Experiment No. 7

Name – Mehatab Mahibub Sanadi
Roll No. – CO2056

```
section .data
 menumsg db 10,'##Menu for Non-overlapped Block Transfer##',10
      db 10,'1.Block Transfer without using string instructions'
      db 10,'2.Block Transfer with using string instructions'
      db 10,'3.Exit',10
 menumsg_len equ $-menumsg

 blk_bfrmsg db 10,'Block contents before transfer'
 blk_bfrmsg_len equ $-blk_bfrmsg

 blk_afrmsg db 10,'Block contents after transfer'
 blk_afrmsg_len equ $-blk_afrmsg

 srcmsg db 10,'Source block contents::'
 srcmsg_len equ $-srcmsg

 dstmsg db 10,'Destination block contents::'
 dstmsg_len equ $-dstmsg

 srcblk db 01h,02h,03h,04h,05h
 dstblk db 00,00,00,00,00


 spacechar db 20h
 spchlength equ $-spacechar

;***************.bss Section**************************
section .bss
   optionbuff resb 02
   dispbuff resb 02

%macro display 2
   mov    eax,4    ;print
   mov    ebx,1    ;stdout/screen
   mov    ecx,%1    ;msg
   mov    edx,%2    ;msg_len
   int 80h
%endmacro

%macro accept 2
   mov    eax,3    ;read
   mov    ebx,0    ;stdin/keyboard
   mov    ecx,%1    ;buf
   mov    edx,%2    ;buf_len
```

```
    int 80h
%endmacro

;******************.text Section************************
section .text
    global _start
_start:

    display blk_bfrmsg,blk_bfrmsg_len

    call dispsrc_blk_proc

    call dispdest_blk_proc

   menu: display menumsg,menumsg_len

        accept optionbuff,02

        cmp byte [optionbuff],31h

        je wos

        cmp byte [optionbuff],32h

        je ws


exit:
            mov eax,01
            mov ebx,00
            int 80h

;**************Display Source Block Procedure****************

dispsrc_blk_proc:
        display srcmsg,srcmsg_len
        mov esi,srcblk
        mov ecx,05h

    up1:push ecx
        mov bl,[esi]
        push esi

        call disp8_proc

        display spacechar,spchlength

            pop esi
        inc esi
        pop ecx
```

```
        loop up1
        ret

;**************Display Destination Block Procedure*************

dispdest_blk_proc:
        display dstmsg,dstmsg_len
        mov edi,dstblk
        mov ecx,05

    up2:push ecx
        mov bl,[edi]
        push edi

        call disp8_proc

        display spacechar,spchlength

            pop edi
        inc edi
        pop ecx
        loop up2
        ret

;**************Without String Procedure************
wos:
    mov esi,srcblk
    mov edi,dstblk
mov ecx,05
         again: mov bl,[esi]
                mov [edi],bl
                inc esi
                inc edi
                loop again

         display blk_afrmsg,blk_afrmsg_len
         call dispsrc_blk_proc
         call dispdest_blk_proc
         jmp menu
                                          ;************Using String Procedure***************

ws:
    mov esi,srcblk
    mov edi,dstblk
    mov ecx,05

     cld
                                        rep movsb
; CLD (Clear Direction Flag) = Clears the DF flag in the EFLAGS register. When the
;                                  DF flag is set to 0, string operations increment
```

```
;                                   the index register (SEI and or EDI)

; The direction flag is used to influence the direction in which some of the
; instructions work when used with the REP prefix.

; REP (Repeat) = Repeats a string instruction the number of times specified in the
; count register ((E)CX) or until the indicated condition of the ZF flag is no longer
; met.

; MOVSB = MOVSB copies the byte at [DS:DI] to [ES:SI]. It then increments or
; decrements (depending on the direction flag).


   display blk_afrmsg,blk_afrmsg_len
   call dispsrc_blk_proc
   call dispdest_blk_proc
   jmp menu

;***********Display Procedure*******************

disp8_proc:
   mov esi,dispbuff
   mov ecx,02

dup1:
   rol bl,4
   mov dl,bl
   and dl,0Fh
   cmp dl,09H
   jbe dskip
   add dl,07h

dskip:add dl,30h
     mov [esi],dl
     inc esi
     loop dup1

     display dispbuff,02

     ret
```