Experiment No 5: Matrix Operations

Date: 11-10-2020 **NAME: A Susmithaa Raam**
REG.NO: 185001181

A. AIM:

Program for Matrix addition.

ALGORITHM:

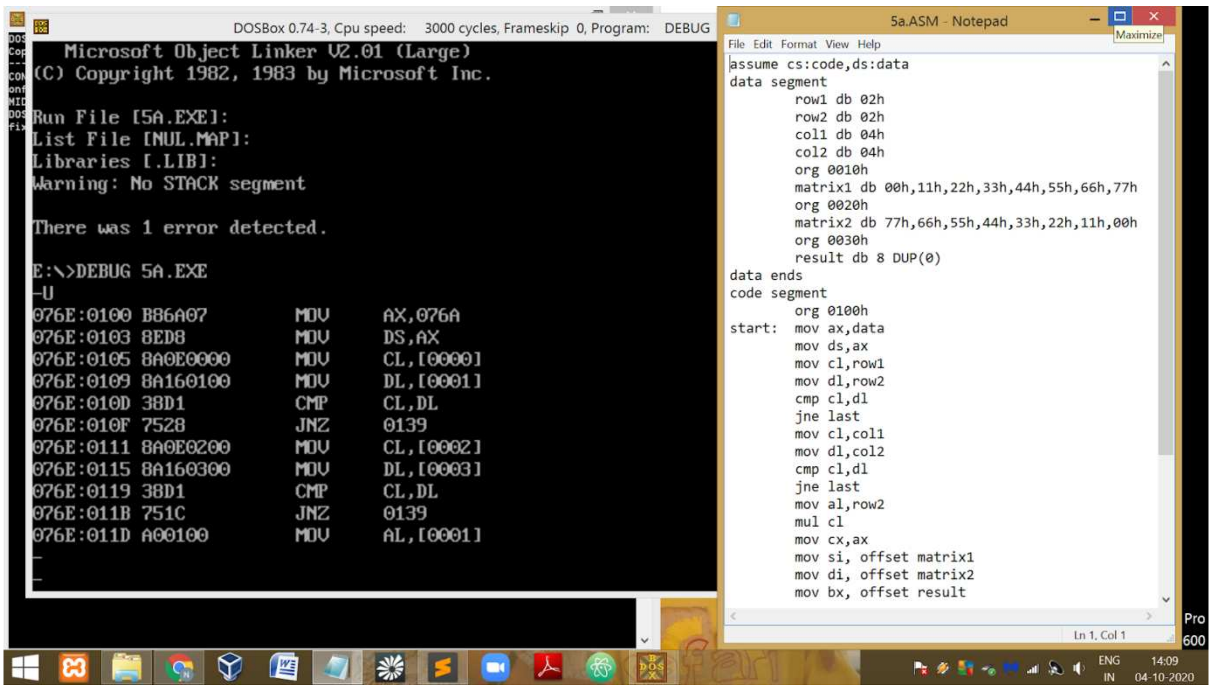
- Initialize the data segment.
- Move data segment address to ds
- Load row1 to cl, row2 to dl.
- Compare cl and dl and terminate if not equal.
- Load col1 to cl, col2 to dl.
- Compare cl and dl and terminate if not equal.
- Move row2 to al.
- Multiply al with cl and move ax to cx.
- Move offset of matrix1 to si, matrix2 to di, result to bx
- Loop here:
  - Move contents pointed by si to al and add al and contents pointed by di.
  - Move al to result matrix
  - Increment si,di,bl
- Terminate the program

PROGRAM:

PROGRAM	COMMENTS
assume cs:code,ds:data	
data segment	
row1 db 02h	
row2 db 02h	
col1 db 04h	
col2 db 04h	
org 0010h	
matrix1 db 00h,11h,22h,33h,44h,55h,66h,77h	
org 0020h	
matrix2 db 77h,66h,55h,44h,33h,22h,11h,00h	
org 0030h	
result db 8 DUP(0)	
data ends	
code segment	
org 0100h	
start:	
mov ax,data	Load data segment to ds

mov ds,ax	
mov cl,row1	Load row1 value to cl
mov dl,row2	Load row2 value to dl
cmp cl,dl	Compare cl and dl
jne last	Jump to last if not equal
mov cl,col1	Load col1 value to cl
mov dl,col2	Load col2 value to dl
cmp cl,dl	Compare cl and dl
jne last	Jump to last if not equal
mov al,row2	Load row2 value to al
mul cl	Multiply al with cl
mov cx,ax	Load value of ax to cx
mov si, offset matrix1	Load offset of matrix1 to si
mov di, offset matrix2	Load offset of matrix2 to di
mov bx, offset result	Load offset of result to bx
Here:	cx register indicates the loop
mov al, [si]	count
add al, [di]	Load contents pointed by si to al
mov [bx], al	Add all with contents pointed by
inc si	di
inc di	Load al to result matrix
inc bl	Increment si
loop here	Increment di
	Increment bl
last:	
mov ah,4ch	Terminate the program
int 21h	

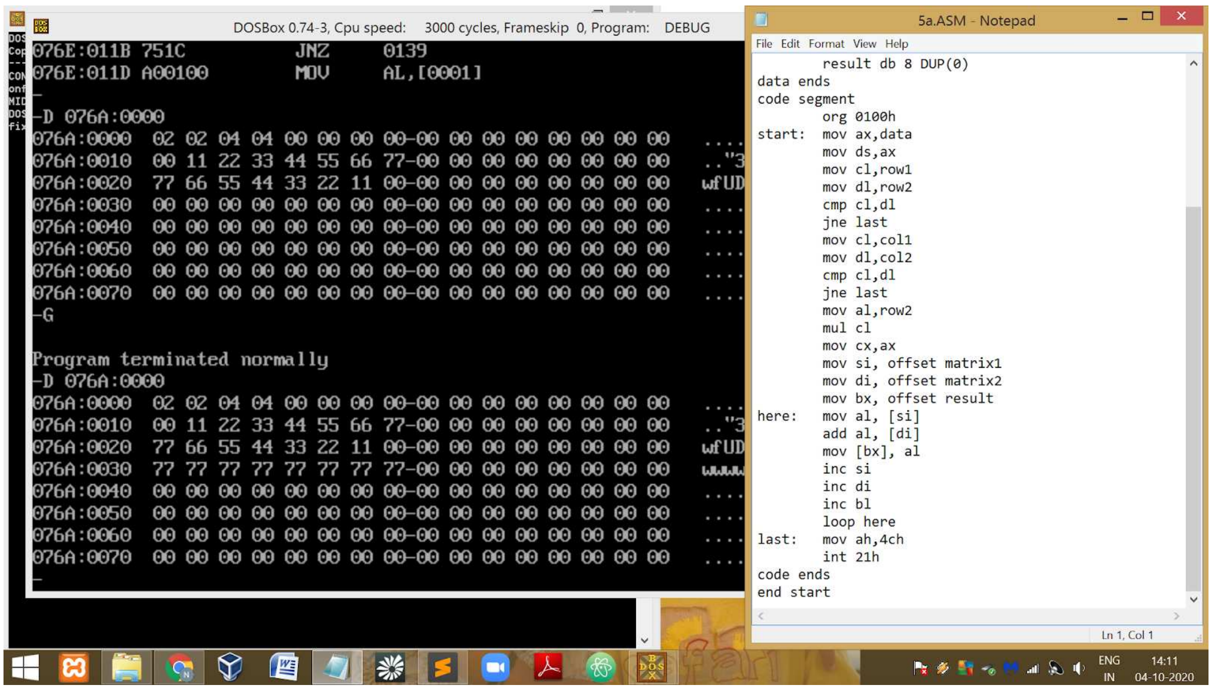
UNASSEMBLED CODE:


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG
5a.ASM - Notepad
File Edit Format View Help
Maximize
Ln 1, Col 1
ENG IN 14:09 04-10-2020

```
 Microsoft Object Linker V2.01 (Large)
(C) Copyright 1982, 1983 by Microsoft Inc.

Run File [5A.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
Warning: No STACK segment

There was 1 error detected.

E:\>DEBUG 5A.EXE
-U
076E:0100 B86A07        MOV     AX,076A
076E:0103 8ED8          MOV     DS,AX
076E:0105 8A0E0000      MOV     CL,[0000]
076E:0109 8A160100      MOV     DL,[0001]
076E:010D 38D1          CMP     CL,DL
076E:010F 7528          JNZ     0139
076E:0111 8A0E0200      MOV     CL,[0002]
076E:0115 8A160300      MOV     DL,[0003]
076E:0119 38D1          CMP     CL,DL
076E:011B 751C          JNZ     0139
076E:011D A00100        MOV     AL,[0001]
-
```

```
assume cs:code,ds:data
data segment
        row1 db 02h
        row2 db 02h
        col1 db 04h
        col2 db 04h
        org 0010h
        matrix1 db 00h,11h,22h,33h,44h,55h,66h,77h
        org 0020h
        matrix2 db 77h,66h,55h,44h,33h,22h,11h,00h
        org 0030h
        result db 8 DUP(0)
data ends
code segment
        org 0100h
start:  mov ax,data
        mov ds,ax
        mov cl,row1
        mov dl,row2
        cmp cl,dl
        jne last
        mov cl,col1
        mov dl,col2
        cmp cl,dl
        jne last
        mov al,row2
        mul cl
        mov cx,ax
        mov si, offset matrix1
        mov di, offset matrix2
        mov bx, offset result
```

SAMPLE INPUT/OUTPUT:


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG
5a.ASM - Notepad
File Edit Format View Help
Ln 1, Col 1
ENG IN 14:11 04-10-2020

```
076E:011B 751C          JNZ     0139
076E:011D A00100        MOV     AL,[0001]
-
-D 076A:0000
076A:0000  02 02 04 04 00 00 00 00-00 00 00 00 00 00 00 00   ....
076A:0010  00 11 22 33 44 55 66 77-00 00 00 00 00 00 00 00   .."3
076A:0020  77 66 55 44 33 22 11 00-00 00 00 00 00 00 00 00   wfUD
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ....
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ....
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ....
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ....
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ....
-G

Program terminated normally
-D 076A:0000
076A:0000  02 02 04 04 00 00 00 00-00 00 00 00 00 00 00 00   ....
076A:0010  00 11 22 33 44 55 66 77-00 00 00 00 00 00 00 00   .."3
076A:0020  77 66 55 44 33 22 11 00-00 00 00 00 00 00 00 00   wfUD
076A:0030  77 77 77 77 77 77 77 77-00 00 00 00 00 00 00 00   wwww
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ....
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ....
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ....
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ....
-
```

```
        result db 8 DUP(0)
data ends
code segment
        org 0100h
start:  mov ax,data
        mov ds,ax
        mov cl,row1
        mov dl,row2
        cmp cl,dl
        jne last
        mov cl,col1
        mov dl,col2
        cmp cl,dl
        jne last
        mov al,row2
        mul cl
        mov cx,ax
        mov si, offset matrix1
        mov di, offset matrix2
        mov bx, offset result
here:   mov al, [si]
        add al, [di]
        mov [bx], al
        inc si
        inc di
        inc bl
        loop here
last:   mov ah,4ch
        int 21h
code ends
end start
```

RESULT:

Thus matrix addition has been achieved.

B. AIM:

Program for matrix subtraction.

ALGORITHM:

- Initialize the data segment.
- Move data segment address to ds

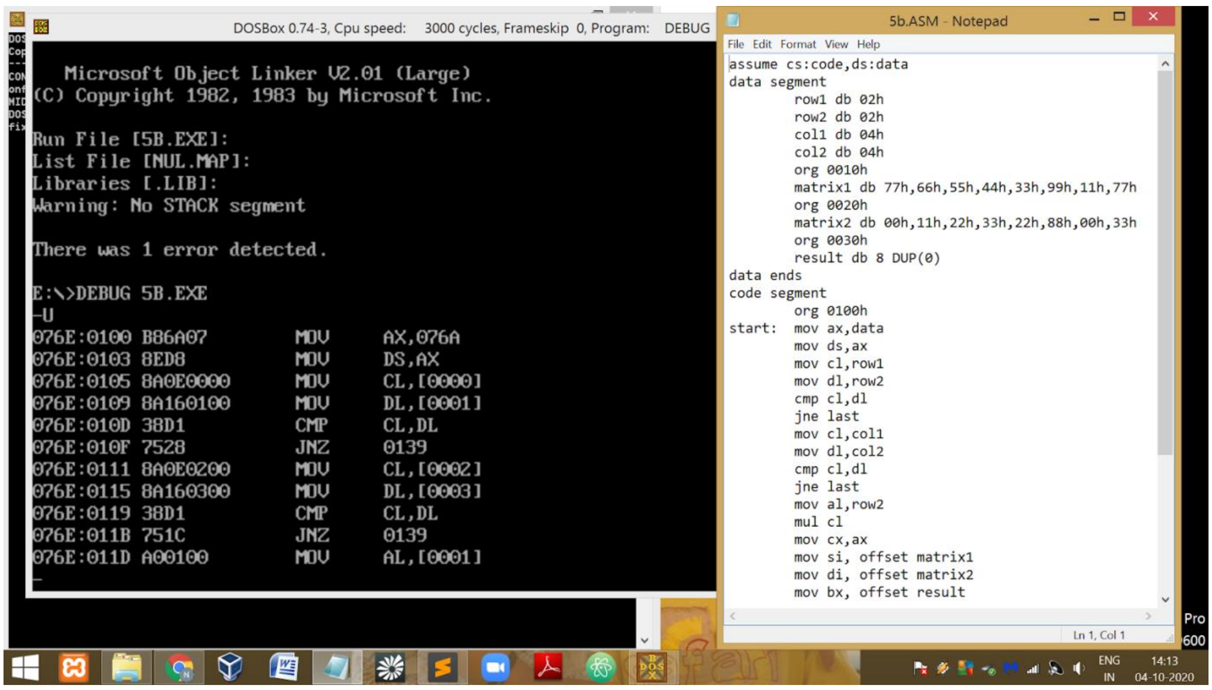
- Load row1 to cl, row2 to dl.
- Compare cl and dl and terminate if not equal.
- Load col1 to cl, col2 to dl.
- Compare cl and dl and terminate if not equal.
- Move row2 to al.
- Multiply al with cl and move ax to cx.
- Move offset of matrix1 to si, matrix2 to di, result to bx
- Loop here:
    - Move contents pointed by si to al and subtract al and contents pointed by di from al.
    - Move al to result matrix
    - Increment si,di,bl
- Terminate the program

PROGRAM:

PROGRAM	**COMMENTS**
assume cs:code,ds:data	
data segment	
row1 db 02h	
row2 db 02h	
col1 db 04h	
col2 db 04h	
org 0010h	
matrix1 db	
77h,66h,55h,44h,33h,99h,11h,77h	
org 0020h	
matrix2 db	
00h,11h,22h,33h,22h,88h,00h,33h	
org 0030h	
result db 8 DUP(0)	
data ends	
code segment	
org 0100h	
start: mov ax,data	Load data segment to ds
mov ds,ax	
mov cl,row1	Load row1 value to cl
mov dl,row2	Load row2 value to dl
cmp cl,dl	Compare cl and dl
jne last	Jump to last if not equal
mov cl,col1	Load col1 value to cl
mov dl,col2	Load col2 value to dl
cmp cl,dl	Compare cl and dl
jne last	Jump to last if not equal
mov al,row2	Load row2 value to al
mul cl	Multiply al with cl
mov cx,ax	Load value of ax to cx

mov si, offset matrix1 mov di, offset matrix2 mov bx, offset result	Load offset of matrix1 to si Load offset of matrix2 to di Load offset of result to bx
Here: mov al, [si] add al, [di] mov [bx], al inc si inc di inc bl loop here	cx register indicates the loop count Load contents pointed by si to al Add all with contents pointed by di Load al to result matrix Increment si Increment di Increment bl
last: mov ah,4ch int 21h	Terminate the program

UNASSEMBLED CODE:



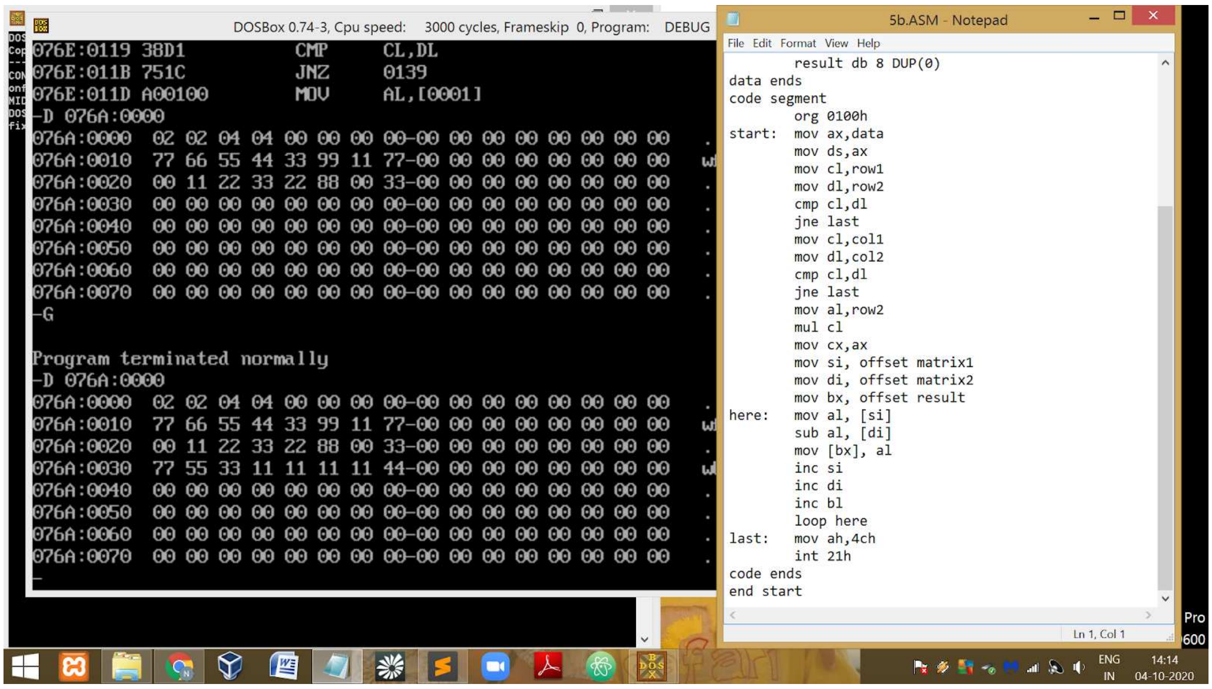
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG

```
  Microsoft Object Linker V2.01 (Large)
(C) Copyright 1982, 1983 by Microsoft Inc.

Run File [5B.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
Warning: No STACK segment

There was 1 error detected.

E:\>DEBUG 5B.EXE
-U
076E:0100 B86A07        MOV     AX,076A
076E:0103 8ED8          MOV     DS,AX
076E:0105 8A0E0000      MOV     CL,[0000]
076E:0109 8A160100      MOV     DL,[0001]
076E:010D 38D1          CMP     CL,DL
076E:010F 7528          JNZ     0139
076E:0111 8A0E0200      MOV     CL,[0002]
076E:0115 8A160300      MOV     DL,[0003]
076E:0119 38D1          CMP     CL,DL
076E:011B 751C          JNZ     0139
076E:011D A00100        MOV     AL,[0001]
-
```

5b.ASM - Notepad

```
assume cs:code,ds:data
data segment
        row1 db 02h
        row2 db 02h
        col1 db 04h
        col2 db 04h
        org 0010h
        matrix1 db 77h,66h,55h,44h,33h,99h,11h,77h
        org 0020h
        matrix2 db 00h,11h,22h,33h,22h,88h,00h,33h
        org 0030h
        result db 8 DUP(0)
data ends
code segment
        org 0100h
start:  mov ax,data
        mov ds,ax
        mov cl,row1
        mov dl,row2
        cmp cl,dl
        jne last
        mov cl,col1
        mov dl,col2
        cmp cl,dl
        jne last
        mov al,row2
        mul cl
        mov cx,ax
        mov si, offset matrix1
        mov di, offset matrix2
        mov bx, offset result
```

SAMPLE INPUT/OUTPUT


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG
5b.ASM - Notepad
File Edit Format View Help
Ln 1, Col 1
ENG IN
14:14 04-10-2020

```
076E:0119 38D1          CMP     CL,DL
076E:011B 751C          JNZ     0139
076E:011D A00100        MOV     AL,[0001]
-D 076A:0000
076A:0000  02 02 04 04 00 00 00 00-00 00 00 00 00 00 00 00   .
076A:0010  77 66 55 44 33 99 11 77-00 00 00 00 00 00 00 00   w
076A:0020  00 11 22 33 22 88 00 33-00 00 00 00 00 00 00 00   .
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   .
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   .
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   .
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   .
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   .
-G

Program terminated normally
-D 076A:0000
076A:0000  02 02 04 04 00 00 00 00-00 00 00 00 00 00 00 00   .
076A:0010  77 66 55 44 33 99 11 77-00 00 00 00 00 00 00 00   w
076A:0020  00 11 22 33 22 88 00 33-00 00 00 00 00 00 00 00   .
076A:0030  77 55 33 11 11 11 11 44-00 00 00 00 00 00 00 00   w
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   .
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   .
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   .
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   .
-
```

```
        result db 8 DUP(0)
data ends
code segment
        org 0100h
start:  mov ax,data
        mov ds,ax
        mov cl,row1
        mov dl,row2
        cmp cl,dl
        jne last
        mov cl,col1
        mov dl,col2
        cmp cl,dl
        jne last
        mov al,row2
        mul cl
        mov cx,ax
        mov si, offset matrix1
        mov di, offset matrix2
        mov bx, offset result
here:   mov al, [si]
        sub al, [di]
        mov [bx], al
        inc si
        inc di
        inc bl
        loop here
last:   mov ah,4ch
        int 21h
code ends
end start
```

RESULT:

Thus matrix subtraction has been achieved.

Experiment No 6: Sorting

A. AIM:

Program for sorting in ascending order.

ALGORITHM:

- Initialize the data segment.
- Move data segment address to ds
- Initialize ah with 00h.
- Move row value to al and col value to bl.
- Multiply al with bl.
- Decrement ax.
- Here :
    - Move ax value to cx
    - Load offset of matrix1 to si
- Here1 :
    - Move contents pointed by si to bl
    - Compare contents pointed by si+1 with bl
    - If bl is less than or equal to [si+1] jump to next

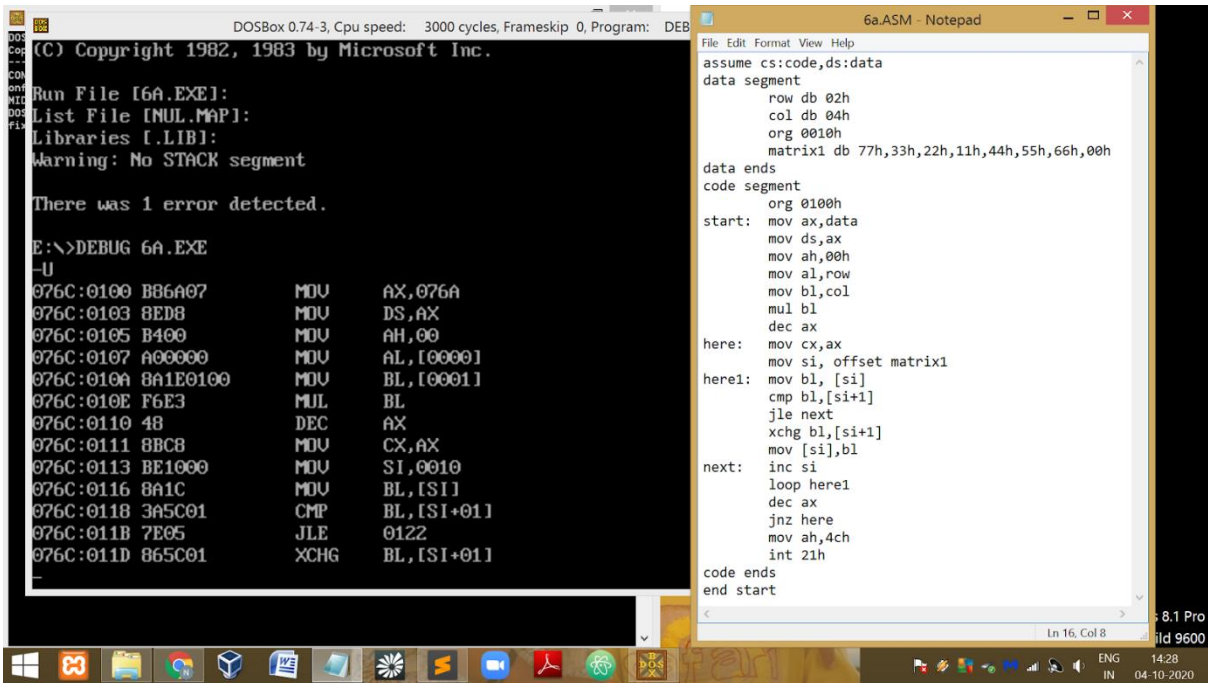
  - Exchange values of bl and [si+1]
  - Move bl to matrix1
- next :
  - Increment si
  - Loop here1
  - Decrement ax
  - Jump to here if not equal to zero
  - Terminate the program

PROGRAM:

PROGRAM	**COMMENTS**
assume cs:code,ds:data data segment row db 02h col db 04h org 0010h matrix1 db 77h,33h,22h,11h,44h,55h,66h,00h data ends code segment org 0100h start: mov ax,data mov ds,ax mov ah,00h mov al,row mov bl,col mul bl dec ax	 Load data segment to ds Initialise ah with 00h Move row value to ah Move col value to bl Multiply al with bl Decrement ax
Here: mov cx,ax mov si, offset matrix1	 Move contents of ax to cx. Move offset of matrix1 to si.
here1: mov bl, [si] cmp bl,[si+1] jle next xchg bl,[si+1] mov [si],bl	 Move contents pointed by si to bl Move contents pointed by si+1 to bl If bl is less than or equal to [si+1] jump to next Exchange values of bl and [si+1] Move bl to matrix1

next: inc si loop here1 dec ax jnz here mov ah,4ch int 21h	 Increment si Start loop here1 Decrement ax Jump to here if not equal to 0 Terminate the program

UNASSEMBLED CODE:

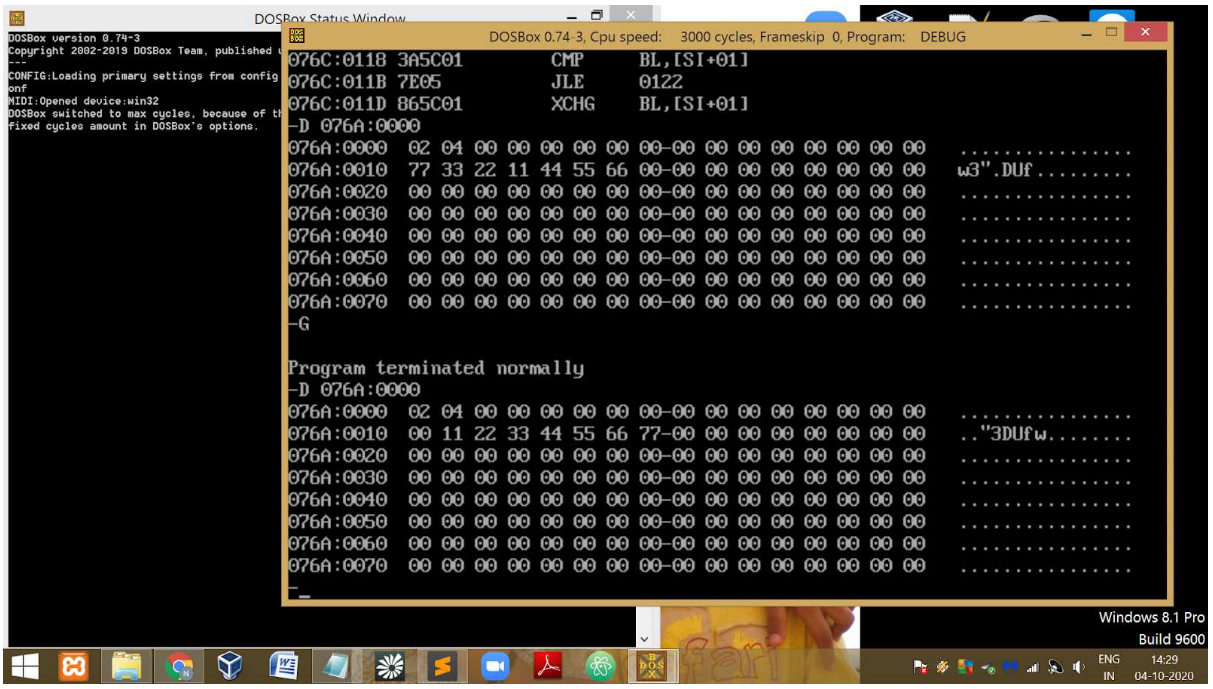

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEB

```
(C) Copyright 1982, 1983 by Microsoft Inc.

Run File [6A.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
Warning: No STACK segment

There was 1 error detected.

E:\>DEBUG 6A.EXE
-U
076C:0100 B86A07        MOV     AX,076A
076C:0103 8ED8          MOV     DS,AX
076C:0105 B400          MOV     AH,00
076C:0107 A00000        MOV     AL,[0000]
076C:010A 8A1E0100      MOV     BL,[0001]
076C:010E F6E3          MUL     BL
076C:0110 48            DEC     AX
076C:0111 8BC8          MOV     CX,AX
076C:0113 BE1000        MOV     SI,0010
076C:0116 8A1C          MOV     BL,[SI]
076C:0118 3A5C01        CMP     BL,[SI+01]
076C:011B 7E05          JLE     0122
076C:011D 865C01        XCHG    BL,[SI+01]
-
```

6a.ASM - Notepad

```
assume cs:code,ds:data
data segment
        row db 02h
        col db 04h
        org 0010h
        matrix1 db 77h,33h,22h,11h,44h,55h,66h,00h
data ends
code segment
        org 0100h
start:  mov ax,data
        mov ds,ax
        mov ah,00h
        mov al,row
        mov bl,col
        mul bl
        dec ax
here:   mov cx,ax
        mov si, offset matrix1
here1:  mov bl, [si]
        cmp bl,[si+1]
        jle next
        xchg bl,[si+1]
        mov [si],bl
next:   inc si
        loop here1
        dec ax
        jnz here
        mov ah,4ch
        int 21h
code ends
end start
```

SAMPLE INPUT/OUTPUT:



```
DOSBox version 0.74-3
Copyright 2002-2019 DOSBox Team, published u
---
CONFIG:Loading primary settings from config
onf
MIDI:Opened device:win32
DOSBox switched to max cycles, because of th
fixed cycles amount in DOSBox's options.
```

```
DOSBox 0.74-3, Cpu speed:    3000 cycles, Frameskip  0, Program:   DEBUG
076C:0118 3A5C01        CMP     BL,[SI+01]
076C:011B 7E05          JLE     0122
076C:011D 865C01        XCHG    BL,[SI+01]
-D 076A:0000
076A:0000  02 04 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0010  77 33 22 11 44 55 66 00-00 00 00 00 00 00 00 00   w3".DUf.........
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
-G

Program terminated normally
-D 076A:0000
076A:0000  02 04 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0010  00 11 22 33 44 55 66 77-00 00 00 00 00 00 00 00   .."3DUfw........
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
-_
```

RESULT:

Thus sorting in ascending order is achieved.

B. AIM:

Program for sorting in descending order.

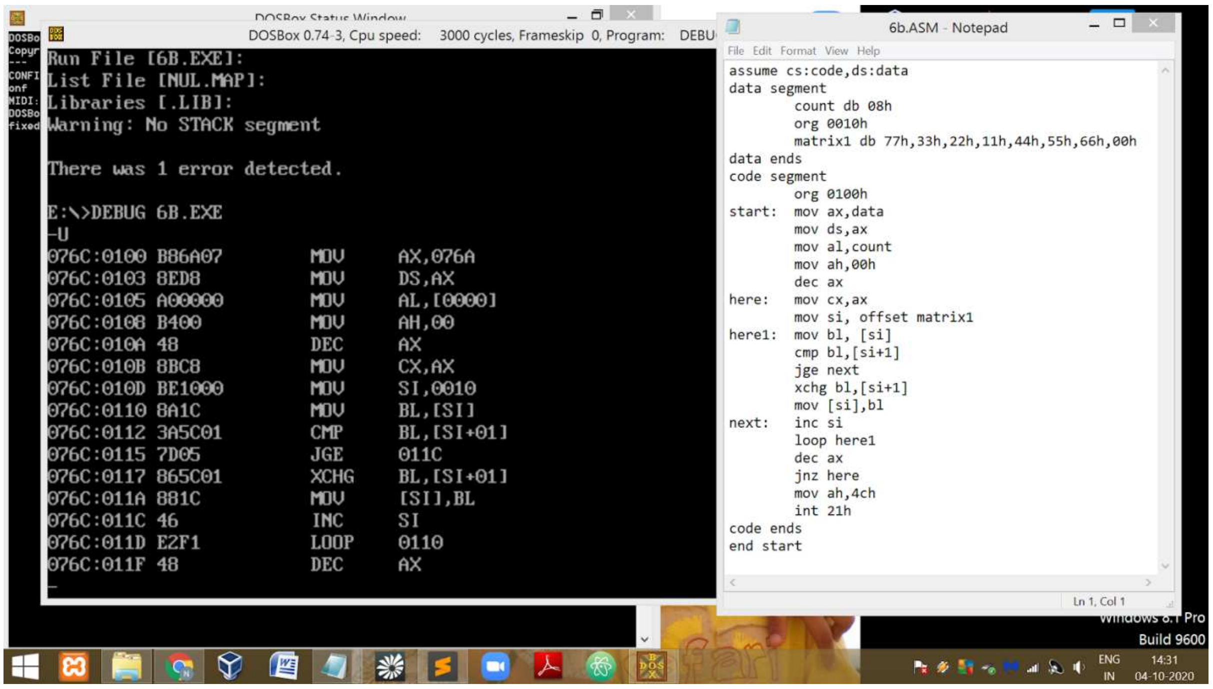
ALGORITHM:

- Initialize the data segment.
- Move data segment address to ds
- Load al with count value
- Initialize ah with 00h.
- Decrement ax.
- Here :
  - Move ax value to cx
  - Load offset of matrix1 to si
- Here1 :
  - Move contents pointed by si to bl
  - Compare contents pointed by si+1 with bl
  - If bl is greater than or equal to [si+1] jump to next
  - Exchange values of bl and [si+1]
  - Move bl to matrix1
- next :
  - Increment si
  - Loop here1
  - Decrement ax
  - Jump to here if not equal to zero
  - Terminate the program

PROGRAM:

PROGRAM	**COMMENTS**
assume cs:code,ds:data	
data segment	
count db 08h	
org 0010h	
matrix1 db 77h,33h,22h,11h,44h,55h,66h,00h	
data ends	
code segment	
org 0100h	
start:	
mov ax,data	Load data segment to ds
mov ds,ax	
mov al,count	Load al with count.
mov ah,00h	Initialise ah with 00h
dec ax	Decrement ax
Here:	
mov cx,ax	Move contents of ax to cx.
mov si, offset matrix1	Move offset of matrix1 to si.
here1:	
mov bl, [si]	Move contents pointed by si to bl
cmp bl,[si+1]	Move contents pointed by si+1 to bl
jge next	If bl is greater than or equal to [si+1]
xchg bl,[si+1]	jump to next
mov [si],bl	Exchange values of bl and [si+1]
	Move bl to matrix1
next:	
inc si	Increment si
loop here1	Start loop here1
dec ax	Decrement ax
jnz here	Jump to here if not equal to 0
mov ah,4ch	Terminate the program
int 21h	

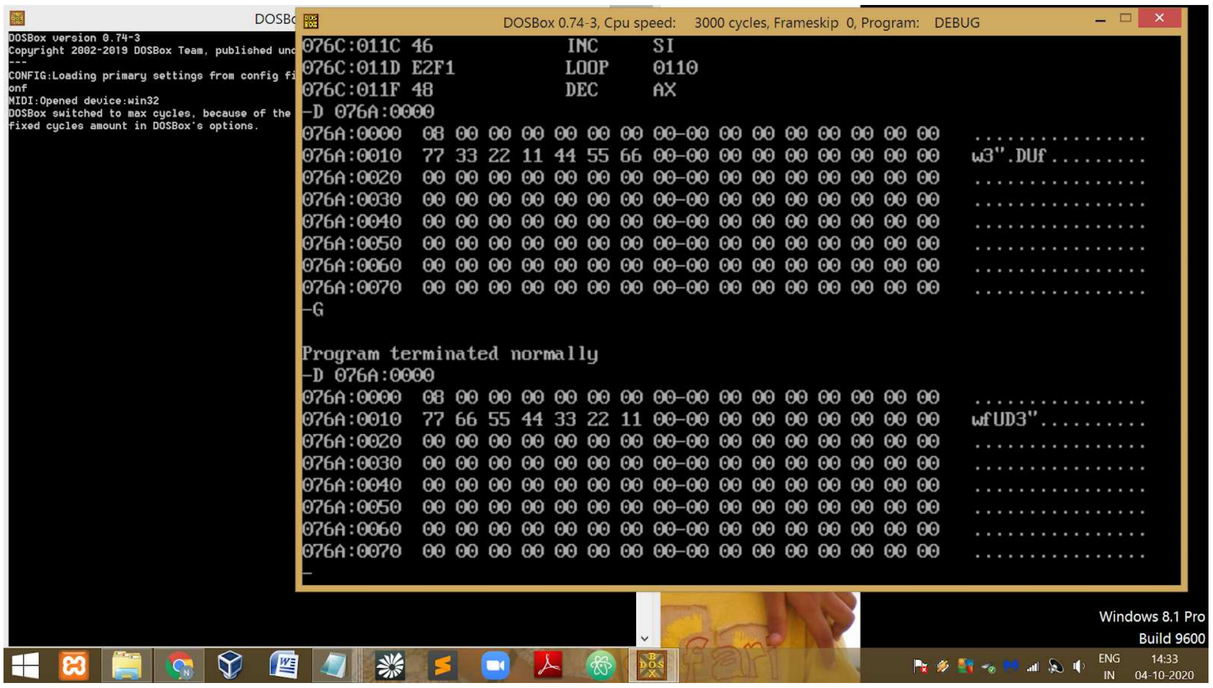
UNASSEMBLED CODE:



```
Run File [6B.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
Warning: No STACK segment

There was 1 error detected.

E:\>DEBUG 6B.EXE
-U
076C:0100 B86A07        MOV     AX,076A
076C:0103 8ED8          MOV     DS,AX
076C:0105 A00000        MOV     AL,[0000]
076C:0108 B400          MOV     AH,00
076C:010A 48            DEC     AX
076C:010B 8BC8          MOV     CX,AX
076C:010D BE1000        MOV     SI,0010
076C:0110 8A1C          MOV     BL,[SI]
076C:0112 3A5C01        CMP     BL,[SI+01]
076C:0115 7D05          JGE     011C
076C:0117 865C01        XCHG    BL,[SI+01]
076C:011A 881C          MOV     [SI],BL
076C:011C 46            INC     SI
076C:011D E2F1          LOOP    0110
076C:011F 48            DEC     AX
-
```

```
assume cs:code,ds:data
data segment
        count db 08h
        org 0010h
        matrix1 db 77h,33h,22h,11h,44h,55h,66h,00h
data ends
code segment
        org 0100h
start:  mov ax,data
        mov ds,ax
        mov al,count
        mov ah,00h
        dec ax
here:   mov cx,ax
        mov si, offset matrix1
here1:  mov bl, [si]
        cmp bl,[si+1]
        jge next
        xchg bl,[si+1]
        mov [si],bl
next:   inc si
        loop here1
        dec ax
        jnz here
        mov ah,4ch
        int 21h
code ends
end start
```

SAMPLE INPUT/OUTPUT



```
DOSBox version 0.74-3
Copyright 2002-2019 DOSBox Team, published unc
---
CONFIG:Loading primary settings from config fi
onf
MIDI:Opened device:win32
DOSBox switched to max cycles, because of the
fixed cycles amount in DOSBox's options.
```

```
076C:011C 46            INC     SI
076C:011D E2F1          LOOP    0110
076C:011F 48            DEC     AX
-D 076A:0000
076A:0000  08 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0010  77 33 22 11 44 55 66 00-00 00 00 00 00 00 00 00   w3".DUf.........
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
-G

Program terminated normally
-D 076A:0000
076A:0000  08 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0010  77 66 55 44 33 22 11 00-00 00 00 00 00 00 00 00   wfUD3"..........
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
-
```

RESULT:

Thus sorting in descending order is achieved.

Experiment No. 7: BCD Addition and Subtraction

A. AIM:

Program for performing addition of two 8-bit BCD numbers.

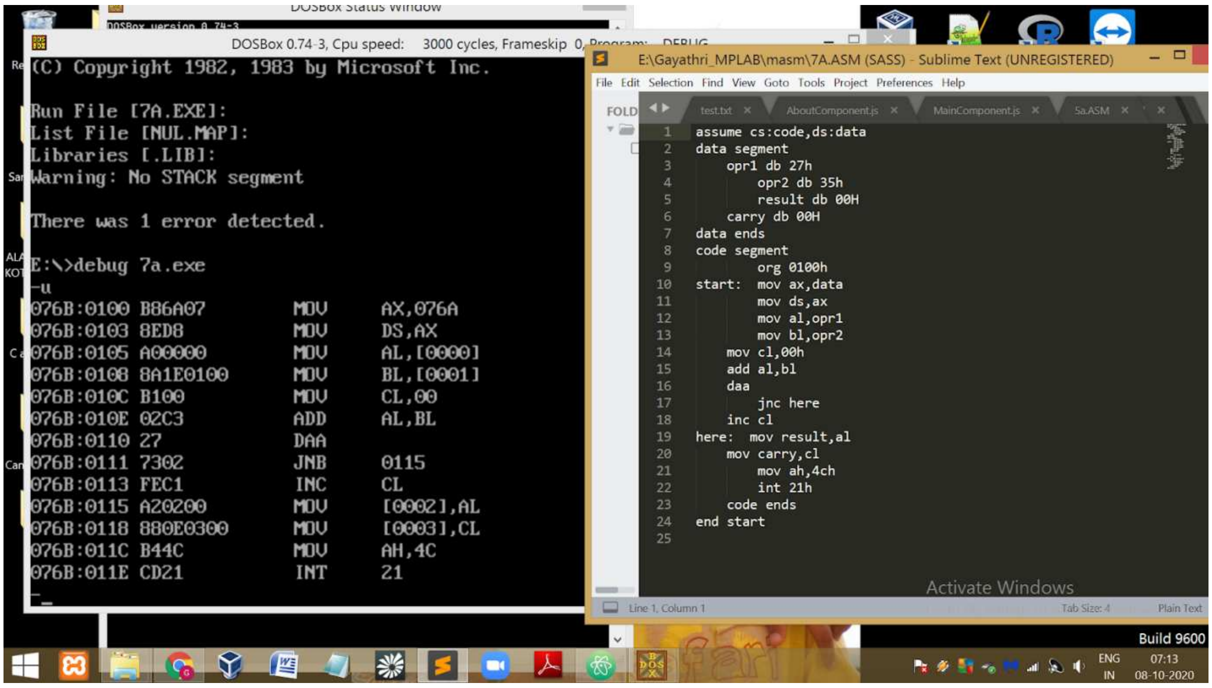
ALGORITHM:

- Initialize the data segment
- Move data segment address to ds
- Load opr1 to al and opr2 to bl
- Load 00h to cl register for carry
- Add al and bl
- Execute daa instruction to adjust the result of the addition of two packed BCD values to create a packed BCD result
- If there is no carry being generated, goto here segment else, increment cl by 1
- In here segment,
  - Load al to result
  - Load cl to carry
  - Terminate the program

PROGRAM:

PROGRAM	**COMMENTS**
assume cs:code,ds:data data segment opr1 db 27h opr2 db 35h result db 00H carry db 00H data ends code segment org 0100h start: mov ax,data mov ds,ax mov al,opr1 mov bl,opr2 mov cl,00h add al,bl daa jnc here inc cl	Transferring address of data segment to ds Value of opr1 is loaded to al Value of opr2 is loaded to bl Initializing the value of cl with 00h al=al+bl Add numbers represented in 8-bit packed BCD code Jump to “here” segment if no carry is generated Increments cl by 1
Here: mov result,al mov carry,cl mov ah,4ch int 21h	Load register value of al to result Load cl value to carry Terminate the program

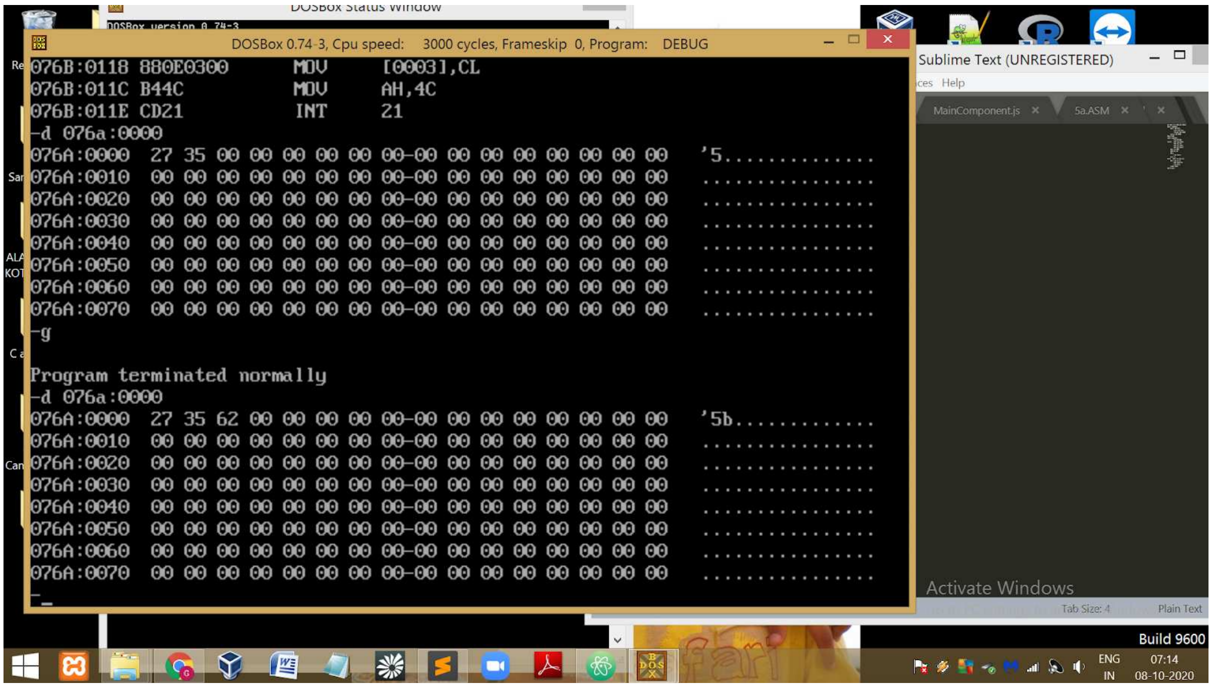
UNASSEMBLED CODE:



```
(C) Copyright 1982, 1983 by Microsoft Inc.

Run File [7A.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
Warning: No STACK segment

There was 1 error detected.

E:\>debug 7a.exe
-u
076B:0100 B86A07        MOV     AX,076A
076B:0103 8ED8          MOV     DS,AX
076B:0105 A00000        MOV     AL,[0000]
076B:0108 8A1E0100      MOV     BL,[0001]
076B:010C B100          MOV     CL,00
076B:010E 02C3          ADD     AL,BL
076B:0110 27            DAA
076B:0111 7302          JNB     0115
076B:0113 FEC1          INC     CL
076B:0115 A20200        MOV     [0002],AL
076B:0118 880E0300      MOV     [0003],CL
076B:011C B44C          MOV     AH,4C
076B:011E CD21          INT     21
-_
```

```
assume cs:code,ds:data
data segment
    opr1 db 27h
        opr2 db 35h
        result db 00H
    carry db 00H
data ends
code segment
        org 0100h
start:  mov ax,data
        mov ds,ax
        mov al,opr1
        mov bl,opr2
    mov cl,00h
    add al,bl
    daa
        jnc here
    inc cl
here:  mov result,al
    mov carry,cl
        mov ah,4ch
        int 21h
    code ends
end start
```

SAMPLE INPUT/OUTPUT:



```
076B:0118 880E0300      MOV     [0003],CL
076B:011C B44C          MOV     AH,4C
076B:011E CD21          INT     21
-d 076a:0000
076A:0000  27 35 00 00 00 00 00 00-00 00 00 00 00 00 00 00   '5..............
076A:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
-g

Program terminated normally
-d 076a:0000
076A:0000  27 35 62 00 00 00 00 00-00 00 00 00 00 00 00 00   '5b.............
076A:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
-_
```

RESULT:

Thus addition of two BCD numbers has been performed.

B. AIM:

Program for performing subtraction of two 8-bit BCD numbers.

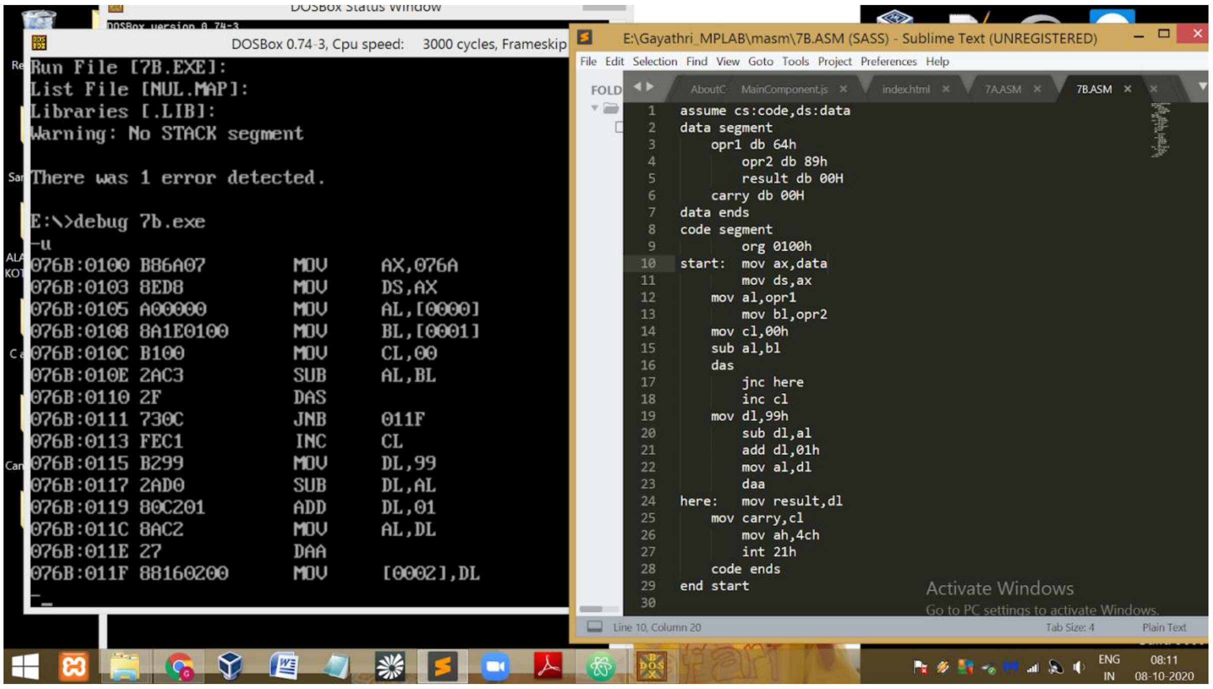
ALGORITHM:

- Initialize the data segment
- Move data segment address to ds
- Load opr1 to al and opr2 to bl
- Load 00h to cl register
- Subtract al and bl
- Execute das instruction to adjust the result of the subtraction of two packed BCD values to create a packed BCD result
- If al is greater than bl, goto here segment else, increment cl by 1 and find the 10’s complement of result and decimal adjust it.
- In here segment,
    - Load dl to result
    - Load cl to carry
    - Terminate the program

PROGRAM:

PROGRAM	**COMMENTS**
assume cs:code,ds:data	
data segment	
opr1 db 64h	
opr2 db 89h	
result db 00H	
carry db 00H	
data ends	
code segment	
org 0100h	
start:	
mov ax,data	Load data segment to ds
mov ds,ax	
mov al,opr1	Value of opr1 is loaded to al
mov bl,opr2	Value of opr2 is loaded to bl
mov cl,00h	Initializing the value of cl with 00h
sub al,bl	al=al-bl
das	Subtract numbers represented in 8-bit packed BCD code
jnc here	Jump to “here” segment if al>bl
inc cl	Increment value of cl
mov dl,99h	Load dl with 99h
sub dl,al	dl=dl-al
add dl,01h	dl=dl+01h
mov al,dl	Load al with value of dl
daa	Add numbers represented in 8-bit packed BCD code
here:	
mov result,dl	Load register value of dl to result
mov carry,cl	Load cl value to carry
mov ah,4ch	Terminate the program
int 21h	

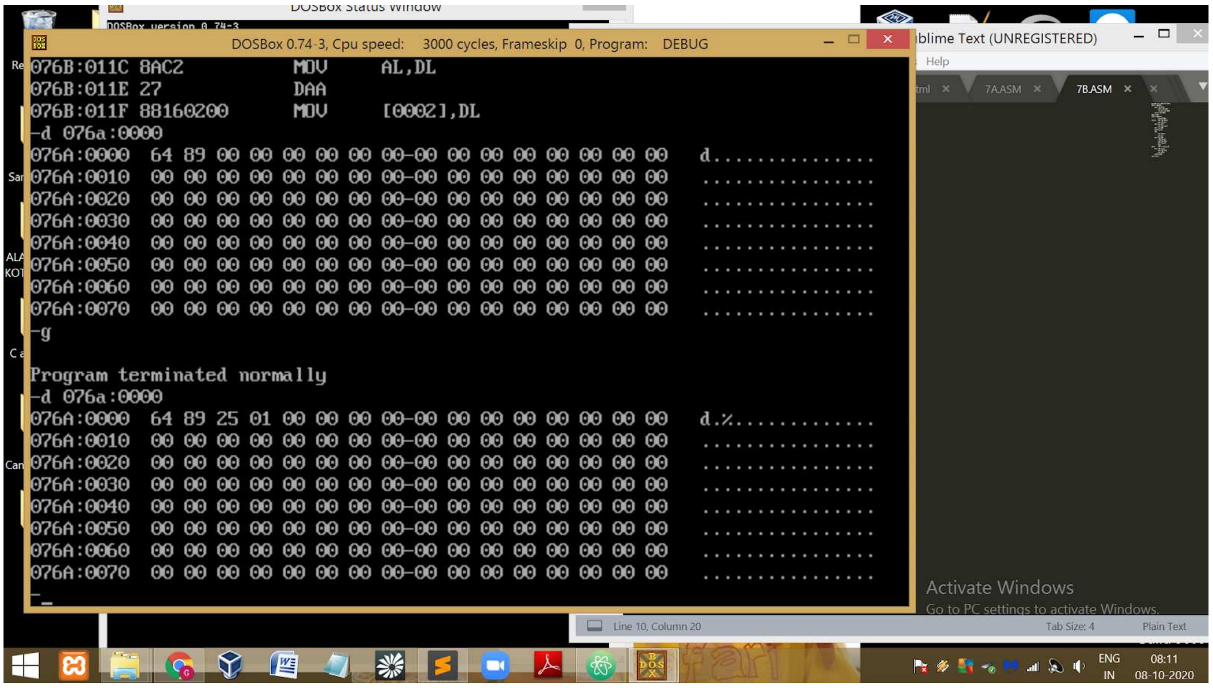
UNASSEMBLED CODE:



```
Run File [7B.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
Warning: No STACK segment

There was 1 error detected.

E:\>debug 7b.exe
-u
076B:0100 B86A07        MOV     AX,076A
076B:0103 8ED8          MOV     DS,AX
076B:0105 A00000        MOV     AL,[0000]
076B:0108 8A1E0100      MOV     BL,[0001]
076B:010C B100          MOV     CL,00
076B:010E 2AC3          SUB     AL,BL
076B:0110 2F            DAS
076B:0111 730C          JNB     011F
076B:0113 FEC1          INC     CL
076B:0115 B299          MOV     DL,99
076B:0117 2AD0          SUB     DL,AL
076B:0119 80C201        ADD     DL,01
076B:011C 8AC2          MOV     AL,DL
076B:011E 27            DAA
076B:011F 88160200      MOV     [0002],DL
-
```

```
assume cs:code,ds:data
data segment
    opr1 db 64h
        opr2 db 89h
        result db 00H
    carry db 00H
data ends
code segment
        org 0100h
start:  mov ax,data
        mov ds,ax
    mov al,opr1
        mov bl,opr2
    mov cl,00h
    sub al,bl
    das
        jnc here
        inc cl
    mov dl,99h
        sub dl,al
        add dl,01h
        mov al,dl
        daa
here:   mov result,dl
    mov carry,cl
        mov ah,4ch
        int 21h
    code ends
end start
```

SAMPLE INPUT/OUTPUT



```
076B:011C 8AC2          MOV     AL,DL
076B:011E 27            DAA
076B:011F 88160200      MOV     [0002],DL
-d 076a:0000
076A:0000  64 89 00 00 00 00 00 00-00 00 00 00 00 00 00 00   d...............
076A:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
-g

Program terminated normally
-d 076a:0000
076A:0000  64 89 25 01 00 00 00 00-00 00 00 00 00 00 00 00   d.%.............
076A:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
-
```

RESULT:

Thus subtraction of two BCD numbers has been performed.