*Name: Priyanka Suresh Salunke*

*Class: SE COMP 1*

*Roll no.: 70*

*PRN: F19111151*

***SOURCE CODE:***

%macro IO 4

mov rax,%1

mov rdi,%2

mov rsi,%3

mov rdx,%4

syscall

%endmacro

section .data

m1 db "Enter the five 64 bit numbers:" ,10 ; 10d -> line feed

l1 equ $-m1

m2 db "The five 64 bit numbers are:" ,10

l2 equ $-m2

m3 db "Priyanka Salunke F19111151" ,10

l3 equ $-m3

m4 db "Write an X86/64 ALP to accept five 64 bit Hexadecimal numbers from user and store them in an array and display the accepted numbers." ,10d

l4 equ $-m4

m5 db 10,"Exiting now" ,10

l5 equ $-m5

m6 db "incorrect input error" ,10

l6 equ $-m6

m7 db 10

debug db "debug "

debug\_l equ $-debug

time equ 5

size equ 8

section .bss

arr resb 300

\_input resb 20

\_output resb 20

count resb 1

section .text

global \_start

\_start:

IO 1,1,m3,l3

IO 1,1,m4,l4

mov byte[count],time ; store time = 5 in count;

mov rbp,arr ;rbp points to begining of arr

IO 1,1,m1,l1

input:

IO 0,0,\_input,17

IO 1,1,debug,debug\_l

IO 1,1,\_input,17

call ascii\_to\_hex

mov [rbp],rbx ; put the complete summed rbx value to arr[n]

add rbp,size ; move to next value of array 8 -> 4\*2 = 1 place -> arr[n+1]

dec byte[count] ; loop

jnz input

mov byte[count],time ; set loop count to 5

mov rbp,arr ;make rbp point to arr beginning

jmp display

display:

mov rax,[rbp] ; address of rbp in rax

call hex\_to\_ascii

IO 1,1,m7,1

IO 1,1,\_output,16

add rbp,size ; move to next value of array 8 -> 4\*2 = 1 place arr[n+1]

dec byte[count] ; loop

jnz display

jmp exit

exit:

IO 1,1,m5,l5

mov rax,60

mov rdi,0

syscall

error:

IO 1,1,m6,l6

jmp exit

ascii\_to\_hex:

mov rsi,\_input

mov rcx,16

xor rbx,rbx ;cleaning rbx since rbx == rbx , rbx is set to 0without wasting the space

xor rax,rax ;cleaning rax

letter:

rol rbx,4 ; shifting rbx to left by 4 bytes

mov al,[rsi] ; adrress of rsi (\_input ) in al \_input[0]

cmp al,47h ; error checking

jge error ;

cmp al,39h ;if < ascii 39 => 0-9

jbe skip

sub al,07h ;else => ascii is (A-F)

skip:

sub al,30h ; get value between 0-9

add rbx,rax ; add generated hex value to rbx

inc rsi ; now rsi points at \_input[n+1]

dec rcx ; loop

jnz letter

ret

hex\_to\_ascii:

mov rsi,\_output+15 ;max display of 16 characters and rsi points to \_output[16]

mov rcx,16 ;loop runs 16 times

letter2:

xor rdx,rdx ;cleaning rdx need dl for division remainder and

mov rbx,16 ;base 16

div rbx ;dividing by base 16

cmp dl,09h ;checking if hex value < 9

jbe add30 ;if yes simply add 30h to get the ascii

add dl,07h ;else => (A-F) so add 7 to make it 37 total

add30:

add dl,30h ;common step of adding 30h

mov [rsi],dl ;move generated ascii to \_output[n]

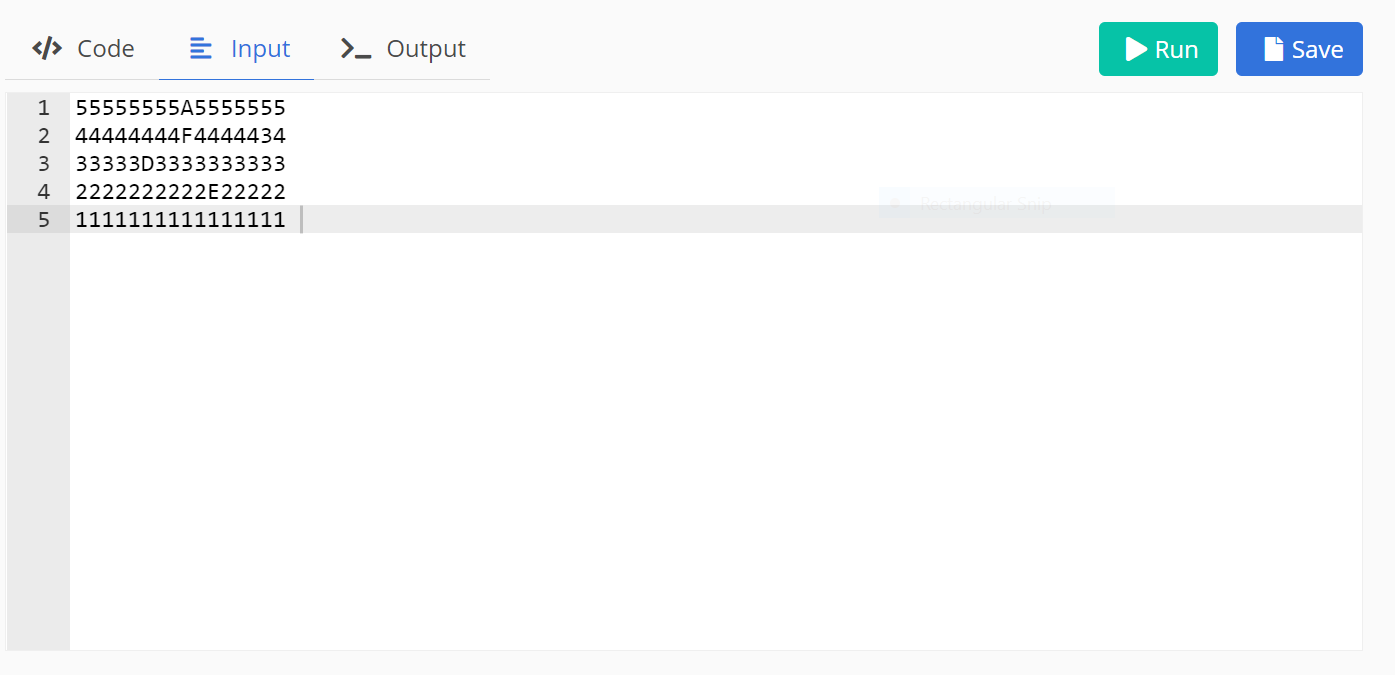
dec rsi ;rsi points to \_output[n-1]

dec rcx ;loop

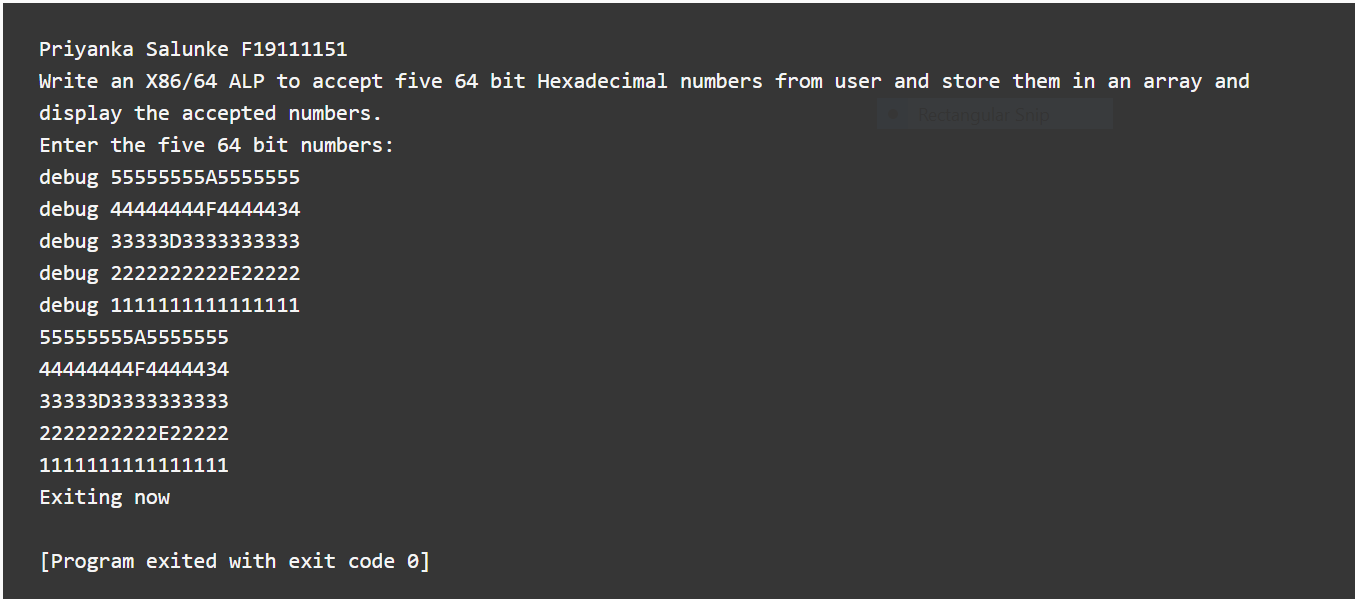
jnz letter2

ret

***INPUT:***

******

***OUTPUT:***

******