

Experiment No 3: String Manipulations

Date: 07-09-2020

NAME: Harshini S

REG.NO: 185001058

A. AIM:

Program for moving a string of bytes.

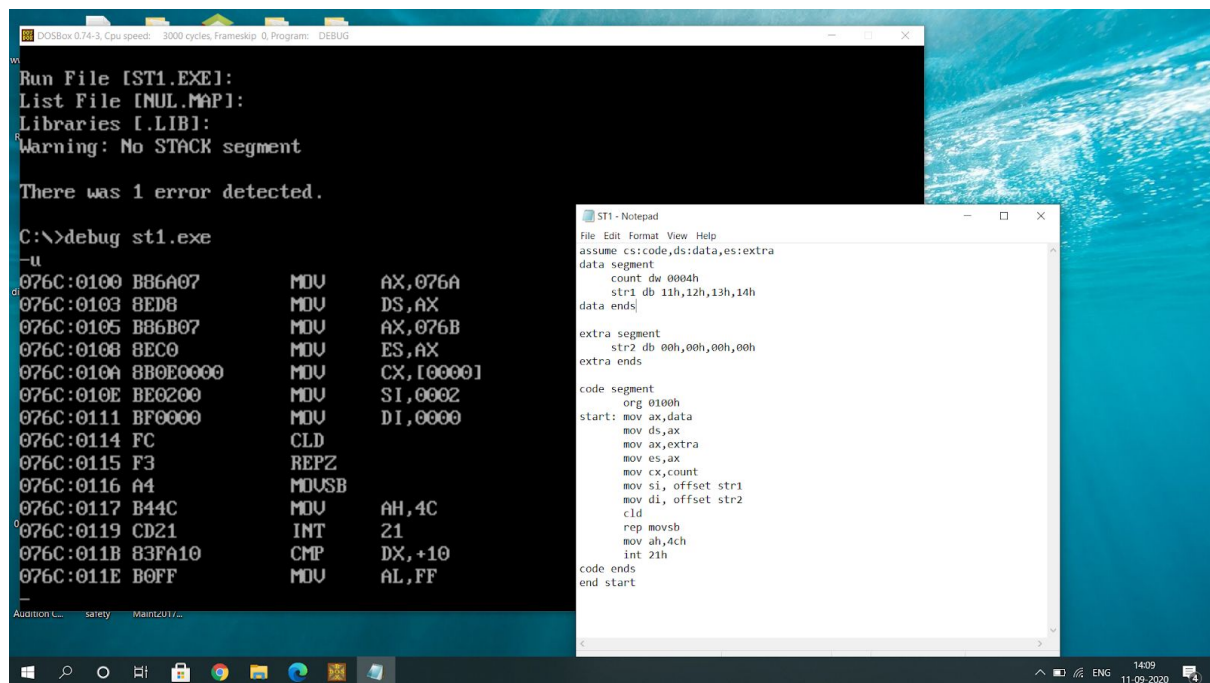
ALGORITHM:

- Initialize the data segment and the extra segment
- Move data segment address to ds
- Move extra segment to es
- Load count to cx
- Load offset of str1 to si and offset of str2 to di
- Clear directory flag
- Move the string of bytes
- Terminate the program

PROGRAM:

PROGRAM	COMMENTS
mov ax,data mov ds,ax	Load data segment to ds
mov ax,extra mov es,ax	Load extra segment to es
mov cx,count	Load count value to cx
mov si, offset str1	Load offset of str1 to si
mov di, offset str2	Load offset of str2 to di
cld	Clear directory flag
rep movsb	Copy string of bytes to extra segment
mov ah,4ch int 21h	Terminate the program

UNASSEMBLED CODE:



The screenshot shows a DOSBox window with the command prompt displaying the unassembly of ST1.EXE. The assembly code is as follows:

```
076C:0100 B86A07      MOV     AX,076A
076C:0103 8ED8      MOV     DS,AX
076C:0105 B86B07      MOV     AX,076B
076C:0108 8EC0      MOV     ES,AX
076C:010A 8B0E0000    MOV     CX,[0000]
076C:010E BE0200    MOV     SI,0002
076C:0111 BF0000    MOV     DI,0000
076C:0114 FC      CLD
076C:0115 F3      REPZ
076C:0116 A4      MOUSB
076C:0117 B44C      MOV     AH,4C
076C:0119 CD21      INT     21
076C:011B 83FA10    CMP     DX,+10
076C:011E B0FF      MOV     AL,FF
```

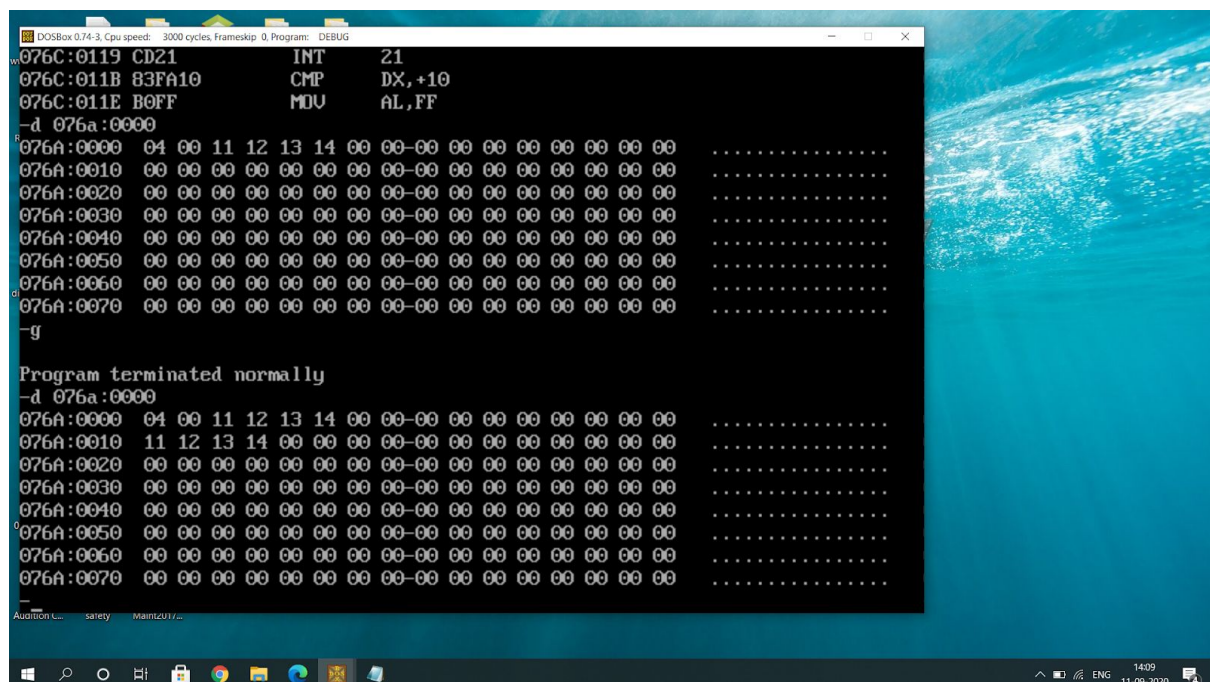
Simultaneously, a Notepad window displays the source code of ST1.EXE:

```
assume cs:code,ds:data,es:extra
data segment
    count dw 0004h
    str1 db 11h,12h,13h,14h
data ends

extra segment
    str2 db 00h,00h,00h,00h
extra ends

code segment
    org 0100h
start: mov ax,data
        mov ds,ax
        mov ax,extra
        mov es,ax
        mov cx,count
        mov si, offset str1
        mov di, offset str2
        cld
        rep movsb
        mov ah,4ch
        int 21h
code ends
end start
```

SAMPLE INPUT/OUTPUT:



The screenshot shows the memory dump and program termination in DOSBox. The memory dump shows the following data:

```
076A:0000 04 00 11 12 13 14 00 00-00 00 00 00 00 00 00 00 .....
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 .....
```

The program terminated normally, and the memory dump shows the following data:

```
076A:0000 04 00 11 12 13 14 00 00-00 00 00 00 00 00 00 00 .....
076A:0010 11 12 13 14 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 a string of bytes has been moved.

B. AIM:

Program for comparing 2 strings of bytes.

ALGORITHM:

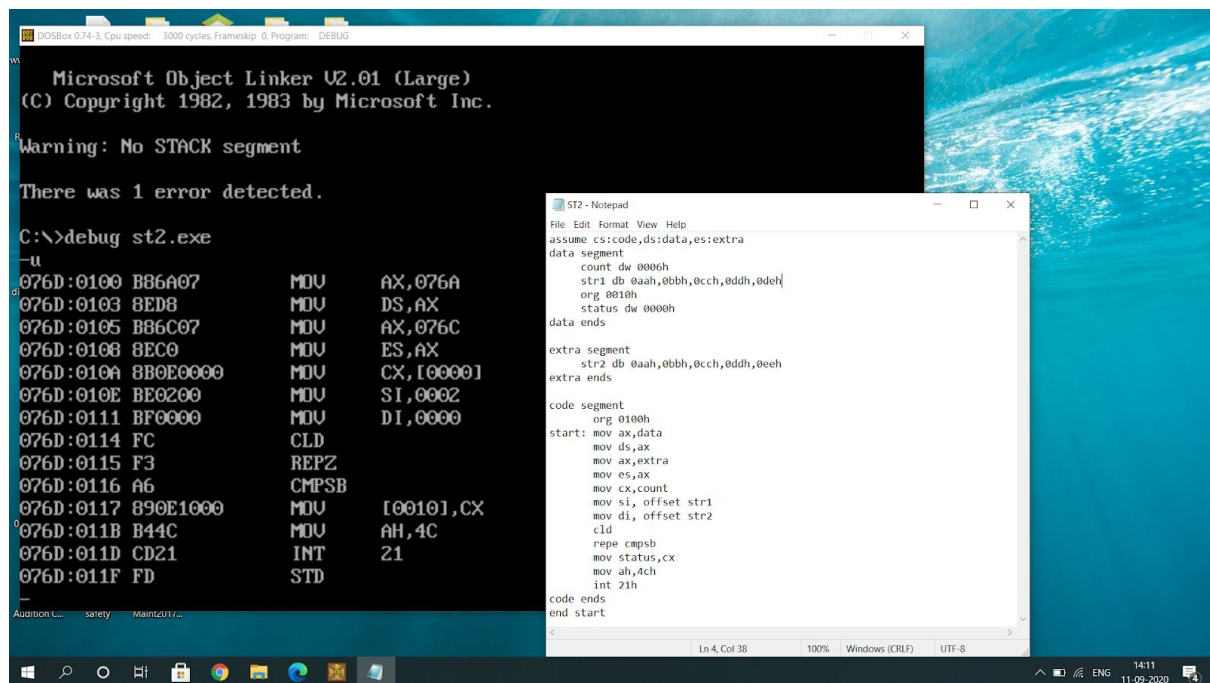
- Initialize the data segment and the extra segment

- Move data segment address to ds and extra segment to es
- Load count to cx
- Load offset of str1 to si and offset of str2 to di
- Clear directory flag
- Compare the 2 sets of strings
- Load cx to status
- Terminate the program

PROGRAM:

PROGRAM	COMMENTS
mov ax,data mov ds,ax	Load data segment to ds
mov ax,extra mov es,ax	Load extra segment to es
mov cx,count	Load count value to cx
mov si, offset str1	Load offset of str1 to si
mov di, offset str2	Load offset of str2 to di
cld	Clear directory flag
repe cmpsb	Compare string of bytes
mov status, cx	Load cx to status
mov ah,4ch int 21h	Terminate the program

UNASSEMBLED CODE:



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

Warning: No STACK segment

There was 1 error detected.

C:\>debug st2.exe
-u
076D:0100 B86A07      MOV     AX,076A
076D:0103 8ED8        MOV     DS,AX
076D:0105 B86C07      MOV     AX,076C
076D:0108 8EC0        MOV     ES,AX
076D:010A 8B0E0000     MOV     CX,[0000]
076D:010E BE0200     MOV     SI,0002
076D:0111 BF0000     MOV     DI,0000
076D:0114 FC         CLD
076D:0115 F3         REPZ
076D:0116 A6         CMPSB
076D:0117 890E1000     MOV     [0010],CX
076D:011B B44C        MOV     AH,4C
076D:011D CD21        INT     21
076D:011F FD         STD

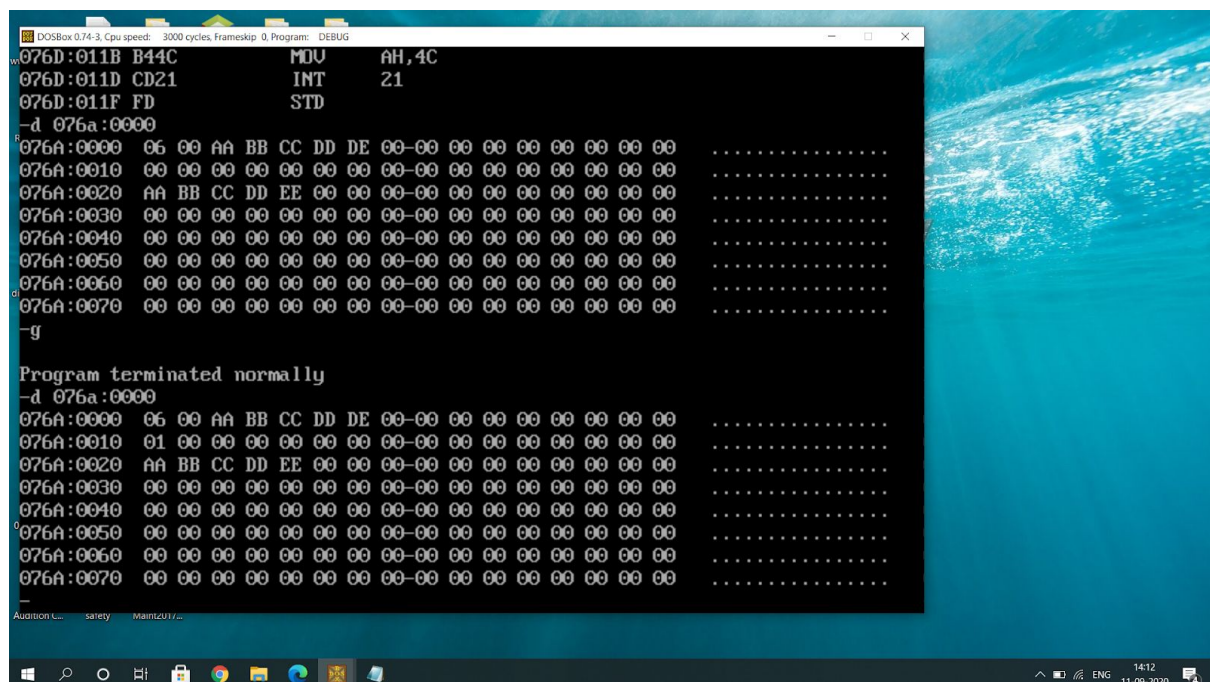
--
Assault L...  safety  MaintzU17...
```

```
ST2 - Notepad
File Edit Format View Help
assume cs:code,ds:data,es:extra
data segment
    count dw 0006h
    str1 db 'aaah,bbbh,ccch,dddh,eeeh'
    org 0010h
    status dw 0000h
data ends

extra segment
    str2 db 'aaah,bbbh,ccch,dddh,eeeh'
extra ends

code segment
    org 0100h
start: mov ax,data
        mov ds,ax
        mov ax,extra
        mov es,ax
        mov cx,count
        mov si,offset str1
        mov di,offset str2
        cld
        repe cmpsb
        mov status,cx
        mov ah,4ch
        int 21h
code ends
end start
```

SAMPLE INPUT/OUTPUT



```
076D:011B B44C        MOV     AH,4C
076D:011D CD21        INT     21
076D:011F FD         STD

--d 076a:0000
076A:0000 06 00 AA BB CC DD DE 00-00 00 00 00 00 00 00 00 .....
076A:0010 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
076A:0020 AA BB CC DD EE 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 .....
076A:0040 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 .....
076A:0060 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 .....

--g

Program terminated normally
--d 076a:0000
076A:0000 06 00 AA BB CC DD DE 00-00 00 00 00 00 00 00 00 .....
076A:0010 01 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
076A:0020 AA BB CC DD EE 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 .....
076A:0040 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 .....
076A:0060 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 .....
```

RESULT:

The 2 string of bytes have been compared.

3. AIM:

Program for searching a byte in a string.

ALGORITHM:

- Initialize the data segment and extra segment.
- Move data segment address to ds and extra segment to es.
- Load count to cx
- Load str2 to al
- Load offset of str1 to di
- Clear directory flag
- Compare the 2 set of strings
- Load cx to count
- Halt until the next external interrupt is fired.
- Terminate the program

PROGRAM:

PROGRAM	COMMENTS
mov ax,data mov ds,ax	Load data segment to ds
mov ax,extra mov es,ax	Load extra segment to es
mov cx,count	Load count value to cx
mov al,str2	Load str2 to al
mov di, offset str1	Load offset of str1 to di
cld	Clear directory flag
repne scasb str1	Scan for str2 in str1
mov count, cx	Load cx to count
mov ah,4ch int 21h	Terminate the program

UNASSEMBLED CODE:

The screenshot shows a DOSBox window with the Microsoft Object Linker V2.01 (Large) interface. It displays a warning: "Warning: No STACK segment" and "There was 1 error detected." Below this, the command "C:\>debug st3.exe" is entered, followed by a list of assembly instructions for st3.exe. To the right, a Notepad window titled "ST3 - Notepad" shows the source code for st3.exe, which includes data and code segments with various instructions like mov, count, str1, str2, cld, repne, and hlt.

```

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

Warning: No STACK segment

There was 1 error detected.

C:\>debug st3.exe
-u
076D:0100 B86A07      MOV     AX,076A
076D:0103 8ED8        MOV     DS,AX
076D:0105 B86C07      MOV     AX,076C
076D:0108 8EC0        MOV     ES,AX
076D:010A 8B0E1000     MOV     CX,[0010]
076D:010E A01200     MOV     AL,[0012]
076D:0111 BF0000     MOV     DI,0000
076D:0114 FC          CLD
076D:0115 F2          REPNZ
076D:0116 AE          SCASB
076D:0117 890E1000     MOV     [0010],CX
076D:011B F4          HLT
076D:011C B44C        MOV     AH,4C
076D:011E CD21        INT     21

Autosave C:\safety MaintzU17...

ST3 - Notepad
File Edit Format View Help
assume cs:code,ds:data,es:extra
data segment
    org 0010h
    count dw 0006h
    str2 db 12h
data ends

extra segment
    str1 db 11h,12h,13h,14h
extra ends

code segment
    org 0100h
start: mov ax,data
        mov ds,ax
        mov ax,extra
        mov es,ax
        mov cx,count
        mov al,str2
        mov di,offset str1
        cld
        repne scasb str1
        mov count,cx
        hlt
        mov ah,4ch
        int 21h
code ends
end start

```

SAMPLE INPUT/OUTPUT

The screenshot shows the execution of st3.exe in DOSBox. The assembly instructions from the previous image are visible at the top. Below them, a memory dump is shown for address 076A:0000, displaying a series of hexadecimal values and their corresponding ASCII characters. The program terminates normally, as indicated by the message "Program terminated normally".

```

076D:011B F4          HLT
076D:011C B44C        MOV     AH,4C
076D:011E CD21        INT     21

-d 076a:0000
076A:0000 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0010 06 00 12 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0020 11 12 13 14 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 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
076A:0010 04 00 12 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
076A:0020 11 12 13 14 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:

Program for searching a byte in a string is thus shown

4. AIM:

Program for moving a string without using string instructions

ALGORITHM:

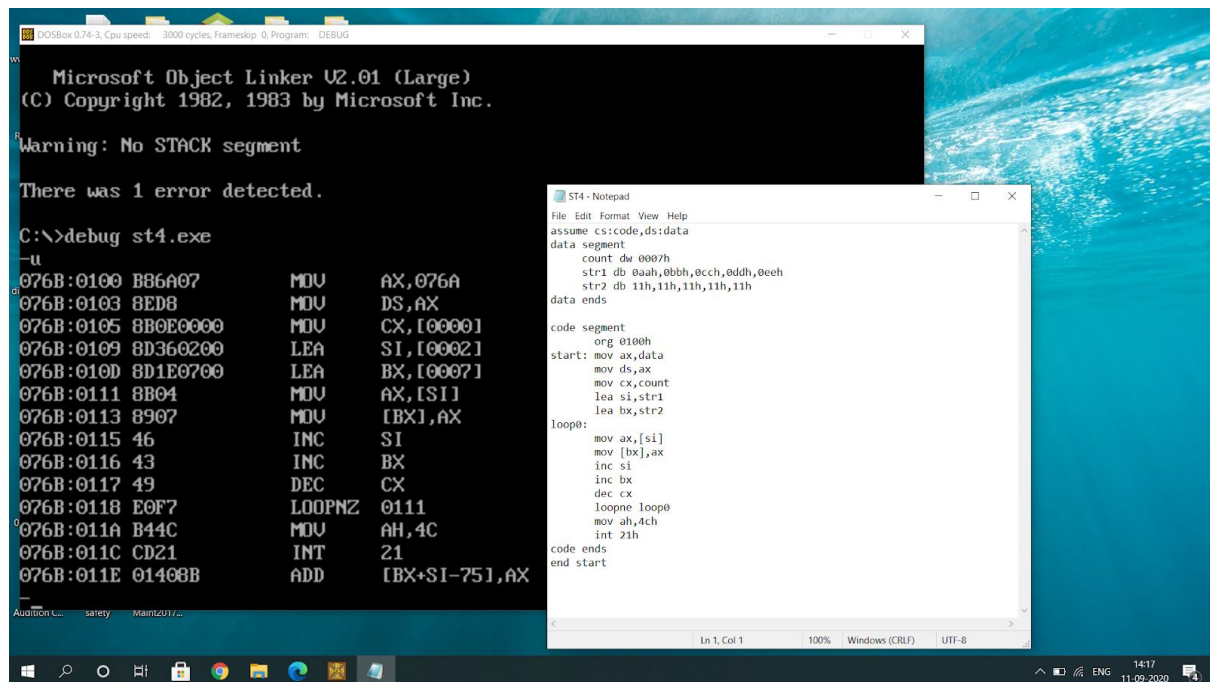
- Initialize the data segment

- Move data segment address to ds
- Load count to cx
- Load effective address of str1 to si and str2 to bx
- In loop0
 - Move data of si to ax and ax to location of bx
 - Increment si,bx and decrement cx
 - end if count = 0 and ZF=0
- Terminate the program

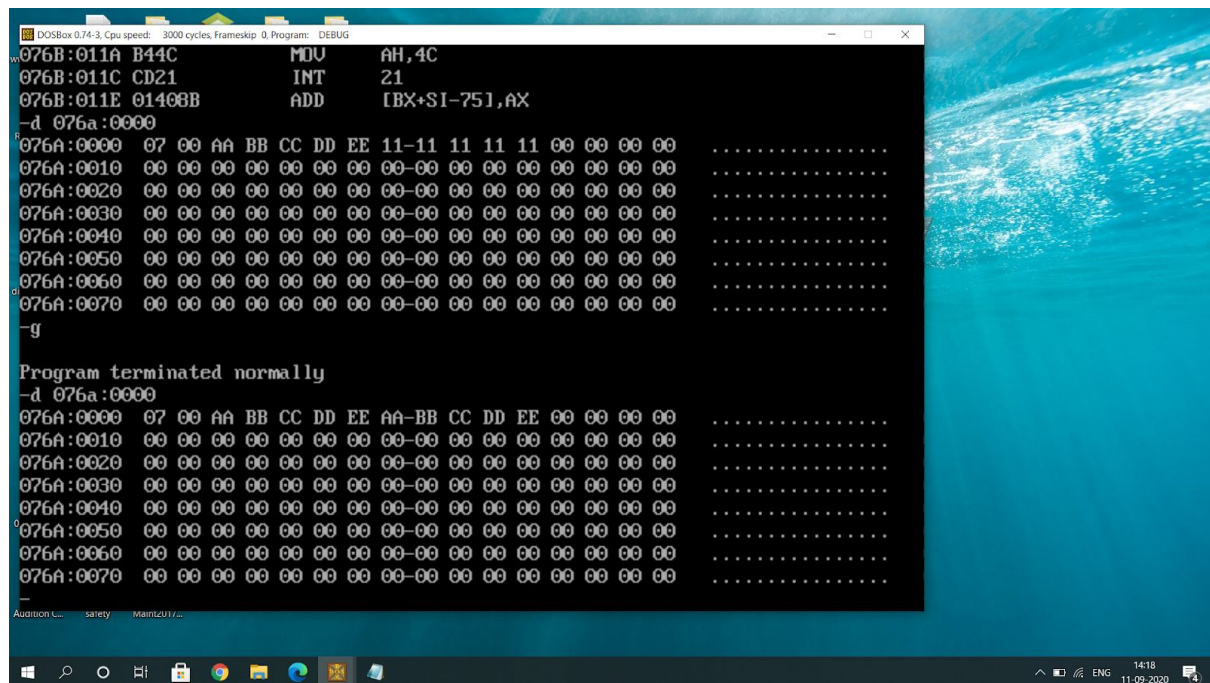
PROGRAM:

PROGRAM	COMMENTS
mov ax,data mov ds,ax	Load data segment to ds
mov cx,count	Load count value to cx
lea si,str1	Load effective address of str1 to si
lea bx,str2	Load effective address of str2 to bx
loop0:	
mov ax,[si]	Load data from [ds:si] to ax
mov [bx],ax	Load data from ax to bx
inc si	Increment si
inc bx	Increment bx
dec cx	Decrement cx
loopne loop0	End loop if count = 0 and ZF=0
mov ah,4ch int 21h	Terminate the program

UNASSEMBLED CODE:



SAMPLE INPUT/OUTPUT



RESULT:

Thus moving a string of bytes without using string instructions is thus shown.