section .data

array db 10h,20h,30h,40h,50h

msg1 db “Before overlapping : ”,0ah

len equ $-msg1

msg2 db “After overlapping :”,0ah

len2 equ$-msg2

msg3 db ‘’,0ah

len3 equ$-msg3

msg4 db “:”

len4 equ $-msg4

count db 0

count1 db 0

count2 db 0

count 3db 0

count 4 db 0

count 5 db 0

section .bss

addr resb 16

num resb 2

%macro print 2

mov rax,01

mov rdi,01

mov rsi%1

mov rdx,%2

syscall

%endmacro

section .text

global \_start

\_start:

print msg1,len1

xor rsi,rsi ;since we recently used rsi to print msg rsi may contain some values

;therefore we clear the rsi register

mov rsi,array ;we let rsi point to arrays first index location

mov byte[count],05 ;now we assign count=5

up:

mov rbx,rsi ;mov rsi content to rbx

push rsi ;push rsi address to stack

mov rdi,addr ;rdi is used to print values ,we let rdi point to addr buffer

call htoa1 ;to print 16-bit address currently pointed by rdi

pop rsi

mov dl,[rsi] ; we are using dl register to store 2 digits of rsi

push rsi

mov rdi , num1

call htoa2

pop rsi

inc rsi

dec byte[count]

jnz up

;MOVING ARRAY ELEMENT TO 10 AFTER 10 POSITION

mov rsi,array

mov rdi,array+0ah

mov byte[count3],05h

loop10:

mov dl,00h

movsb

dec byte[count3]

jnz loop10

;after third position actual overlapping

xor rsi,rsi

mov rsi,array+3h ;overlapping on pos 3

mov rdi,array+0ah

mov byte[count],05h

loop l1:mov dl,byte[rdi]

mov byte[rsi],dl

inc rsi

inc rdi

dec byte[count5]

jnz loopl1

xor rsi,rsi

mov rsi,array

mov byte[count4],08h

up10:

mov rbx,rsi ;mov rsi content to rbx

push rsi ;push rsi address to stack

mov rdi,addr ;rdi is used to print values ,we let rdi point to addr buffer

call htoa1 ;to print 16-bit address currently pointed by rdi

pop rsi

mov dl,[rsi] ; we are using dl register to store 2 digits of rsi

push rsi

mov rdi , num1

call htoa2

pop rsi

inc rsi

dec byte[count4]

jnz up10

mov rax,60

mov rdi,0

syscall

htoa1:

mov byte[count],16

dup1:

rol rbx,4

mov al,bl

and al,0fh

cmp al,09h

jg p3

add al,30h

jmp p4

p3:add al,37h

p4:mov [rdi],al

inc rdi

dec byte[count1]