV RESULT, AX

    MOV AH, 4CH
    INT 21H

CODE ENDS
END START
```

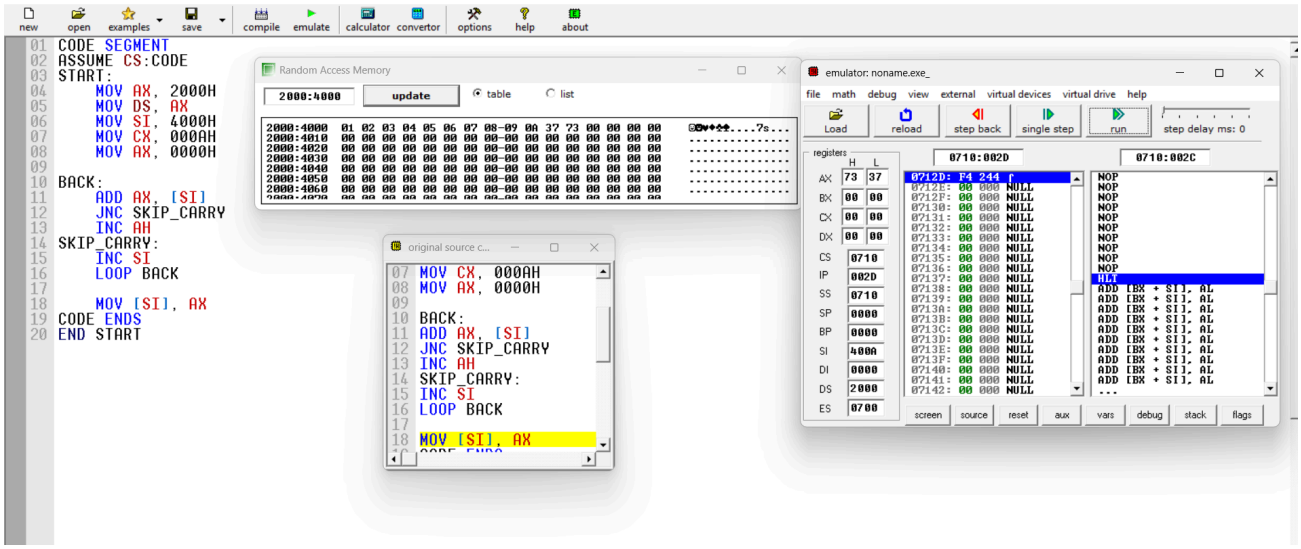
4. Write ALP to perform multiplication of two signed 8 bit numbers



```
DATA SEGMENT
    NUM1   DB  -5
    NUM2   DB   6
    RESULT DW   ?
DATA ENDS

CODE SEGMENT
ASSUME CS:CODE, DS:DATA

START:
    MOV AX, DATA
    MOV DS, AX
    MOV AL, NUM1
    IMUL NUM2
    MOV RESULT, AX

    MOV AH, 4CH
    INT 21H

CODE ENDS
END START
```

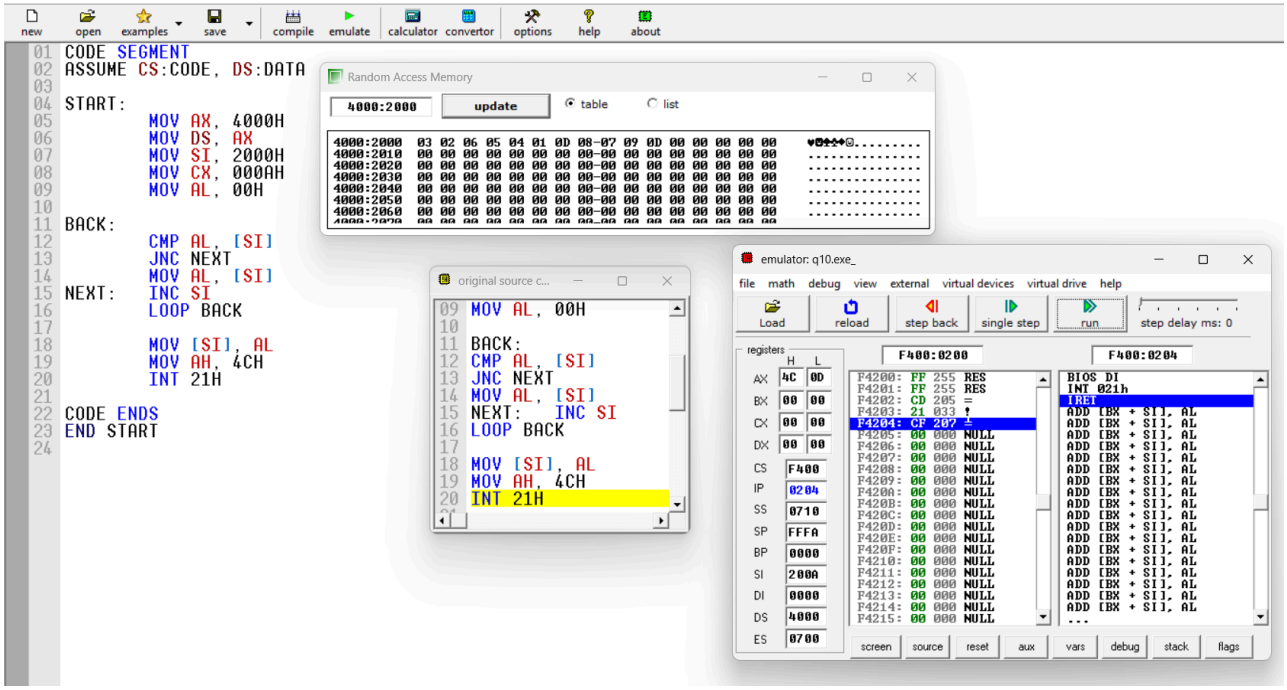
MIC - Practical - QB - Solutions - By - Th3_

5. Write an ALP to add series of 10 Numbers



```
CODE SEGMENT
ASSUME CS:CODE
START:
    MOV AX, 2000H
    MOV DS, AX
    MOV SI, 4000H
    MOV CX, 000AH
    MOV AX, 0000H

BACK:
    ADD AX, [SI]
    JNC SKIP_CARRY
    INC AH
SKIP_CARRY:
    INC SI
    LOOP BACK

    MOV [SI], AX
CODE ENDS
END START
```

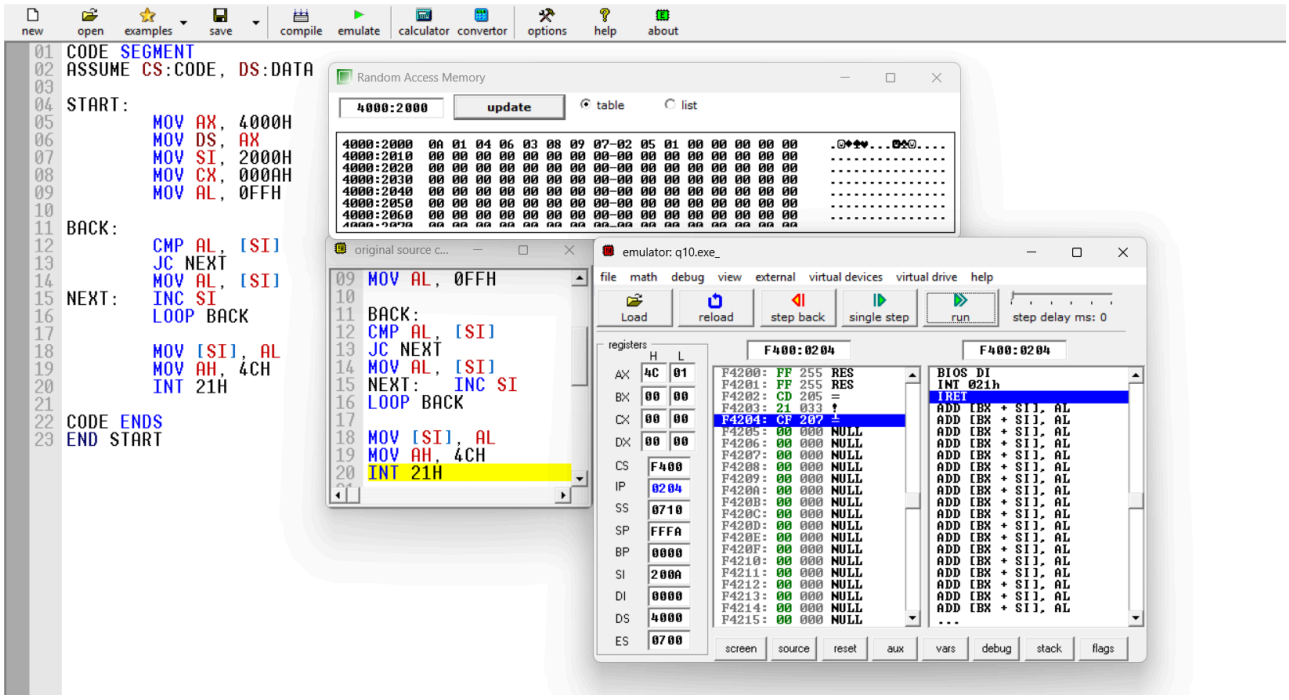
6. Write an ALP to Find Largest number



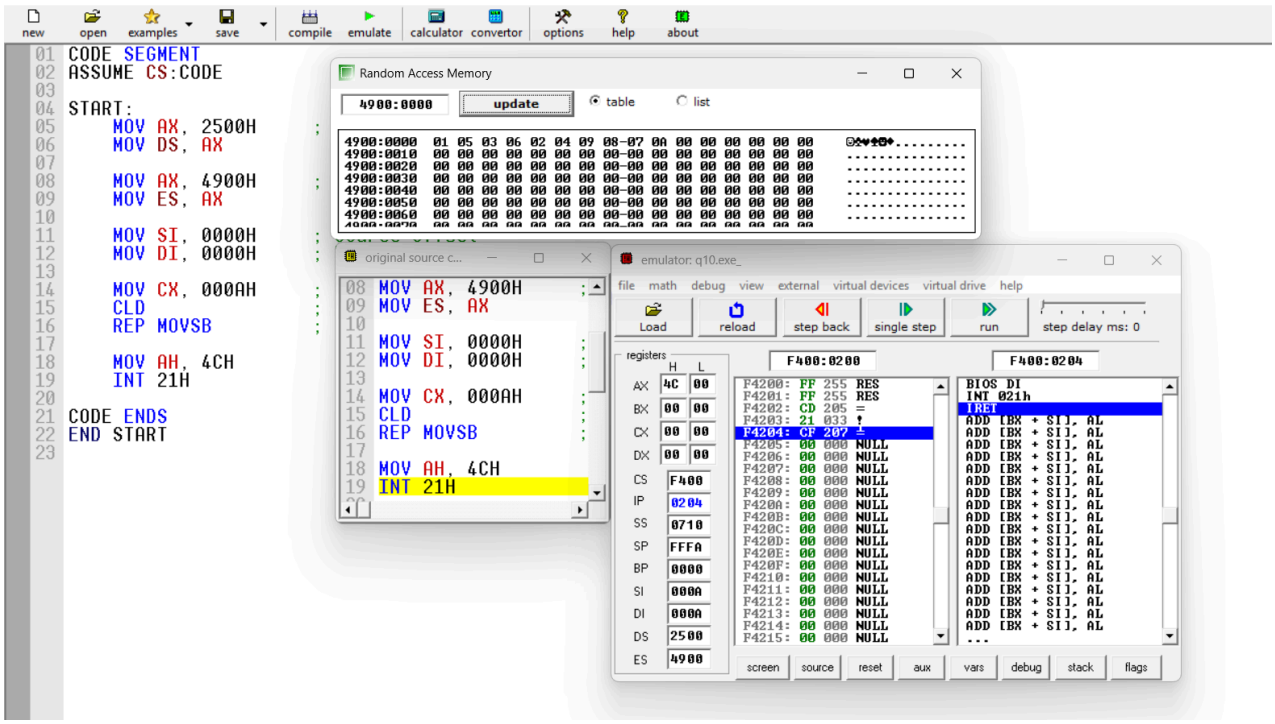
```
CODE SEGMENT
ASSUME CS:CODE, DS:DATA

START:
        MOV AX, 4000H
        MOV DS, AX
        MOV SI, 2000H
        MOV CX, 000AH
        MOV AL, 00H

BACK:
        CMP AL, [SI]
        JNC NEXT
        MOV AL, [SI]
NEXT:   INC SI
        LOOP BACK

        MOV [SI], AL
        MOV AH, 4CH
        INT 21H

CODE ENDS
END START
```

MIC - Practical - QB - Solutions - By - Th3_

7. Write an ALP to Find Smallest number

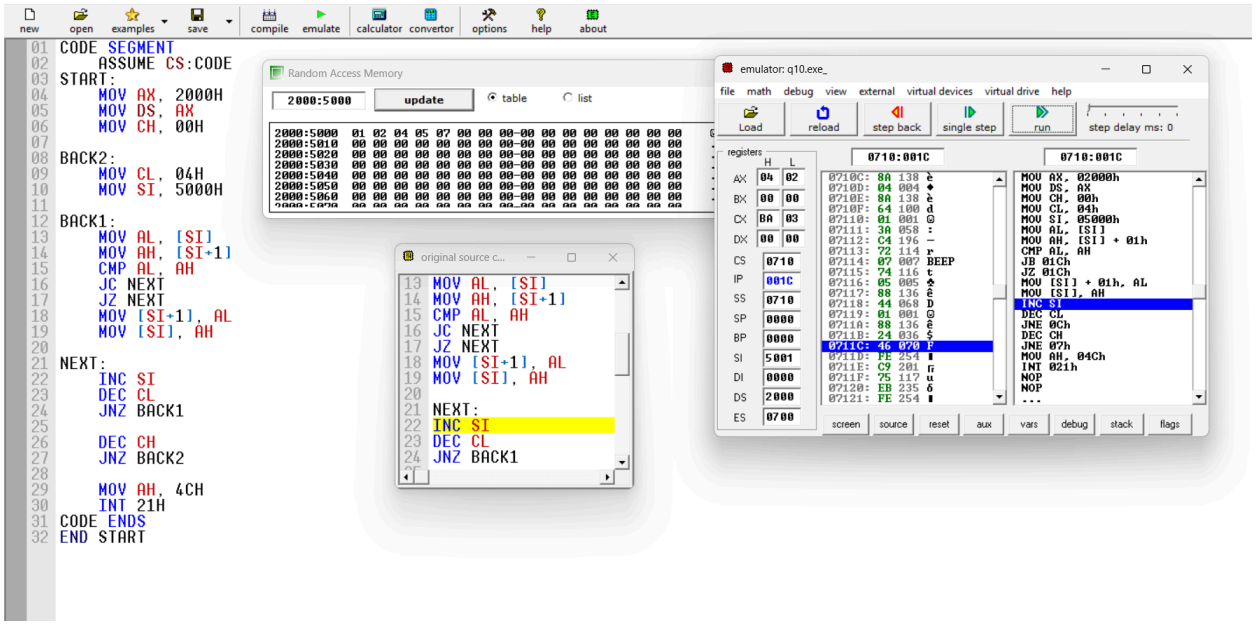


8. Write an ALP to Transfer 10 Bytes of Data From Memory Location 25000H to 49000H



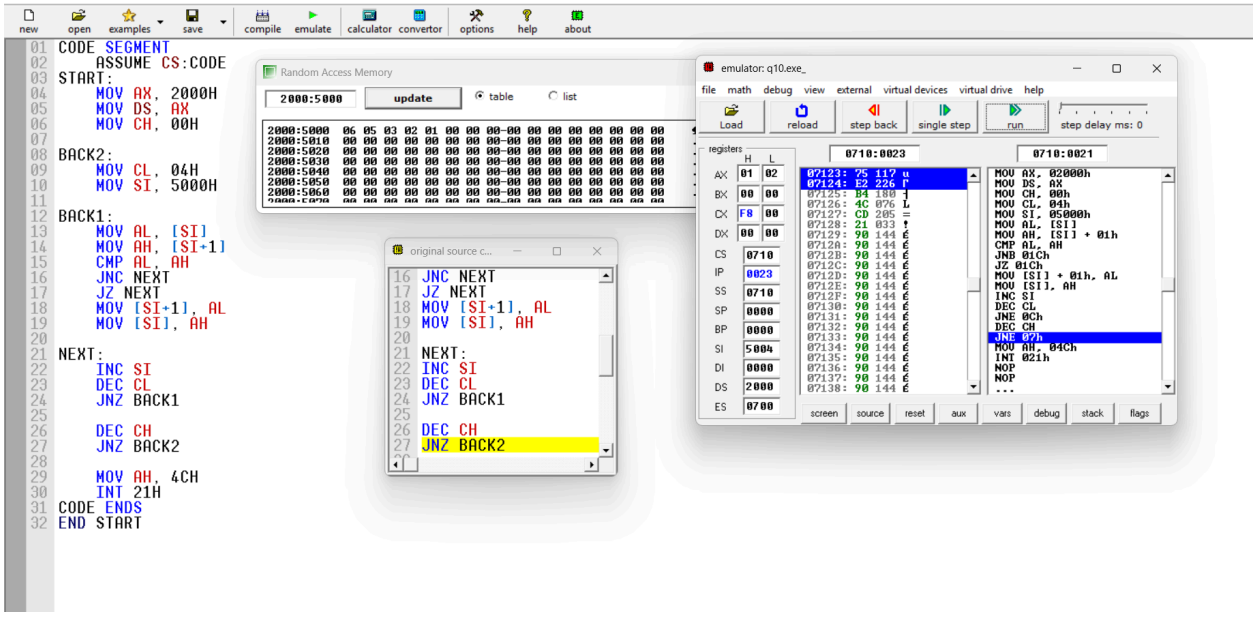
MIC - Practical - QB - Solutions - By - Th3_

9. Write an ALP to Sort a Series in Ascending Order



```
CODE SEGMENT
    ASSUME CS:CODE
START:
    MOV AX, 2000H
    MOV DS, AX
    MOV CH, 00H

BACK2:
    MOV CL, 04H
    MOV SI, 5000H

BACK1:
    MOV AL, [SI]
    MOV AH, [SI+1]
    CMP AL, AH
    JC NEXT
    JZ NEXT
    MOV [SI+1], AL
    MOV [SI], AH

NEXT:
    INC SI
    DEC CL
    JNZ BACK1

    DEC CH
    JNZ BACK2

    MOV AH, 4CH
    INT 21H
CODE ENDS
END START
```

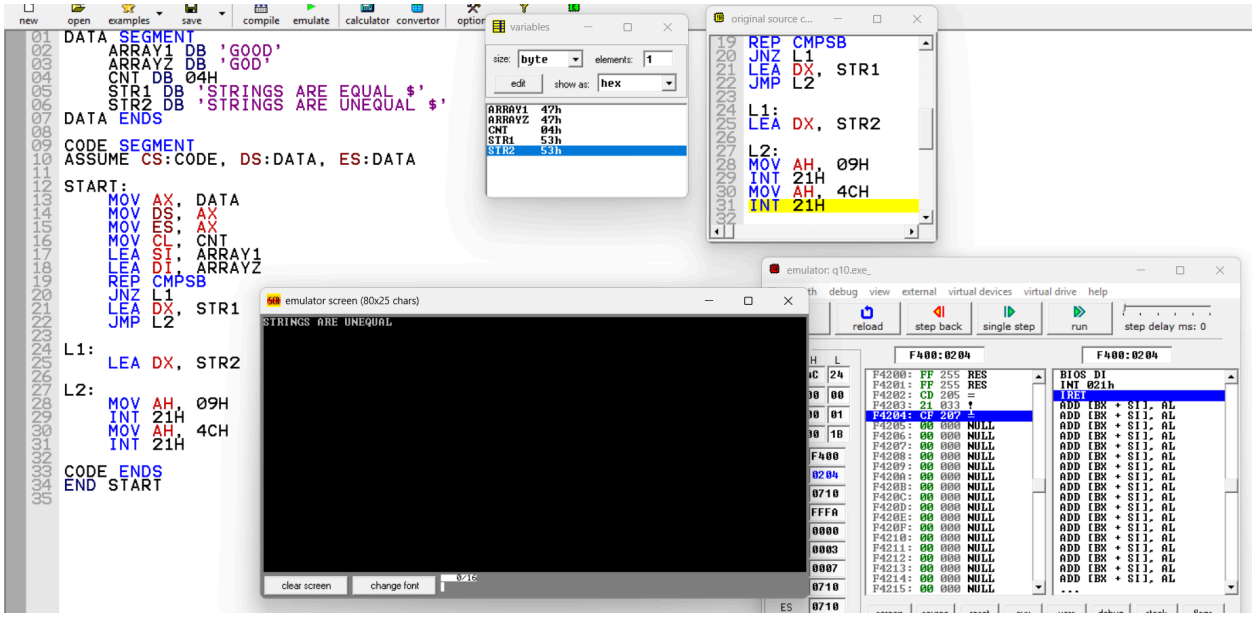
10. Write an ALP to Sort a Series in Descending Order



```
CODE SEGMENT
    ASSUME CS:CODE
START:
    MOV AX, 2000H
    MOV DS, AX
    MOV CH, 00H

BACK2:
    MOV CL, 04H
    MOV SI, 5000H

BACK1:
    MOV AL, [SI]
    MOV AH, [SI+1]
    CMP AL, AH
    JNC NEXT
    JZ NEXT
    MOV [SI+1], AL
    MOV [SI], AH

NEXT:
    INC SI
    DEC CL
    JNZ BACK1

    DEC CH
    JNZ BACK2

    MOV AH, 4CH
    INT 21H
CODE ENDS
END START
```

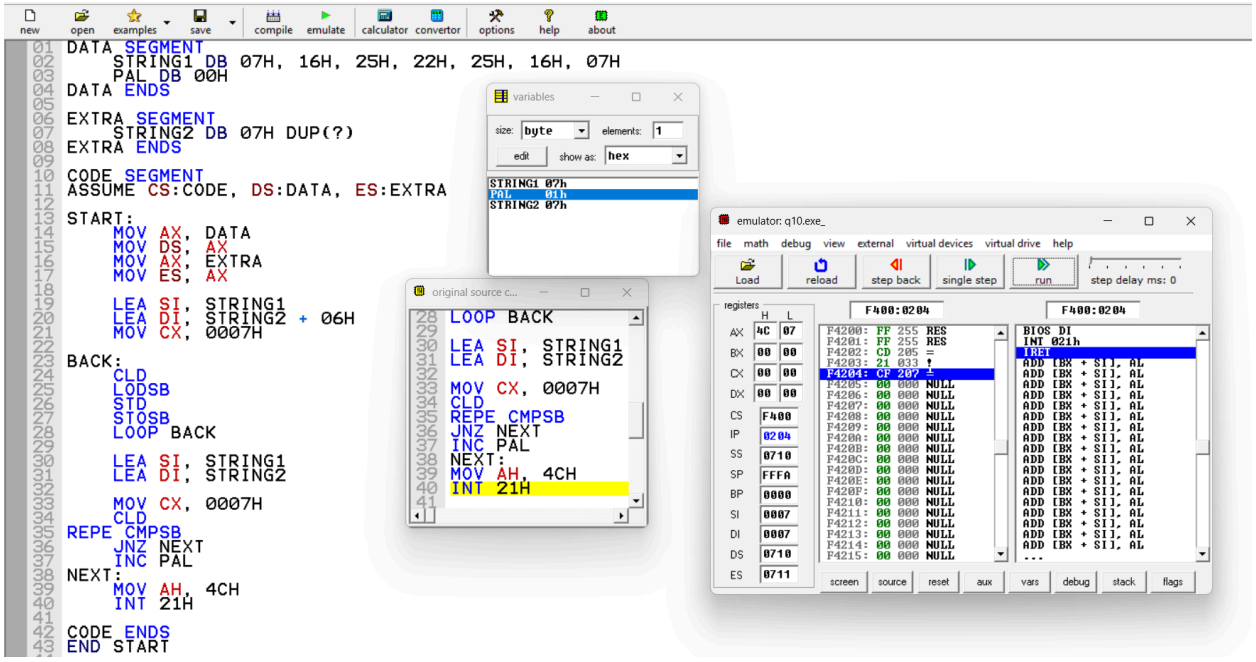
MIC - Practical - QB - Solutions - By - Th3_

11. Write an ALP to Compare Two Strings



```
DATA SEGMENT
    ARRAY1 DB 'GOOD'
    ARRAYZ DB 'GOD'
    CNT DB 04H
    STR1 DB 'STRINGS ARE EQUAL $'
    STR2 DB 'STRINGS ARE UNEQUAL $'
DATA ENDS

CODE SEGMENT
ASSUME CS:CODE, DS:DATA, ES:DATA

START:
    MOV AX, DATA
    MOV DS, AX
    MOV ES, AX
    MOV CL, CNT
    LEA SI, ARRAY1
    LEA DI, ARRAYZ
    REP CMPSB
    JNZ L1
    LEA DX, STR1
    JMP L2

L1:
    LEA DX, STR2

L2:
    MOV AH, 09H
    INT 21H
    MOV AH, 4CH
    INT 21H

CODE ENDS
END START
```

12. Write an ALP to Check whether String is Palindrome or not

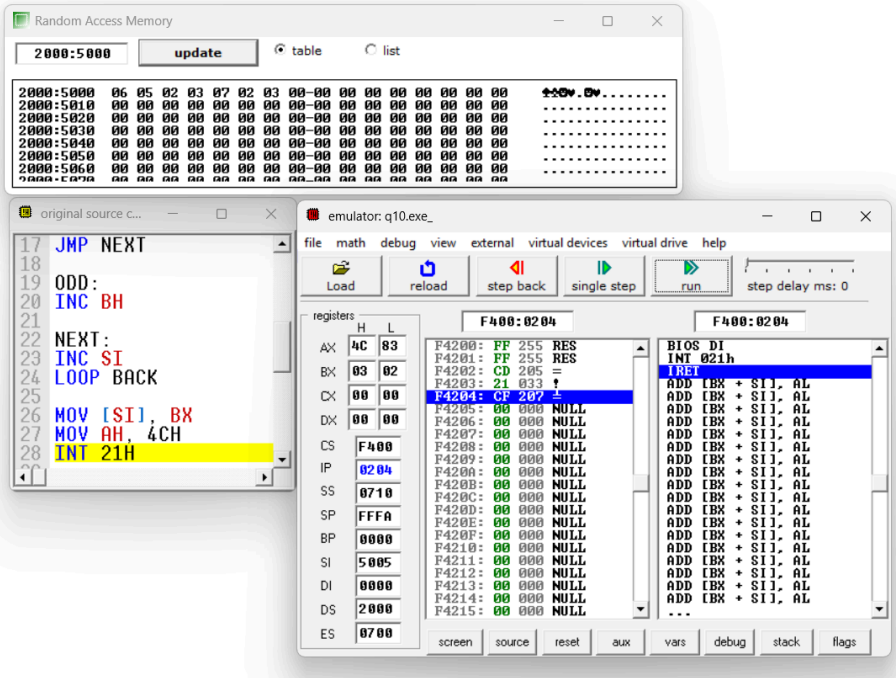


```
DATA SEGMENT
    STRING1 DB 07H, 16H, 25H, 22H, 25H, 16H, 07H
    PAL DB 00H
DATA ENDS

EXTRA SEGMENT
    STRING2 DB 07H DUP(?)
EXTRA ENDS

CODE SEGMENT
ASSUME CS:CODE, DS:DATA, ES:EXTRA

START:
    MOV AX, DATA
    MOV DS, AX
    MOV AX, EXTRA
    MOV ES, AX

    LEA SI, STRING1
    LEA DI, STRING2 + 06H
    MOV CX, 0007H

BACK:
    CLD
    LODSB
    STD
    STOSB
    LOOP BACK

    LEA SI, STRING1
    LEA DI, STRING2

    MOV CX, 0007H
    CLD
REPE CMPSB
    JNZ NEXT
    INC PAL
NEXT:
    MOV AH, 4CH
    INT 21H

CODE ENDS
END START
```

MIC - Practical - QB - Solutions - By - Th3_

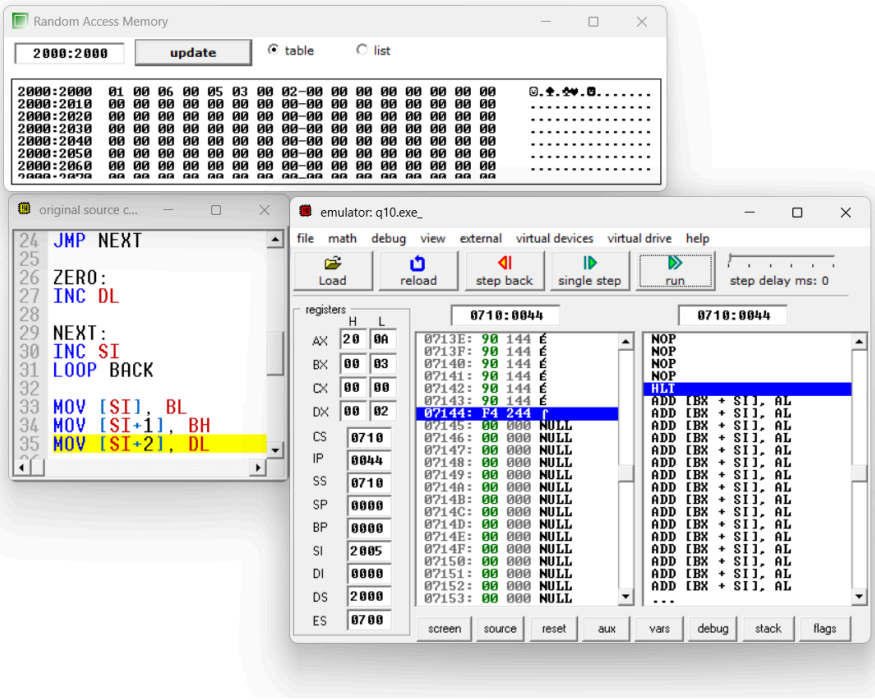
13. Write an ALP to count ODD and EVEN numbers

```
CODE SEGMENT
ASSUME CS:CODE

START:
    MOV AX, 2000H
    MOV DS, AX
    MOV SI, 5000H
    MOV CX, 05H
    MOV BL, 00H
    MOV BH, 00H

BACK:
    MOV AL, [SI]
    ROR AL, 01H
    JC ODD
    INC BL
    JMP NEXT

ODD:
    INC BH

NEXT:
    INC SI
    LOOP BACK

    MOV [SI], BX
    MOV AH, 4CH
    INT 21H

CODE ENDS
END START
```



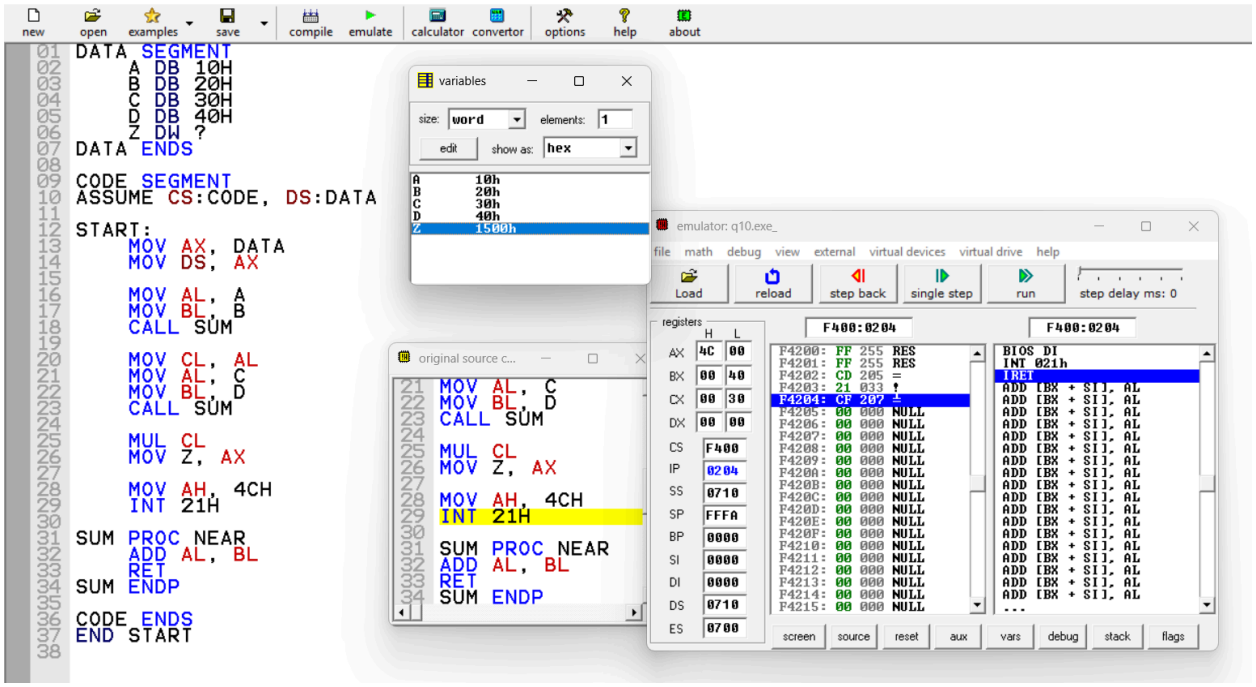
14. Write an ALP to Check whether given number is Zero, Positive or Negative

```
CODE SEGMENT
ASSUME CS:CODE

START:
    MOV AX, 2000H
    MOV DS, AX
    MOV SI, 2000H
    MOV CX, 05H
    MOV BL, 00H
    MOV BH, 00H
    MOV DL, 00H

BACK:
    MOV AL, [SI]
    ADD AL, 00H
    JZ ZERO
    ROL AL, 01H
    JC NEGATIVE
    INC BL
    JMP NEXT

NEGATIVE:
    INC BH
    JMP NEXT

ZERO:
    INC DL

NEXT:
    INC SI
    LOOP BACK

    MOV [SI], BL
    MOV [SI+1], BH
    MOV [SI+2], DL

CODE ENDS
END START
```



MIC - Practical - QB - Solutions - By - Th3_

15. Write an ALP using procedure to solve equation such as Z = (A + B) * (C + D)




```
DATA SEGMENT
     A DB 10H
     B DB 20H
     C DB 30H
     D DB 40H
     Z DW ?
DATA ENDS

CODE SEGMENT
ASSUME CS:CODE, DS:DATA

START:
     MOV AX, DATA
     MOV DS, AX

     MOV AL, A
     MOV BL, B
     CALL SUM

     MOV CL, AL
     MOV AL, C
     MOV BL, D
     CALL SUM

     MUL CL
     MOV Z, AX

     MOV AH, 4CH
     INT 21H

SUM PROC NEAR
     ADD AL, BL
     RET
SUM ENDP

CODE ENDS
END START
```