Prajwal Khobragade
SYCOB138

Assignment 2

```
;---------------------------------------------------
section .data
pmsg db    10,"Assignment no 2:A",10
    db   10,"Block transfer non overlapped without string instruction",10
pmsg_len equ $-pmsg


bmsg db 10,"Before Transfer: ",10
bmsg_len equ $-bmsg

amsg db 10,"After transfer : ",10
amsg_len equ $-amsg

smsg db 10,"Source block : ",10
smsg_len equ $-smsg

dmsg db 10," Destination block : ",10
dmsg_len equ $-dmsg


sblock db 11h, 22h, 33h, 44h, 55h
dblock db 00h, 00h, 00h, 00h, 00h

nline db 10,10
nline_len equ $-nline

space db  "  "

%macro Print 2
    mov rax,1
```

```
    mov rdi,1
    mov rsi,%1
    mov rdx,%2
    syscall
%endmacro

%macro Read 2
    mov rax,0
    mov rdi,0
    mov rsi,%1
    mov rdx,%2
    syscall
%endmacro
%macro Exit 0
   Print nline,nline_len
   mov rax,60
   mov rdi,0
   syscall
%endmacro



;-----------------------------------------------------------

section .bss
char_ans resb  2     ;char_ans is of 2 byte because we have 2 byte nos

;-----------------------------------------------------------
section .text
global _start
_start:
   Print  pmsg,pmsg_len
```

```
    Print bmsg,bmsg_len  ;block values before transfer
    Print smsg,smsg_len  ;procedure for display sblock
    mov rsi, sblock
    call disp_block

     Print dmsg,dmsg_len  ;procedure for display dblock
    mov rsi, dblock
    call disp_block

call bt_no               ;call for actual block transfer
      Print amsg,amsg_len ;block values before transfer
    Print smsg,smsg_len
    mov rsi, sblock
    call disp_block

     Print dmsg,dmsg_len
    mov rsi, dblock
    call disp_block




;-----------------------------------------------------
bt_no:
    mov rsi, sblock   ;1 memory location of sblock to rsi
    mov rdi, dblock   ;1 memory location of dblock to rdi
    mov rcx, 5        ; intializing the counter to 5

back:   mov al,[rsi] ;copying value of rsi to al register
    mov [rdi],al     ;copying al register value to rdi memory location
```

```
    inc rsi         ;incrementing memory location at rsi
    inc rdi         ;incrementing memory location at rdi

    dec rcx         ;decreament counter
    jnz back        ;jump if not zero

ret
;-------------------------------------------------------
disp_block:
    mov rbp,5       ;initalizing basepointer with 5
    next_num:

    mov al,[rsi]
    push rsi          ;push rsi at top of stack
    call display_8    ;calling to display funnction
    Print space,1

    pop rsi           ;remove rsi from stack
    inc rsi           ;incrementing memory location at source index
    dec rbp           ;decreament basepointer
    jnz next_num      ;jump if not zero to next_num
ret
;-------------------------------------------------------
display_8:
    mov rsi,char_ans+1     ;incrementing memory location of char_ans by 1 and storing it to rsi register
    mov rcx,2              ;initalizing counter to 2
    mov rbx,16             ;moving 16 to rbx for division for division purpose

next_digit:
    xor rdx,rdx            ;making content of rdx register to null
```

```
    div rbx
    cmp dl,9              ;check for remainder in rdx
    jbe add30             ;jump if below or equal
    add dl,07h            ;calculating ascii code
add30:
    add dl,30h            ;calculating ascci code
    mov [rsi],dl          ;storing the result in buffer

    dec rsi
    dec rcx               ; decreament counter
    jnz next_digit        ; jump if not zero

    Print char_ans,2
ret
;-----------------------------------
```

Output:

```
Before Transfer:

Source block :
11 22 33 44 55
 Destination block :
00 00 00 00 00
After transfer :

Source block :
11 22 33 44 55
 Destination block :
11 22 33 44 55
[Execution complete with exit code -11]
```