

Experiment No 3: String Manipulations

Date: 09-09-2020

NAME: Gayathri M

REG.NO: 185001050

1. AIM:

Program for moving a string of bytes.

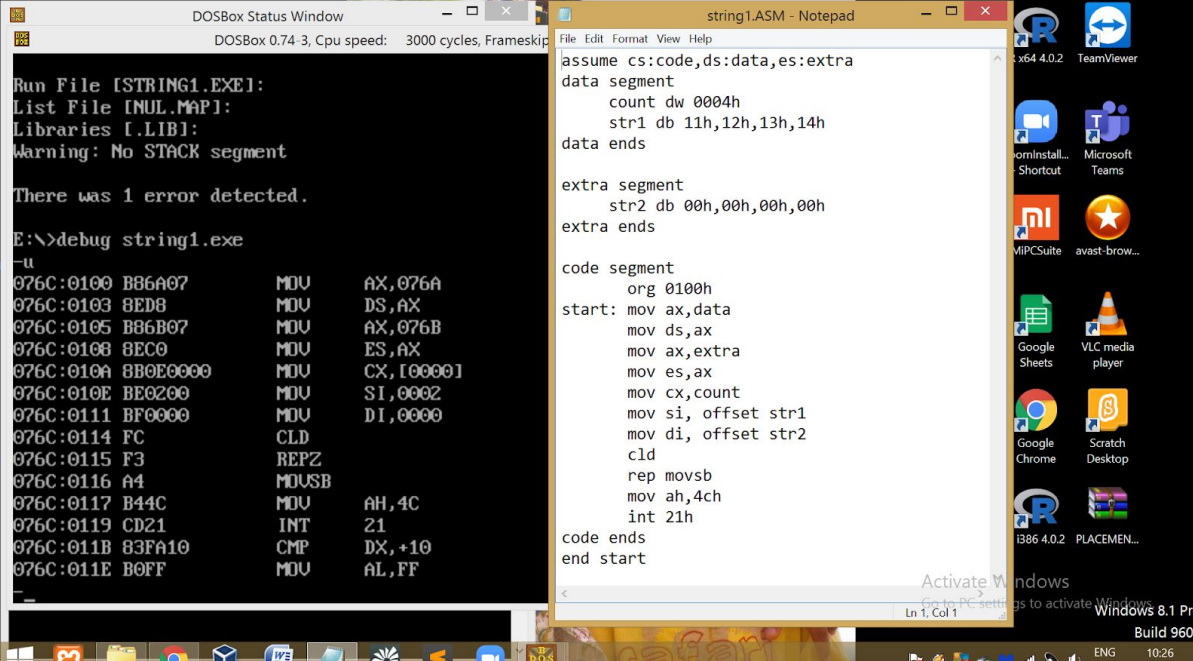
ALGORITHM:

- 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 on the left and a Notepad window on the right. The DOSBox window displays the execution of string1.EXE, showing a warning about no STACK segment and a list of assembly instructions. The Notepad window shows the source assembly code for string1.ASM.

```
Run File [STRING1.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
Warning: No STACK segment

There was 1 error detected.

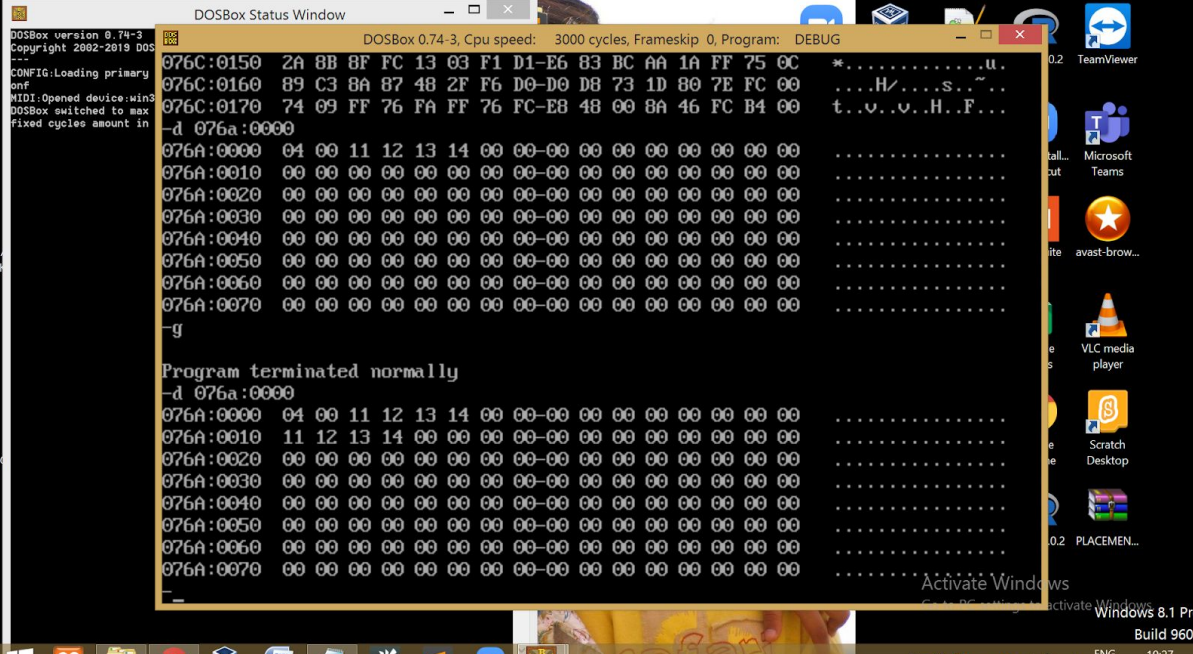
E:\>debug string1.exe
-u
076C:0100 B86A07      MOV     AX,076A
076C:0103 8ED8        MOV     DS,AX
076C:0105 B86B07      MOV     AX,076B
076C:0108 BEC0        MOV     ES,AX
076C:010A 8B0E0000      MOV     CX,[0000]
076C:010E BE0200      MOV     SI,0002
076C:0111 BF0000      MOV     DI,0000
076C:0114 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

string1.ASM - Notepad
File Edit Format View Help
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 DOSBox in DEBUG mode. The memory dump shows the string "H.S.F" at address 076A:0000. The ASCII output shows the string "H.S.F" followed by a newline.

```
DOSBox version 0.74-3
Copyright 2002-2019 DOS
CONFIG:Loading primary
onf
MIDI:Opened device:win3
DOSBox switched to max
fixed cycles amount in

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG
076C:0150 2A 8B 8F FC 13 03 F1 D1-E6 83 BC AA 1A FF 75 0C *.....u.
076C:0160 89 C3 8A 87 48 2F F6 D0-D0 D8 73 1D 80 7E FC 00 ....H/...s.~
076C:0170 74 09 FF 76 FA FF 76 FC-E8 48 00 8A 46 FC B4 00 t..v..H..F...
-d 076a:0000
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 .....
-g
Program terminated normally
-d 076a:0000
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.

2. AIM:

Program for comparing 2 strings of bytes.

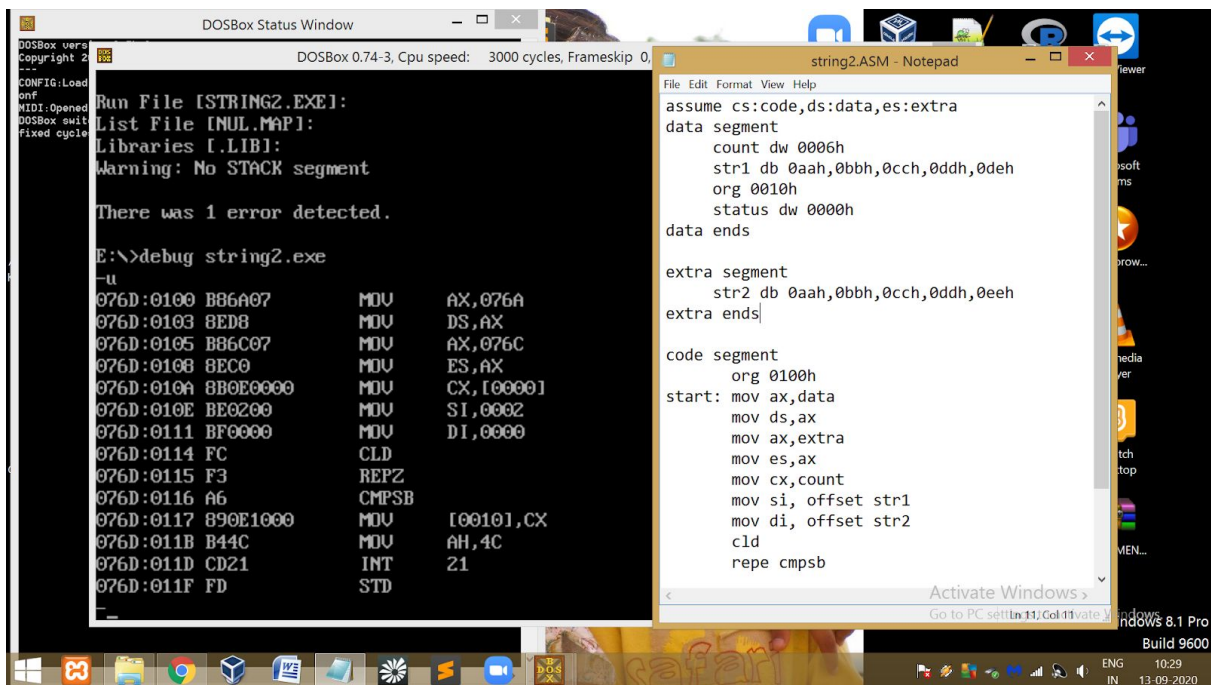
ALGORITHM:

- 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:



The screenshot shows a DOSBox window running a program and a Notepad window displaying the assembly code. The DOSBox window shows the execution of string2.exe, listing files, libraries, and a warning about the stack segment. It then displays the assembly code for string2.exe, showing instructions like MOV, DS, MOV, ES, MOV, CX, MOV, SI, MOV, DI, CLD, REPZ, CMPSB, MOV, AH, INT, and STD. The Notepad window shows the assembly code for string2.asm, which defines two strings, str1 and str2, and compares them using the CMPSB instruction.

```
Run File [STRING2.EXE]:
List File [NUL.MAP]:
Libraries [LIB1]:
Warning: No STACK segment

There was 1 error detected.

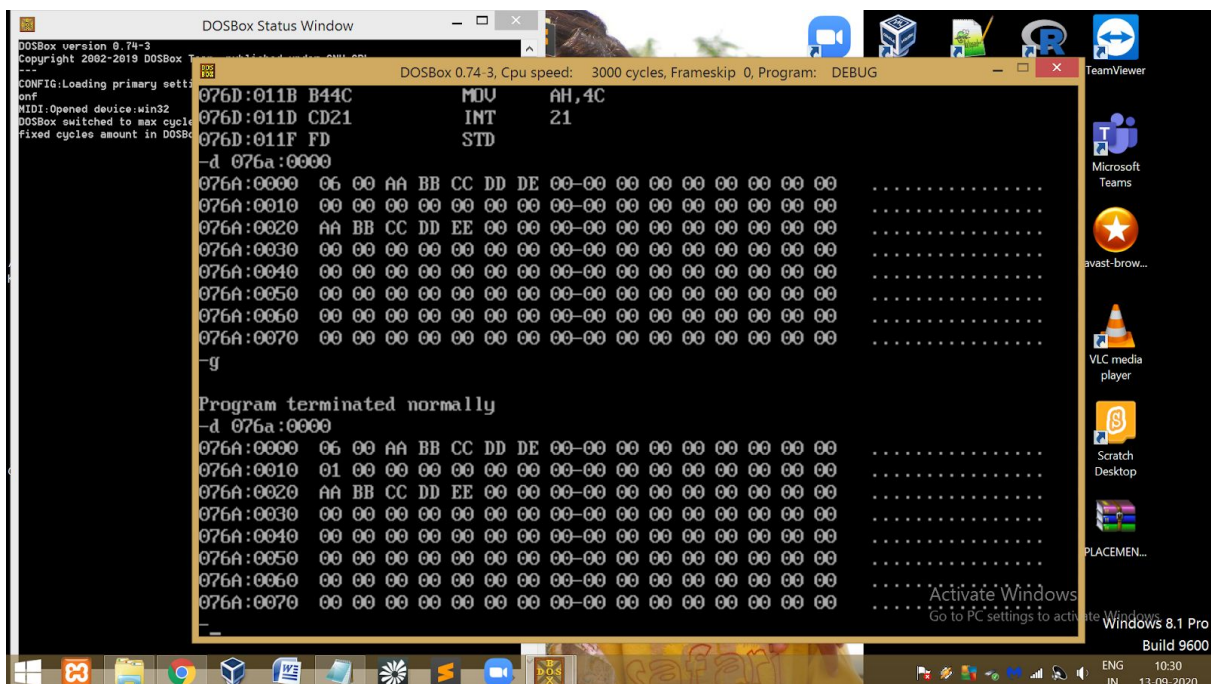
E:\>debug string2.exe
-u
076D:0100 B86A07      MOV     AX,076A
076D:0103 B8D8      MOV     DS,AX
076D:0105 B86C07      MOV     AX,076C
076D:0108 B8E0      MOV     ES,AX
076D:010A B80E0000      MOV     CX,[0000]
076D:010E B80200      MOV     SI,0002
076D:0111 B80000      MOV     DI,0000
076D:0114 FC        CLD
076D:0115 F3        REPZ
076D:0116 A6        CMPSB
076D:0117 B80E1000      MOV     [0010],CX
076D:011B B44C      MOV     AH,4C
076D:011D CD21      INT     21
076D:011F FD        STD
```

```
File Edit Format View Help
assume cs:code,ds:data,es:extra
data segment
    count dw 0006h
    str1 db 0aah,0bbh,0cch,0ddh,0deh
    org 0010h
    status dw 0000h
data ends

extra segment
    str2 db 0aah,0bbh,0cch,0ddh,0eeh
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
```

SAMPLE INPUT/OUTPUT



The screenshot shows the DOSBox window displaying the memory dump and the program termination message. The memory dump shows the contents of memory addresses 076A:0000 to 076A:0070, which correspond to the strings str1 and str2. The program terminated normally.

```
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
076A:0020 AA BB CC DD EE 00 00 00 00 00 00 00 00 00
076A:0030 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
076A:0050 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
076A:0070 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
076A:0010 01 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
076A:0030 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
076A:0050 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
076A:0070 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.

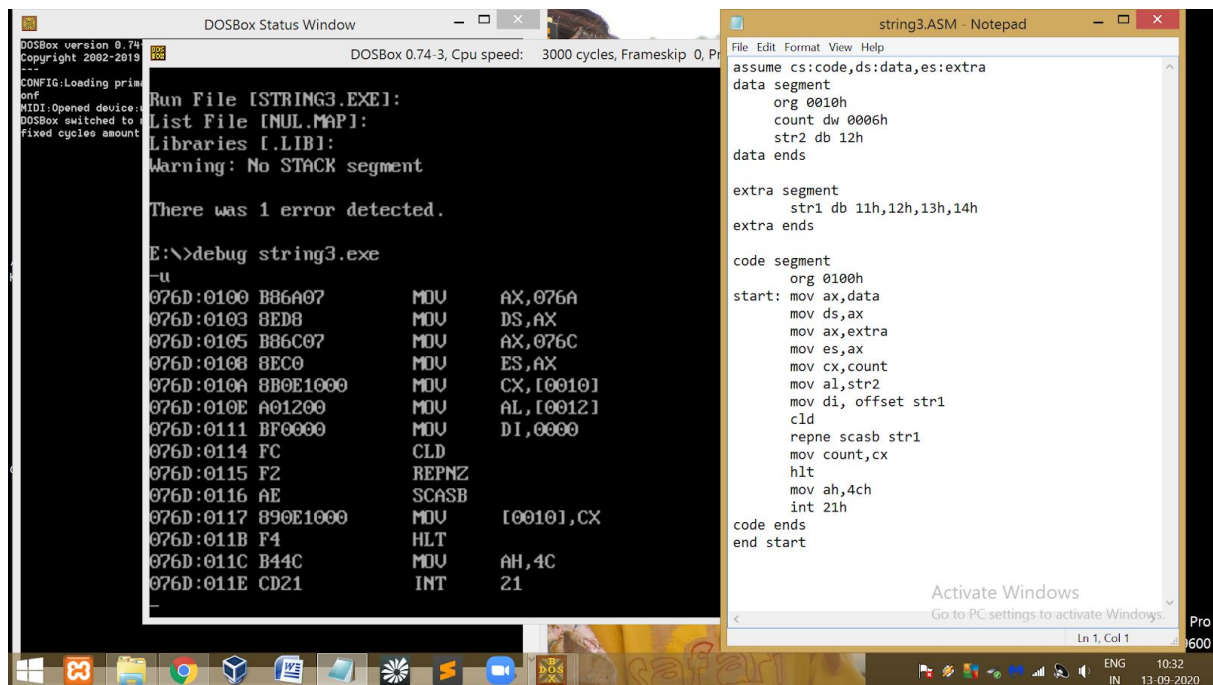
ALGORITHM:

- 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 two windows. The left window is DOSBox Status Window, displaying the execution of string3.exe. The right window is Notepad, showing the assembly code for string3.ASM.

DOSBox Status Window:

```
DOSBox version 0.74-3
Copyright 2002-2019
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: string3.exe
Run File [STRING3.EXE]:
List File [INUL.MAPI]:
Libraries [LIB]:
Warning: No STACK segment
There was 1 error detected.
E:\>debug string3.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        REPNEZ SCASB
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
```

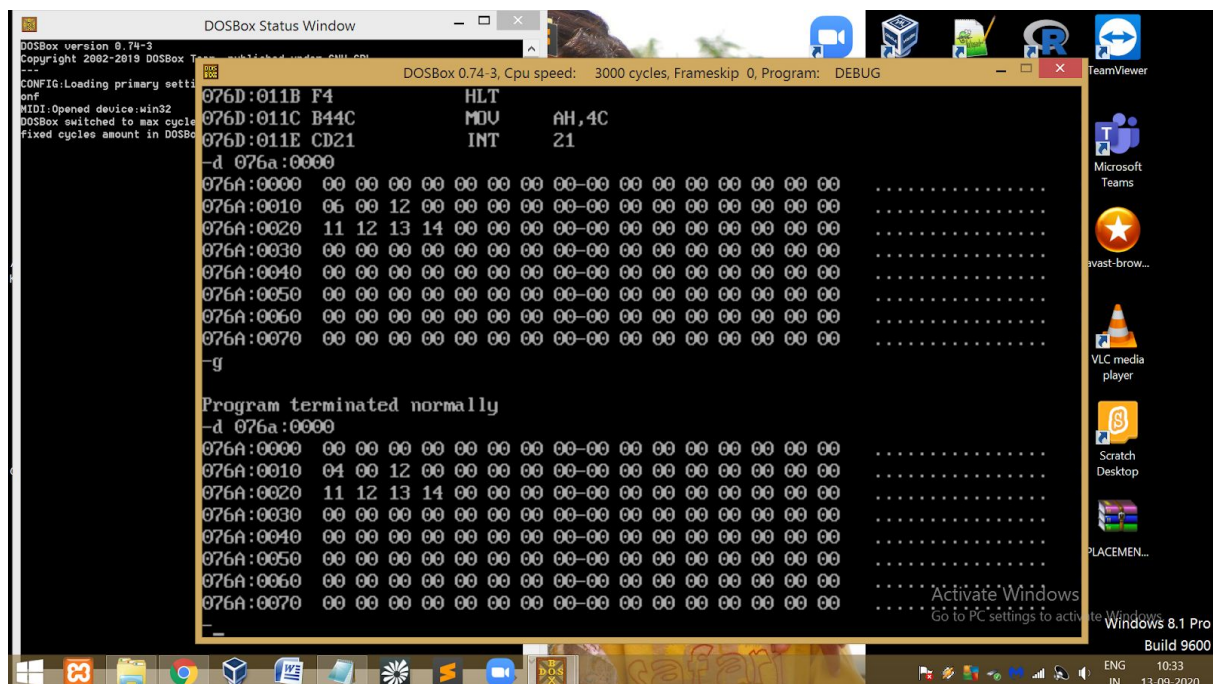
string3.ASM - Notepad:

```
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 DOSBox Status Window displaying the execution of string3.exe. The program has terminated normally, and the memory dump shows the string "11 12 13 14" in memory.

DOSBox Status Window:

```
DOSBox version 0.74-3
Copyright 2002-2019
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG
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
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

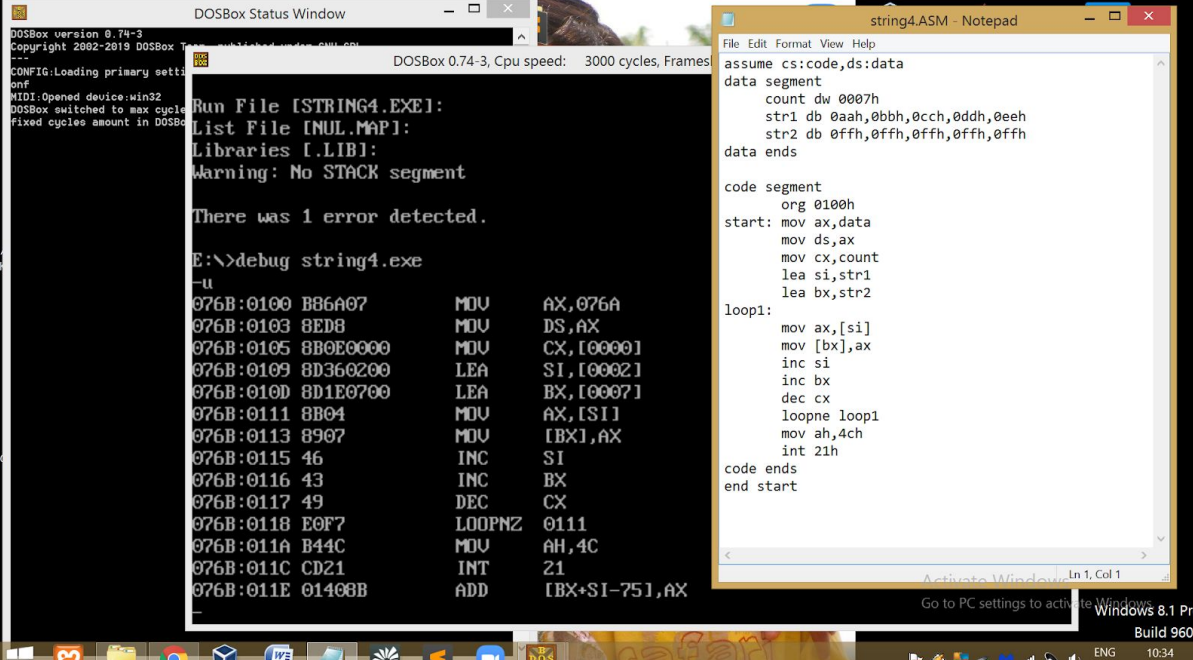
ALGORITHM:

- 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:



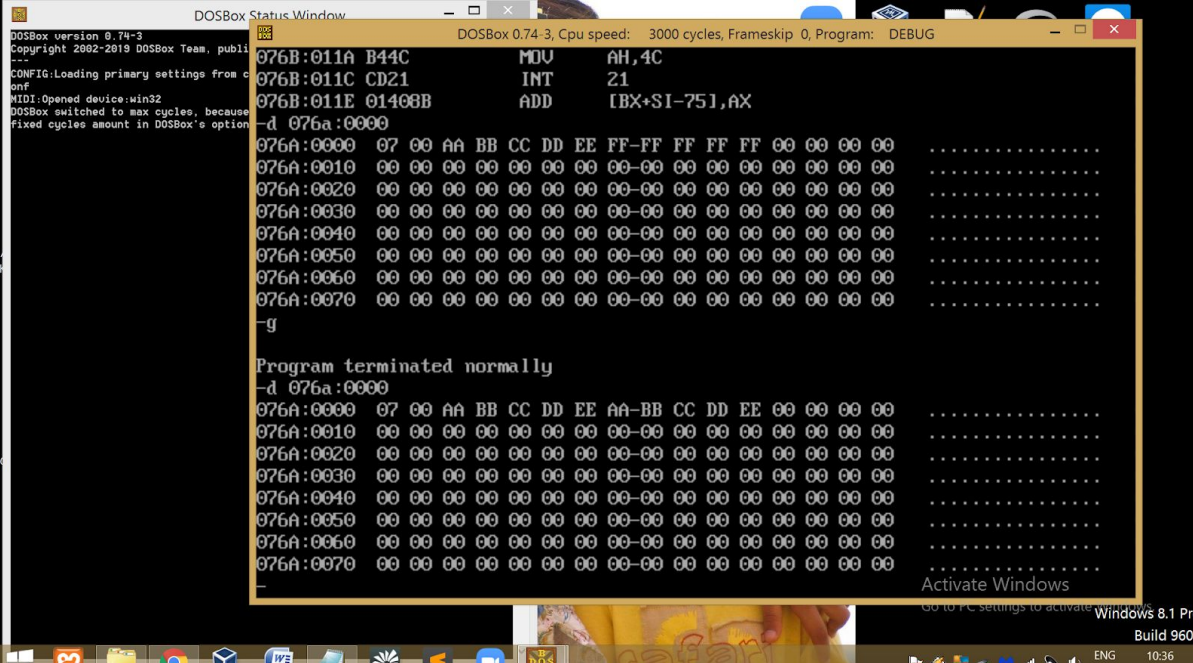
The screenshot shows a DOSBox window on the left and a Notepad window on the right. The DOSBox window displays the execution of string4.exe, showing a warning about no stack segment and a list of assembly instructions. The Notepad window shows the source code of string4.asm, which uses MOV, DS, CX, SI, and BX registers to move a string of bytes.

```
DOSBox Status Window
DOSBox version 0.74-3
Copyright 2002-2019 DOSBox Team, published under GNU GPL
---
CONFIG:Loading primary settings from config.txt
MIDI:Opened device:win32
DOSBox switched to max cycles, because fixed cycles amount in DOSBox's option is not enough
Run File [STRING4.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
Warning: No STACK segment
There was 1 error detected.
E:\>debug string4.exe
-u
076B:0100 B86A07      MOV     AX,076A
076B:0103 8ED8        MOV     DS,AX
076B:0105 8B0E0000      MOV     CX,[0000]
076B:0109 8D360200      LEA     SI,[0002]
076B:010D 8D1E0700      LEA     BX,[0007]
076B:0111 8B04        MOV     AX,[SI]
076B:0113 8907      MOV     [BX],AX
076B:0115 46        INC     SI
076B:0116 43        INC     BX
076B:0117 49        DEC     CX
076B:0118 E0F7      LOOPNZ 0111
076B:011A B44C      MOV     AH,4C
076B:011C CD21      INT     21
076B:011E 01408B      ADD     [BX+SI-75],AX
Program terminated normally
E:\>
```

```
string4.ASM - Notepad
File Edit Format View Help
assume cs:code,ds:data
data segment
    count dw 0007h
    str1 db 0aah,0bbh,0cch,0ddh,0eeh
    str2 db 0ffh,0ffh,0ffh,0ffh,0ffh
data ends

code segment
    org 0100h
start: mov ax,data
        mov ds,ax
        mov cx,count
        lea si,str1
        lea bx,str2
loop1: mov ax,[si]
        mov [bx],ax
        inc si
        inc bx
        dec cx
        loopne loop1
        mov ah,4ch
        int 21h
code ends
end start
```

SAMPLE INPUT/OUTPUT



The screenshot shows the DOSBox window displaying the sample input/output for string4.exe. The program terminates normally, and the memory dump shows the string '07 00 AA BB CC DD EE FF FF FF FF' at address 076A:0000.

```
DOSBox Status Window
DOSBox version 0.74-3
Copyright 2002-2019 DOSBox Team, published under GNU GPL
---
CONFIG:Loading primary settings from config.txt
MIDI:Opened device:win32
DOSBox switched to max cycles, because fixed cycles amount in DOSBox's option is not enough
076B:011A B44C      MOV     AH,4C
076B:011C CD21      INT     21
076B:011E 01408B      ADD     [BX+SI-75],AX
Program terminated normally
E:\>-d 076a:0000
076A:0000 07 00 AA BB CC DD EE FF FF FF FF 00 00 00 00 .....
076A:0010 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 .....
076A:0030 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 .....
076A:0050 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 .....
076A:0070 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
E:\>-g
Program terminated normally
E:\>-d 076a:0000
076A:0000 07 00 AA BB CC DD EE AA-BB CC DD EE 00 00 00 00 .....
076A:0010 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 .....
076A:0030 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 .....
076A:0050 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 .....
076A:0070 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
E:\>
```

RESULT:

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