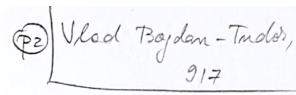
the C-th of N by F. We will consider that the value of N by F. We will consider that the value of h. We will shift EDX to the left of times. Then, we will shift EDX to the left of this will set all a bits from the C-th mibble make this procedure occasible to the main program by making it global.

This is code from module. som:

global swap-to-f segment data me32

regment rode rusisz rwap-to-f:

mor eax, [esp + h]; Moved the dword that needs to be modified in EAX mor eax, [esp + 8]; ECX now has the value C



mor edx, of h ; EDX will be used to change the C-th byte to F ; Now we show to shift EDX to the left Ctimes, where C is now in ECX (by 4 bits) ; If ECX is 0, we don't stood the loop; it ; means that, the o-th byte has to be set 1 to Faling je no-shifting je return if ECX is 8, we don't shift shift - left: she edx, 4 loop. shift_left . no - shifting ; All Shot is left is to opply the OR spelator ; on the with the or lox, edx

. retur :

ret

Now we write-the code in the main module. We will take each dirord in S, find the run of the high word with the low word, and then count the longest sequence of bits of one by consumtive shifts. After this, we have our C and we can opply the procedure swap-to-f. We will make this procedure usable from main. som by telling the ossembler that it is a procedure from enother file. For this, we will use calobel >. Also, not will pass parameters to this procedure by pushing them on

Sobol stort
extern exil, may - to-f, printf
import exit movert all
import printf movert all
regnerat data use 32

A dol 65534, -4473007, 15, -1, 2004322440
len equ (\$- A) 14

r resol len
max-ones resol 1
hexa-output allo "%x ", 0

regment code use 32 ; We'll use ESI to iterate through 5 mor Esi, o · for - welly - alword - in -s: more eax, [sterian] mor lex, lex she ecx, 16 odd ex, ex ; with the above code we've found the sum; of the low word with the high word ; Now we need the longest reprience of ones. more dword [max-ones], o ; imax - ones will have the number of ones in the ; longest seg. of conseculive ones mor ebx,0 ; EBX will key track of the current reg. ; of consecutive ones mor ecx, 16; We shift Ax

. shift - ex:

mov edx, 0

shl ex, 1

edc edx, 0

comp edx, o je. reset i If I we get here => we continue to count add ebx, edx Jup next

. reset:

emp ebx, divoral [max-ones] jb. ebx-to-zero ; We get here -> new longest requence mor dword [max-ones], ebx · ebx - to-zoro:

mov ebx, o

. next:

loop. shift-ax

; With the above code, we've found in max-ones ; the number of ones in the maximum seg of

; Now we need to divide max-ones by 2; This can be done easily by shifting the ; value to the right by one bit. mor ebx, dword [max-ones] shr ebx, 1 ; Now we have our "C" in EBX more eax, [steriary] ; Now we have our rurrent "N" in Ax We just have to call our written procedure ; the proceedured stready takes core of this)

je .dont_call

push ebx

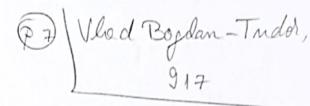
push eax

call swap_to-f

add ey, 2 x h

i and now have the desired value in EAX.

mov [++ esi], lax ; We soved the value in P



; Now we can more on to the next dward in S inc esi emp esi, len jle. for - every-dword-in_s ; At this point, we have our string R and ; we just need to print it more esi, o · for - every - dword-in- 2:

; We push each dword in R on the stock and
; rell printf to print it

push dword [r+ esix4]

push hexa - output

sall [printf]
add esp, 2*4

inc esi

comp esi, len

jb. for _ every - dword _ in _ r

push dword o

coll [exit]