ASM Assignment 2

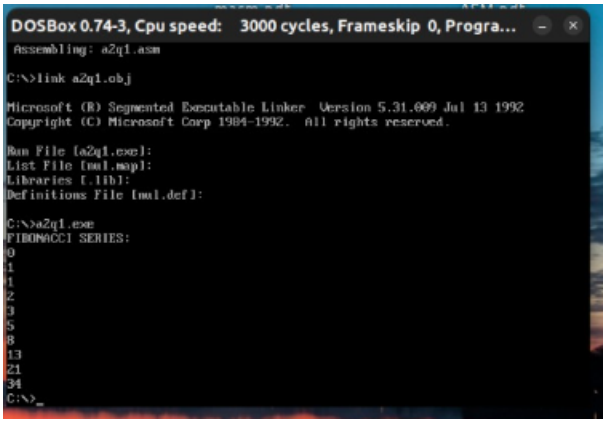
Submitted By: Kaji Manirul Islam
Roll: 0022-1100-1100
Dept: Information Technology
Class: 2nd Year 1st Semester

---

1. Write an assembly language program to display the first 10 Fibonacci numbers.

```
.MODEL SMALL
.STACK 100H
.DATA
VAR1    DB "FIBONACCI SERIES:",10,13,"$"
T1      DB 00H
T2      DB 01H
T3      DB 00H

.CODE
MAIN PROC
   MOV AX, @DATA
   MOV DS, AX
   MOV DX, OFFSET VAR1
   MOV AH, 09H
   INT 21H

   MOV DL, T2
   ADD DL, 30H
   MOV AH, 02H
   INT 21H

FIBO:
   MOV CX, 10
L1:
   PUSH CX
   MOV DL, 10
   MOV AH, 02H
   INT 21H
```

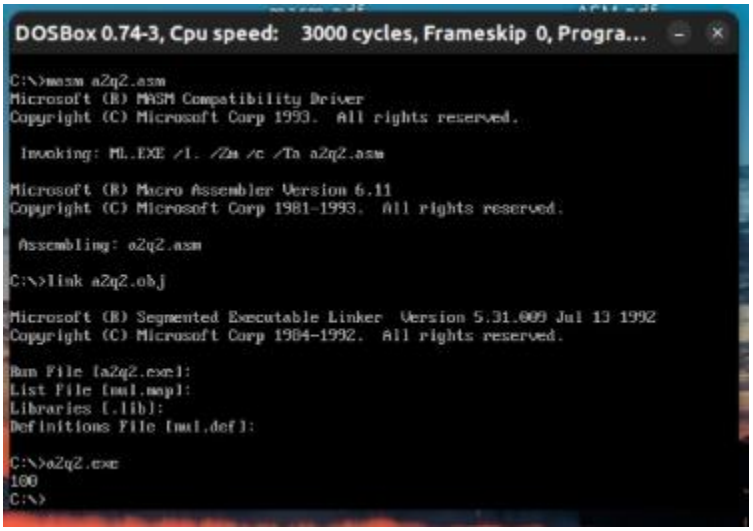
```
    MOV DL, 13
    MOV AH, 02H
    INT 21H
    MOV BL, T1
    ADD BL, T2
    MOV T3, BL
    MOV AH, 0
    MOV AL, T3
    MOV DX, 0
    MOV BX, 10
    MOV CX, 0

L2:
    DIV BX
    PUSH DX
    MOV DX, 0
    MOV AH, 0
    INC CX
    CMP AX, 0
    JNE L2

L3:
    POP DX
    ADD DX, 30H
    MOV AH, 02H
    INT 21H
    LOOP L3
    MOV BL, T2
    MOV T1, BL
    MOV BL, T3
    MOV T2, BL
    POP CX
    LOOP L1

    MOV AH, 4CH
    INT 21H

MAIN ENDP
END MAIN
```


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
Assembling: a2q1.asm
C:\>link a2q1.obj
Microsoft (R) Segmented Executable Linker Version 5.31.009 Jul 13 1992
Copyright (C) Microsoft Corp 1984-1992. All rights reserved.
Run File [a2q1.exe]:
List File [nul.map]:
Libraries [.lib]:
Definitions File [nul.def]:
C:\>a2q1.exe
FIBONACCI SERIES:
0
1
1
2
3
5
8
13
21
34
C:\>

2. Write an assembly language program to search the largest number in an array of ten 8-bit numbers. The array elements will be stored in the data segment.

```
.MODEL SMALL
.STACK 100H
.DATA
ARRAY DB 98, 100, 2, 3, 4, 5, 9, 6, 7, 18

.CODE
MAIN PROC
   MOV AX, @DATA
   MOV DS, AX
   MOV SI, OFFSET ARRAY
   MOV CX, 10
   MOV AX, 0
   MOV DL, [SI]

L1:
   MOV BL, [SI]
   CMP BL, DL
   JLE L2
   MOV DL, BL

L2:
   INC SI
   LOOP L1

   MOV AL, DL

PRINT:
   MOV DX, 0
   MOV BX, 10
   MOV CX, 0

L4:
   DIV BX
   PUSH DX
```

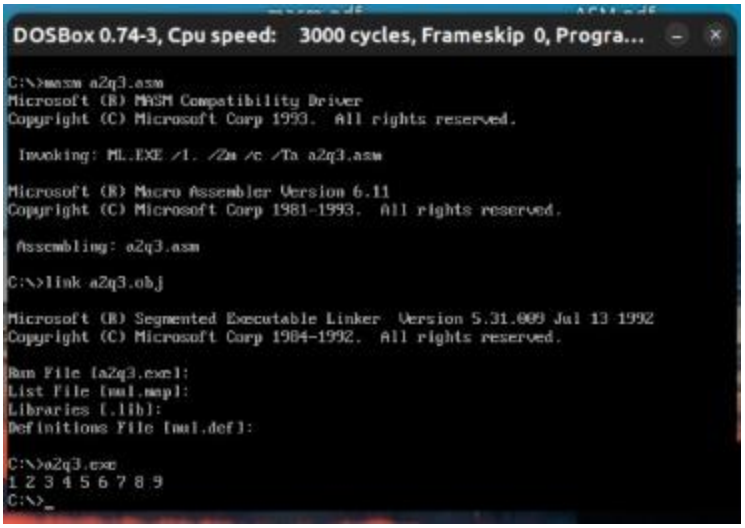
```
    MOV DX, 0
    MOV AH, 0
    INC CX
    CMP AX, 0
    JNE L4

L3:
    POP DX
    ADD DX, 30H
    MOV AH, 02H
    INT 21H
    LOOP L3

    MOV AH, 4CH
    INT 21H

MAIN ENDP
END MAIN
```



3. Write an assembly language program to sort in descending order using bubble sort algorithm a given set of byte sized unsigned numbers in memory.

```
.MODEL SMALL
.STACK 100H
.DATA
ARRAY DB 9, 8, 7, 6, 5, 4, 3, 2, 1

.CODE
MAIN PROC
   MOV AX, @DATA
   MOV DS, AX
   MOV DX, 8

L1:
   MOV CX, DX
   MOV SI, OFFSET ARRAY

L2:
   MOV AL, [SI]
   INC SI
   MOV BL, [SI]
   CMP AL, BL
   JLE L3
   MOV [SI], AL
   DEC SI
   MOV [SI], BL

L3:
   INC SI
   LOOP L2

   DEC DX
   JNZ L1

   MOV SI, OFFSET ARRAY
   MOV CX, 9
```

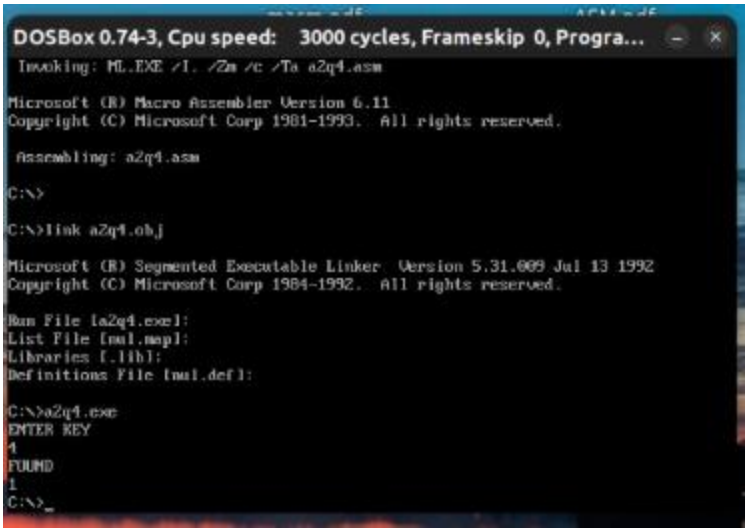
```
L4:
   MOV DL, [SI]
   ADD DL, 30H
   MOV AH, 02H
   INT 21H
   MOV DL, 32
   MOV AH, 02H
   INT 21H
   INC SI
   LOOP L4

   MOV AH, 4CH
   INT 21H

MAIN ENDP
END MAIN
```



4. Write an assembly language program to search for a given 8-bits key using linear search in an array of 10 numbers. The search key will be asked to enter from the keyboard. A message should be displayed indicating whether the search was a success or a failure. If it is a successful case, the position of the number in the array is to be displayed.

```
.MODEL SMALL
.STACK 100H
.DATA
ARR1 DB 1, 4, 2, 5, 6, 7
VAR1 DB "ENTER KEY", 10, 13, "$"
VAR2 DB "FOUND", 10, 13, "$"
VAR3 DB "NOT FOUND", 10, 13, "$"

.CODE
MAIN PROC
   MOV AX, @DATA
   MOV DS, AX
   MOV DX, OFFSET VAR1
   MOV AH, 09H
   INT 21H

   MOV AH, 01H
   INT 21H
   SUB AL, 48
   MOV SI, OFFSET ARR1
   MOV CX, 6

L1:
   MOV BL, [SI]
   CMP AL, BL
   JE PRINT
   INC SI
   LOOP L1

   MOV DX, 10
   MOV AH, 02H
   INT 21H
```

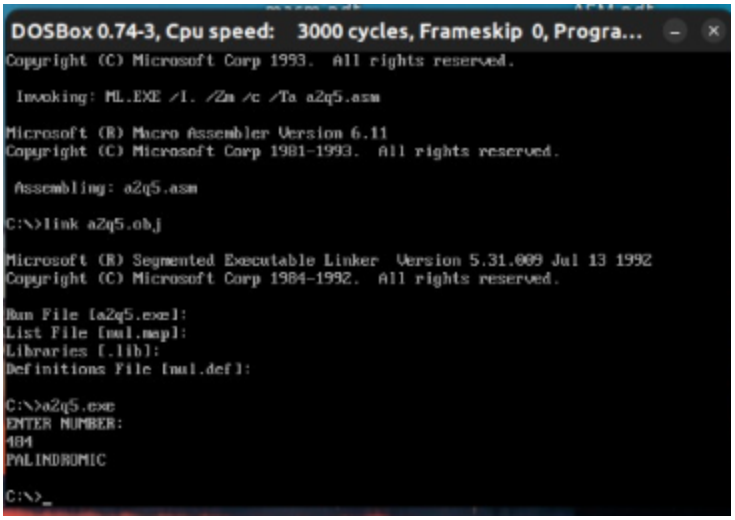
```
    MOV DX, 13
    MOV AH, 02H
    INT 21H

    MOV DX, OFFSET VAR3
    MOV AH, 09H
    INT 21H

    MOV AH, 4CH
    INT 21H

PRINT:
    MOV DX, 10
    MOV AH, 02H
    INT 21H
    MOV DX, 13
    MOV AH, 02H
    INT 21H

    MOV DX, OFFSET VAR2
    MOV AH, 09H
    INT 21H

    MOV BX, 6
    SUB BX, CX
    MOV DX, BX
    ADD DX, 30H
    MOV AH, 02H
    INT 21H

    MOV AH, 4CH
    INT 21H

MAIN ENDP
END MAIN
```


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
Invoking: ML.EXE /I. /Zm /c /Ta a2q4.asm
Microsoft (R) Macro Assembler Version 6.11
Copyright (C) Microsoft Corp 1981-1993. All rights reserved.
Assembling: a2q4.asm
C:\>
C:\>link a2q4.obj
Microsoft (R) Segmented Executable Linker Version 5.31.009 Jul 13 1992
Copyright (C) Microsoft Corp 1984-1992. All rights reserved.
Run File [a2q4.exe]:
List File [nul.map]:
Libraries [.lib]:
Definitions File [nul.def]:
C:\>a2q4.exe
ENTER KEY
4
FUUMD
1
C:\>

5. Write a program to check whether a 16-bit number is a palindrome or not. The number will be entered from the keyboard.

```
.MODEL SMALL
.STACK 100H
.DATA
VAR1    DB "PALINDROMIC", 10, 13, "$"
VAR2    DB "NOT-PALINDROMIC", 10, 13, "$"
VAR3    DB "ENTER NUMBER:", 10, 13, "$"
NUM1    DW ?
NUM2    DW ?

.CODE
MAIN PROC
   MOV AX, @DATA
   MOV DS, AX
   MOV DX, OFFSET VAR3
   MOV AH, 09H
   INT 21H
   MOV CX, 10
   MOV BX, 0

INPUT:
   MOV AH, 01H
   INT 21H
   CMP AL, 13
   JE TER
   SUB AL, 30H
   MOV AH, 0
   PUSH AX
   MOV AX, BX
   MUL CX
   MOV BX, AX
   POP AX
   ADD BX, AX
   JMP INPUT

TER:
```

```
    MOV NUM1, BX
    MOV DX, 0
    MOV CX, 10
    MOV AX, 0

PALINDROME:
    XCHG BX, AX
    DIV CX
    XCHG BX, AX
    ADD AX, DX
    CMP BX, 0
    MOV NUM2, AX
    JE RESULT
    MUL CX
    JMP PALINDROME

RESULT:
    MOV DX, 0
    MOV DX, NUM1
    MOV AX, NUM2
    SUB DX, AX
    CMP DX, 0
    JE SUCCESS
    MOV DX, OFFSET VAR2
    MOV AH, 09H
    INT 21H
    MOV AH, 4CH
    INT 21H

SUCCESS:
    MOV DX, OFFSET VAR1
    MOV AH, 09H
    INT 21H
    MOV AH, 4CH
    INT 21H

MAIN ENDP
END MAIN
```


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
Copyright (C) Microsoft Corp 1993. All rights reserved.
Invoking: ML.EXE /I. /Zm /c /Ta a2q5.asm
Microsoft (R) Macro Assembler Version 6.11
Copyright (C) Microsoft Corp 1981-1993. All rights reserved.
Assembling: a2q5.asm
C:\>link a2q5.obj
Microsoft (R) Segmented Executable Linker Version 5.31.009 Jul 13 1992
Copyright (C) Microsoft Corp 1984-1992. All rights reserved.
Run File [a2q5.exe]:
List File [nul.map]:
Libraries [.lib]:
Definitions File [nul.def]:
C:\>a2q5.exe
ENTER NUMBER:
484
PALINDROMIC
C:\>

6. Write a program to display the G.C.D. of two numbers M and N. Assume that the variables M and N are declared and initialized in the data segment.

```
.MODEL SMALL
.STACK 100H
.DATA
NUM1 DB 90
NUM2 DB 25

.CODE
MAIN PROC
   MOV AX, @DATA
   MOV DS, AX
   MOV AX, 0
   MOV BX, 0
   MOV AL, NUM1
   MOV BL, NUM2

GCD:
   MOV CX, 0
   CMP BL, 0
   JE PRINT
   DIV BL
   MOV CL, AH
   MOV CH, BL
   MOV BL, CL
   MOV AX, 0
   MOV AL, CH
   JMP GCD

PRINT:
   MOV DX, 0
   MOV DL, AL
   ADD DL, 48
   MOV AH, 02H
   INT 21H

   MOV AH, 4CH
```

INT 21H

MAIN ENDP
END MAIN

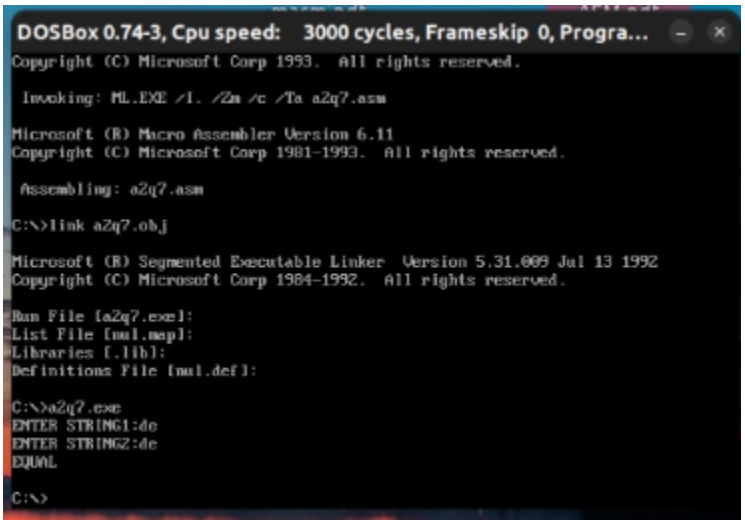
```
C:\>masm gcd.asm
Microsoft (R) Macro Assembler Version 5.00
Copyright (C) Microsoft Corp 1981-1985, 1987.  All rights reserved.

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

  51718 + 464826 Bytes symbol space free

      0 Warning Errors
      0 Severe  Errors

C:\>link gcd.obj

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

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

C:\>gcd.exe
5
C:\>_
```

7. Write an assembly language program to compare two strings.

```
.MODEL SMALL
.STACK 100H
.DATA
VAR1    DB "ENTER STRING1:", "$"
VAR2    DB "ENTER STRING2:", "$"
STR1    DB 50 DUP('$')
STR2    DB 50 DUP('$')
VAR3    DB "EQUAL", 10, 13, "$"
VAR4    DB "NOT EQUAL", 10, 13, "$"
LEN1    DW ?
LEN2    DW ?

.CODE
MAIN PROC
   MOV AX, @DATA
   MOV DS, AX
   MOV DX, OFFSET VAR1
   MOV AH, 09H
   INT 21H
   MOV CX, 0
   MOV SI, OFFSET STR1
   CALL TEXT
   MOV LEN1, CX
   MOV DX, OFFSET VAR2
   MOV AH, 09H
   INT 21H
   MOV CX, 0
   MOV SI, OFFSET STR2
   CALL TEXT2
   MOV LEN2, CX
   MOV CX, 0
   MOV SI, OFFSET STR1
   MOV DI, OFFSET STR2

COMPARE:
   MOV AX, LEN1
```

```
    MOV BX, LEN2
    CMP AX, BX
    JNE KILL
    MOV CX, LEN1

L1:
    MOV AL, [SI]
    MOV BL, [DI]
    CMP AL, BL
    JNE KILL
    INC SI
    INC DI
    LOOP L1

    MOV DX, OFFSET VAR3
    MOV AH, 09H
    INT 21H
    MOV AH, 4CH
    INT 21H

KILL:
    MOV DX, OFFSET VAR4
    MOV AH, 09H
    INT 21H
    MOV AH, 4CH
    INT 21H

MAIN ENDP

TEXT PROC
INPUT:
    MOV AH, 01H
    INT 21H
    CMP AL, 13
    JE TER
    MOV [SI], AL
    INC SI
    INC CX
    JMP INPUT
```

```
TER:
   MOV [SI], '$ '
   RET
TEXT ENDP

TEXT2 PROC
INPUT2:
   MOV AH, 01H
   INT 21H
   CMP AL, 13
   JE TER2
   MOV [SI], AL
   INC SI
   INC CX
   JMP INPUT2

TER2:
   MOV [SI], '$ '
   RET
TEXT2 ENDP

END MAIN
```

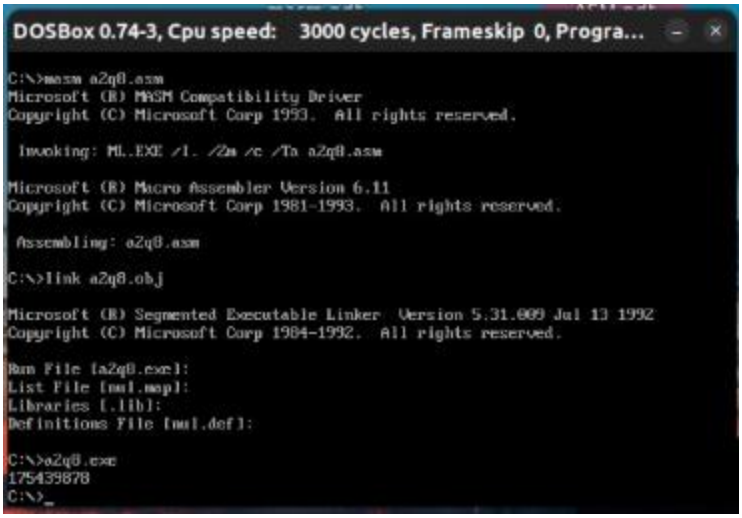


8. Write a program to add two 32-bit numbers and store the result in consecutive memory locations.

```
.MODEL SMALL
.STACK 100H
.DATA
NUM1 DD 78119990H
NUM2 DD 97319888H
RES DD ?

.CODE
MAIN PROC
   MOV AX, @DATA
   MOV DS, AX

   ; Calculate the sum of two double words
   MOV AX, 0
   MOV BX, 0

   ; Calculate the least significant byte
   MOV AL, BYTE PTR NUM1
   MOV BL, BYTE PTR NUM2
   ADD AL, BL
   DAA
   MOV BYTE PTR RES + 3, AL

   ; Calculate the second least significant byte
   MOV AX, 0
   MOV BX, 0
   MOV AL, BYTE PTR NUM1 + 1
   MOV BL, BYTE PTR NUM2 + 1
   JC L5
L5:
   ADD AL, 01H
   ADD AL, BL
   DAA
   MOV BYTE PTR RES + 2, AL
```

```
    ; Calculate the second most significant byte
    MOV AX, 0
    MOV BX, 0
    MOV AL, BYTE PTR NUM1 + 2
    MOV BL, BYTE PTR NUM2 + 2
    JC L7
L7:
    ADD AL, 01H
    ADD AL, BL
    DAA
    MOV BYTE PTR RES + 1, AL

    ; Calculate the most significant byte
    MOV AX, 0
    MOV BX, 0
    MOV AL, BYTE PTR NUM1 + 3
    MOV BL, BYTE PTR NUM2 + 3
    ADD AL, BL
    DAA
    MOV BYTE PTR RES, AL

    ; Check for carry in the most significant byte
    JNC L8
    MOV DX, 01H
    ADD DX, 30H
    MOV AH, 02H
    INT 21H

L8:
    ; Print the result
    MOV AX, 0
    MOV AL, BYTE PTR RES
    CALL PRINT
    MOV AX, 0
    MOV AL, BYTE PTR RES + 1
    CALL PRINT
    MOV AX, 0
    MOV AL, BYTE PTR RES + 2
    CALL PRINT
```

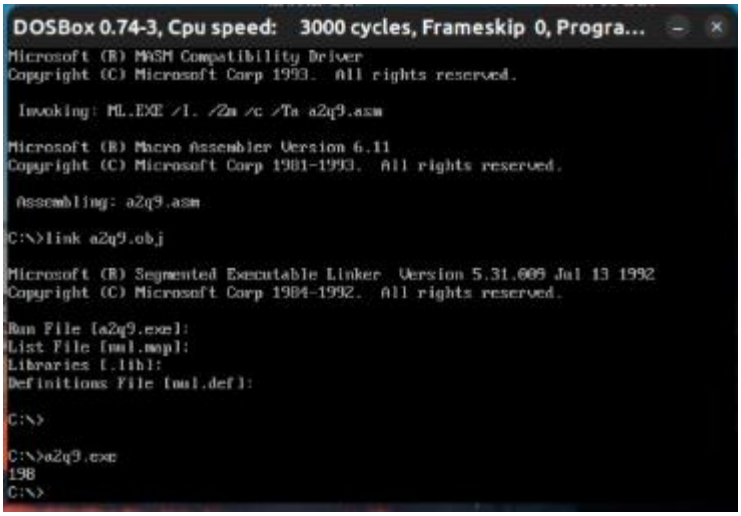
```
    MOV AX, 0
    MOV AL, BYTE PTR RES + 3
    CALL PRINT

    ; Exit the program
    MOV AH, 4CH
    INT 21H

MAIN ENDP

PRINT PROC
    ; Convert and print a single byte
    MOV DX, 0
    MOV BX, 10H
    MOV CX, 0

L3:
    DIV BX
    PUSH DX
    MOV DX, 0
    INC CX
    CMP AX, 0
    JNE L3

L4:
    POP DX
    ADD DX, 48
    MOV AH, 02H
    INT 21H
    LOOP L4
    RET

PRINT ENDP

END MAIN
```


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
C:\>masm a2q8.asm
Microsoft (R) MASM Compatibility Driver
Copyright (C) Microsoft Corp 1993. All rights reserved.
Invoking: ML.EXE /I. /Zm /c /Ta a2q8.asm
Microsoft (R) Macro Assembler Version 6.11
Copyright (C) Microsoft Corp 1981-1993. All rights reserved.
Assembling: a2q8.asm
C:\>link a2q8.obj
Microsoft (R) Segmented Executable Linker Version 5.31.009 Jul 13 1992
Copyright (C) Microsoft Corp 1984-1992. All rights reserved.
Run File [a2q8.exe]:
List File [nul.map]:
Libraries [.lib]:
Definitions File [nul.def]:
C:\>a2q8.exe
175439878
C:\>

9. Assume that two variables x and y are stored in packed BCD format. Write an 8086 alp to add x and y using DAA and display the result in packed BCD format also. Do the same addition without using DAA.

```
.MODEL SMALL
.STACK 100H
.DATA
NUM1 DW ?

.CODE
MAIN PROC
    MOV AX, @DATA
    MOV DS, AX

    ; Initialize AX, AL, and BL
    MOV AX, 0
    MOV AL, 99H
    MOV BL, 99H

    ; Add AL and BL with decimal adjust
    ADD AL, BL
    DAA
    MOV NUM1, AX

    ; Check for carry
    JC L5

L5:
    ; Convert and print the result
    MOV DX, 1
    ADD DX, 30H
    MOV AH, 02H
    INT 21H

    CALL PRINT

    ; Exit the program
    MOV AH, 4CH
```

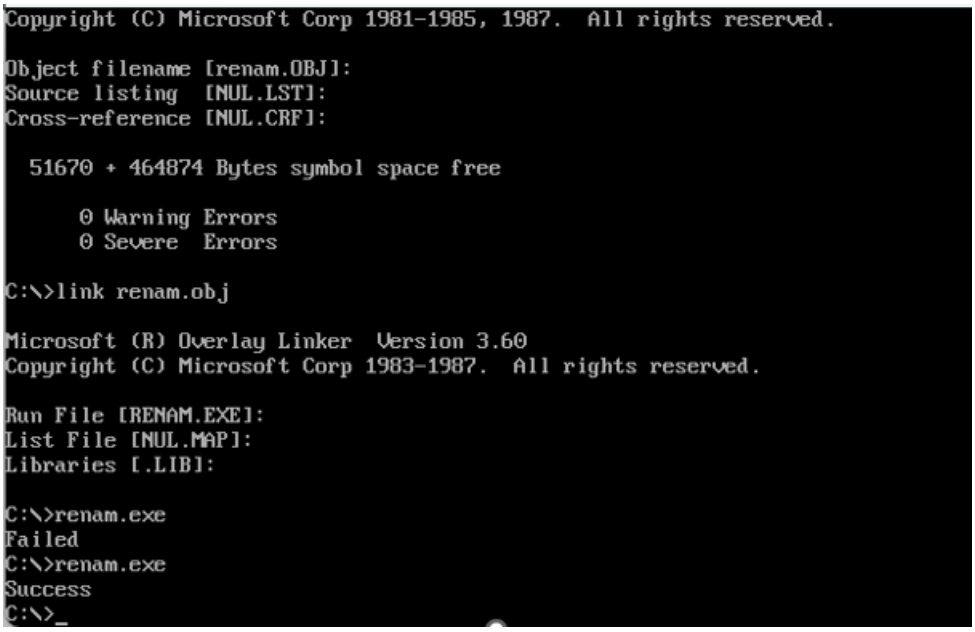
```
    INT 21H

MAIN ENDP

PRINT PROC
    ; Convert and print a word
    MOV AX, NUM1
    MOV DX, 0
    MOV BX, 16
    MOV CX, 0

L3:
    DIV BX
    PUSH DX
    MOV DX, 0
    INC CX
    CMP AX, 0
    JNE L3

L4:
    POP DX
    ADD DX, 48
    MOV AH, 02H
    INT 21H
    LOOP L4
    RET

PRINT ENDP

END MAIN
```


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
Microsoft (R) MASM Compatibility Driver
Copyright (C) Microsoft Corp 1993. All rights reserved.
Invoking: ML.EXE /I. /Zm /c /Ta a2q9.asm
Microsoft (R) Macro Assembler Version 6.11
Copyright (C) Microsoft Corp 1981-1993. All rights reserved.
Assembling: a2q9.asm
C:\>link a2q9.obj
Microsoft (R) Segmented Executable Linker Version 5.31.009 Jul 13 1992
Copyright (C) Microsoft Corp 1984-1992. All rights reserved.
Run File [a2q9.exe]:
List File [nul.map]:
Libraries [.lib]:
Definitions File [nul.def]:
C:\>
C:\>a2q9.exe
198
C:\>

10. Write an 8086 alp to rename a file, if it exists, using DOS interrupt. Otherwise display an error message.

```
.MODEL SMALL
.STACK 100H
.DATA
FILE1   DB "old.txt", 0
FILE2   DB "new.txt", 0
FAIL    DB "Failed$"
PASS    DB "Success$"

.CODE
MAIN PROC
   MOV AX, @DATA
   MOV DS, AX
   MOV ES, AX

   MOV DX, OFFSET FILE1
   MOV DI, OFFSET FILE2
   MOV AH, 56H
   INT 21H
   JC ERROR

   MOV DX, OFFSET PASS
   MOV AH, 09H
   INT 21H
   JMP end1

ERROR:
   MOV DX, OFFSET FAIL
   MOV AH, 09H
   INT 21H

end1:
   MOV AH, 4CH
   INT 21H

MAIN ENDP
```

END MAIN

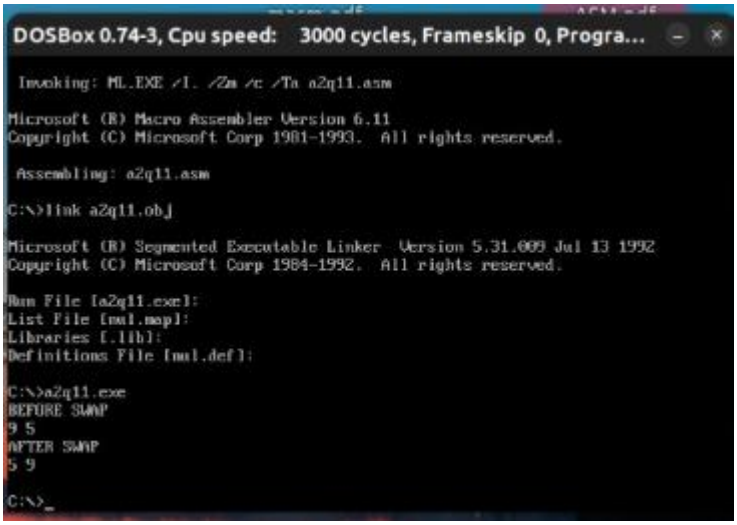

Copyright (C) Microsoft Corp 1981-1985, 1987. All rights reserved.
Object filename [renam.OBJ]:
Source listing [NUL.LST]:
Cross-reference [NUL.CRF]:
51670 + 464874 Bytes symbol space free
0 Warning Errors
0 Severe Errors
C:\>link renam.obj
Microsoft (R) Overlay Linker Version 3.60
Copyright (C) Microsoft Corp 1983-1987. All rights reserved.
Run File [RENAM.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
C:\>renam.exe
Failed
C:\>renam.exe
Success
C:\>_

11. Write a swap procedure that accepts the address of two words, and it exchanges the contents of those words. Write a program to initialize two variables and after the execution of the swap, the procedure displays the contents of the words. (Parameter passing needs to be done).

```
.MODEL SMALL
.STACK 100H
.DATA
VAR1    DB "BEFORE SWAP", 10, 13, "$"
VAR2    DB "AFTER SWAP", 10, 13, "$"
NUM1    DB 9
NUM2    DB 5

.CODE
MAIN PROC
   MOV AX, @DATA
   MOV DS, AX

   ; Define a macro to swap the values of NUM1 and NUM2
   SWAP MACRO NUM1, NUM2
      MOV BL, NUM1
      MOV CL, NUM2
      XCHG BL, CL
      MOV NUM1, BL
      MOV NUM2, CL
   ENDM

   ; Display "BEFORE SWAP"
   MOV DX, OFFSET VAR1
   MOV AH, 09H
   INT 21H

   ; Display the value of NUM1
   MOV DL, NUM1
   ADD DL, 30H
   MOV AH, 02H
   INT 21H
```

```
; Display a space
MOV DL, 32
MOV AH, 02H
INT 21H

; Display the value of NUM2
MOV DL, NUM2
ADD DL, 30H
MOV AH, 02H
INT 21H

; Display newline characters
MOV DL, 10
MOV AH, 02H
INT 21H

MOV DL, 13
MOV AH, 02H
INT 21H

; Use the SWAP macro to swap the values of NUM1 and NUM2
SWAP NUM1, NUM2

; Display "AFTER SWAP"
MOV DX, OFFSET VAR2
MOV AH, 09H
INT 21H

; Display the value of NUM1 (previously NUM2)
MOV DL, NUM1
ADD DL, 30H
MOV AH, 02H
INT 21H

; Display a space
MOV DL, 32
MOV AH, 02H
INT 21H
```

```
    ; Display the value of NUM2 (previously NUM1)
    MOV DL, NUM2
    ADD DL, 30H
    MOV AH, 02H
    INT 21H

    ; Display newline characters
    MOV DL, 10
    MOV AH, 02H
    INT 21H

    MOV DL, 13
    MOV AH, 02H
    INT 21H

    ; Exit the program
    MOV AH, 4CH
    INT 21H

MAIN ENDP
END MAIN
```



12. Write an assembly language program to multiply two 3x3 matrices of signed 8-bit integers. Display result. Assume that each of the elements of the product matrix can be stored in an 8-bit location.

```
.MODEL SMALL
.STACK 100H
.DATA
M1    DB 1,1,1,1,1,1,1,1,1
M2    DB 1,1,1,1,1,1,1,1,1
PROD  DB 0,0,0,0,0,0,0,0,0

.CODE
MAIN PROC
   MOV AX, @DATA
   MOV DS, AX
   MOV BP, OFFSET PROD
   MOV SI, OFFSET M1
   MOV DI, OFFSET M2
   MOV CH, 03H ; matrix dimension
   MOV CL, 03H

L1:
   MOV BL, CH

L2:
   MOV DL, 0
   MOV DH, CH

L3:
   MOV AL, [SI]
   MOV AH, [DI]
   MUL AH
   ADD DL, AL
   INC SI
   ADD DI, 03H
   DEC DH
   CMP DH, 0
   JNZ L3
```

```
    MOV [BP], DL
    INC BP
    SUB SI, 03H
    SUB DI, 09H
    INC DI
    DEC BL
    CMP BL, 0
    JNZ L2

    ADD SI, 03H
    MOV DI, OFFSET M2
    DEC CL
    CMP CL, 0
    JNZ L1

    MOV CX, 9
    MOV BP, OFFSET PROD

PRINT:
    MOV DX, [BP]
    ADD DX, 48
    MOV AH, 02H
    INT 21H
    INC BP
    LOOP PRINT

    ; Exit the program
    MOV AH, 4CH
    INT 21H

MAIN ENDP
END MAIN
```

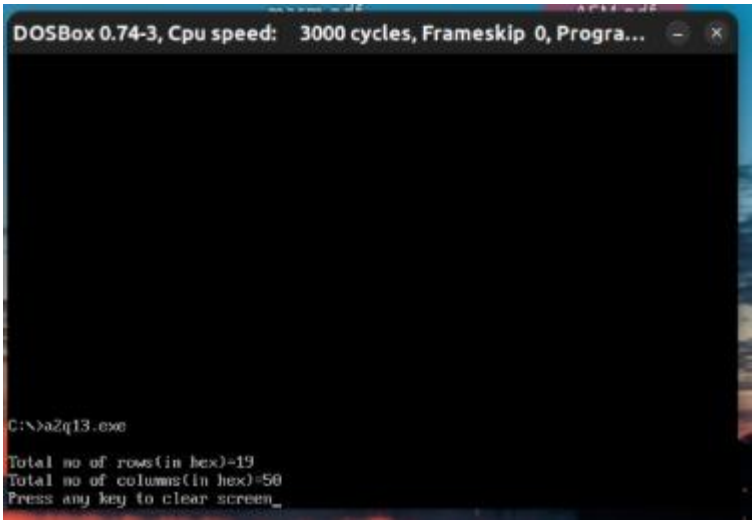

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
Invoking: ML.EXE /I. /Zm /c /Ta a2q12.asm
Microsoft (R) Macro Assembler Version 6.11
Copyright (C) Microsoft Corp 1981-1993. All rights reserved.
Assembling: a2q12.asm
C:\>
C:\>
C:\>link a2q12.obj
Microsoft (R) Segmented Executable Linker Version 5.31.009 Jul 13 1992
Copyright (C) Microsoft Corp 1984-1992. All rights reserved.
Run File [a2q12.exe]:
List File [nul.map]:
Libraries [.lib]:
Definitions File [nul.def]:
C:\>a2q12.exe
333333333
C:\>

13. Write an assembly language program to get the screen width (no of cols) using BIOS interrupt and calculate the no. of rows from the appropriate word location in BIOS data area and clear the screen using BIOS interrupt.

```
.MODEL SMALL
.STACK 64
.DATA
BYTES DD 0040004CH
ROWS DB ?
COLS DB ?
MSG1 DB 0DH, 0AH, 'Total no of rows(in hex)=$'
MSG2 DB 0DH, 0AH, 'Total no of columns(in hex)=$'
MSG3 DB 0DH, 0AH, 'Press any key to clear screen$'
HEXCODE DB '0123456789ABCDEF'

.CODE
DISPLAY PROC
   PUSH AX
   PUSH BX
   PUSH CX
   PUSH DX

   LEA DX, MSG1
   MOV AH, 09H
   INT 21H

   MOV AL, ROWS
   MOV CL, 10H
   MOV AH, 00H
   DIV CL
   MOV BL, AL
   MOV DL, HEXCODE[BX]
   PUSH AX
   MOV AH, 02H
   INT 21H
   POP AX
   MOV BL, AH
   MOV DL, HEXCODE[BX]
```

```
    MOV AH, 02H
    INT 21H

    LEA DX, MSG2
    MOV AH, 09H
    INT 21H

    MOV AL, COLS
    MOV CL, 10H
    MOV AH, 00H
    MOV BH, 00H
    DIV CL
    MOV BL, AL
    MOV DL, HEXCODE[BX]
    PUSH AX
    MOV AH, 02H
    INT 21H
    POP AX
    MOV BL, AH
    MOV DL, HEXCODE[BX]
    MOV AH, 02H
    INT 21H

    POP DX
    POP CX
    POP BX
    POP AX
    RET
DISPLAY ENDP

MAIN:
    MOV AX, @DATA
    MOV DS, AX
    MOV AH, 0FH
    INT 10H
    MOV COLS, AH
    MOV CL, AH
    MOV CH, 0
    PUSH DS
```

```
    LDS SI, BYTES
    MOV AX, [SI]
    POP DS
    SHR AX, 1
    DIV CL
    MOV ROWS, AL
    CALL DISPLAY
    LEA DX, MSG3
    MOV AH, 09H
    INT 21H
    MOV AH, 01H
    INT 21H
    MOV DH, 0

AGAIN:
    MOV BH, 0
    MOV DL, 0
    MOV AH, 02H
    INT 10H
    MOV BL, 0
    MOV AL, 'X'
    MOV AH, 09H
    INT 10H
    INC DH
    CMP DH, ROWS
    JB AGAIN

    MOV AH, 4CH
    INT 21H
END MAIN
```


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
C:\>a2q13.exe
Total no of rows(in hex)=19
Total no of columns(in hex)=50
Press any key to clear screen_