bits 32 ; assembling for the 32 bits architecture

;Se dau doua cuvinte A si B. Sa se obtina un octet C care are:

;pe bitii 0-5, bitii 5-10 ai cuvantului A

;pe bitii 6-7 bitii 1-2 ai cuvantului B.

;Sa se obtina dublucuvantul D care are :

;pe bitii 8-15, bitii lui C

;pe bitii 0-7, bitii 8-15 din B

;pe bitii 24-31, bitii 0-7 din A

;iar pe bitii 16-23, bitii 8-15 din A.

; declare the EntryPoint (a label defining the very first instruction of the program)

global start

; declare external functions needed by our program

extern exit ; tell nasm that exit exists even if we won't be defining it

import exit msvcrt.dll ; exit is a function that ends the calling process. It is defined in msvcrt.dll

; msvcrt.dll contains exit, printf and all the other important C-runtime specific functions

; our data is declared here (the variables needed by our program)

segment data use32 class=data

; ...

a dw 1000000100001010b

b dw 1000001010000010b

c db 0

d dd 0

; our code starts here

segment code use32 class=code

start:

; ...

mov BX, 0 ; in bx we store the result, bx=0

mov AX, [a] ; ax=a

and AX, 0000011111100000b ; isolate bits 5-10 by masking

add CL, 5 ; cl=5

ror AX, CL; rotate the bits from the word to the right with 5 positions

or BX, AX ; put the bits in the final result

mov AX, [b] ;ax=b

and AX, 0000000000000110b; isolate bits 1-2 by masking

mov CL, 5; cl=5

rol AX, CL ; mov the bits to the left with 5 positions

or BX, AX ; add the bits to the result

mov [c], BX ; c=72, memorise the final result

mov ECX, 0 ; ecx=0 , we will store the final result in this register

mov EAX, 0 ; eax=0

mov AX, [b] ; extend the word ax to eax, eax=b

and EAX, 00000000000000001111111100000000b ; mask the bits to isolate the bits 8-15

add CL, 8; cl=8

ror EAX, CL; rotate eax to right with 8 positions

or ECX, EAX ; add the bits to the result

mov EAX, 0; eax=0

mov AX, [c]; extend ax to eax=c

and EAX, 00000000000000000000000011111111b; ,mask the bits to isolate the bits 0-7

mov CL, 8; cl=8

rol EAX, CL; rotate to left eax with 8 positions

or ECX, EAX; append the bits to the result

mov EAX, 0; eax=0

mov AX, [a]; extend ax to eax=0

and EAX, 00000000000000001111111100000000b; mask the bits to isolate the bits 8-15

mov CL, 8;cl=8

rol EAX, CL; rotate to left eax with 8 positions

or ECX, EAX ; add the bits to result

mov EAX, 0; eax=0

mov AX, [a] ; extend ax to eax=a

and EAX, 00000000000000000000000011111111b; mask the other bits to isolate the bits 0-7

mov CL, 24; cl=24

rol EAX, CL ; rotate to the left with 24 positions the bits in eax

or ECX, EAX ; move the bits in ecx

mov [d], ECX ; store the result in the double word d variable, d= 176244866

; exit(0)

push dword 0 ; push the parameter for exit onto the stack

call [exit] ; call exit to terminate the program

