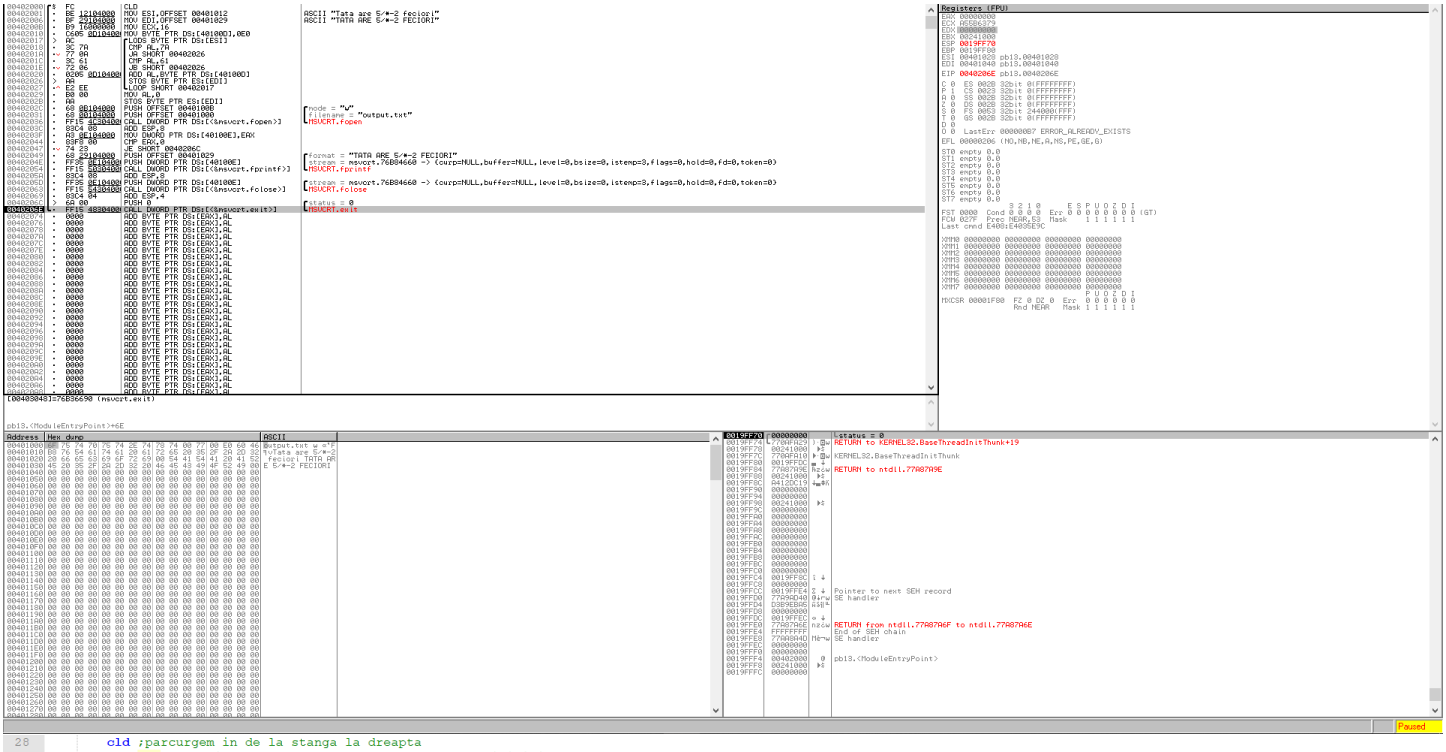
13. Se dau un nume de fisier si un text (definite in segmentul de date). Textul contine litere mici, litere mari, cifre si caractere speciale. Sa se transforme toate literele mici din textul dat in litere mari. Sa se creeze un fisier cu numele dat si sa se scrie textul obtinut in fisier.

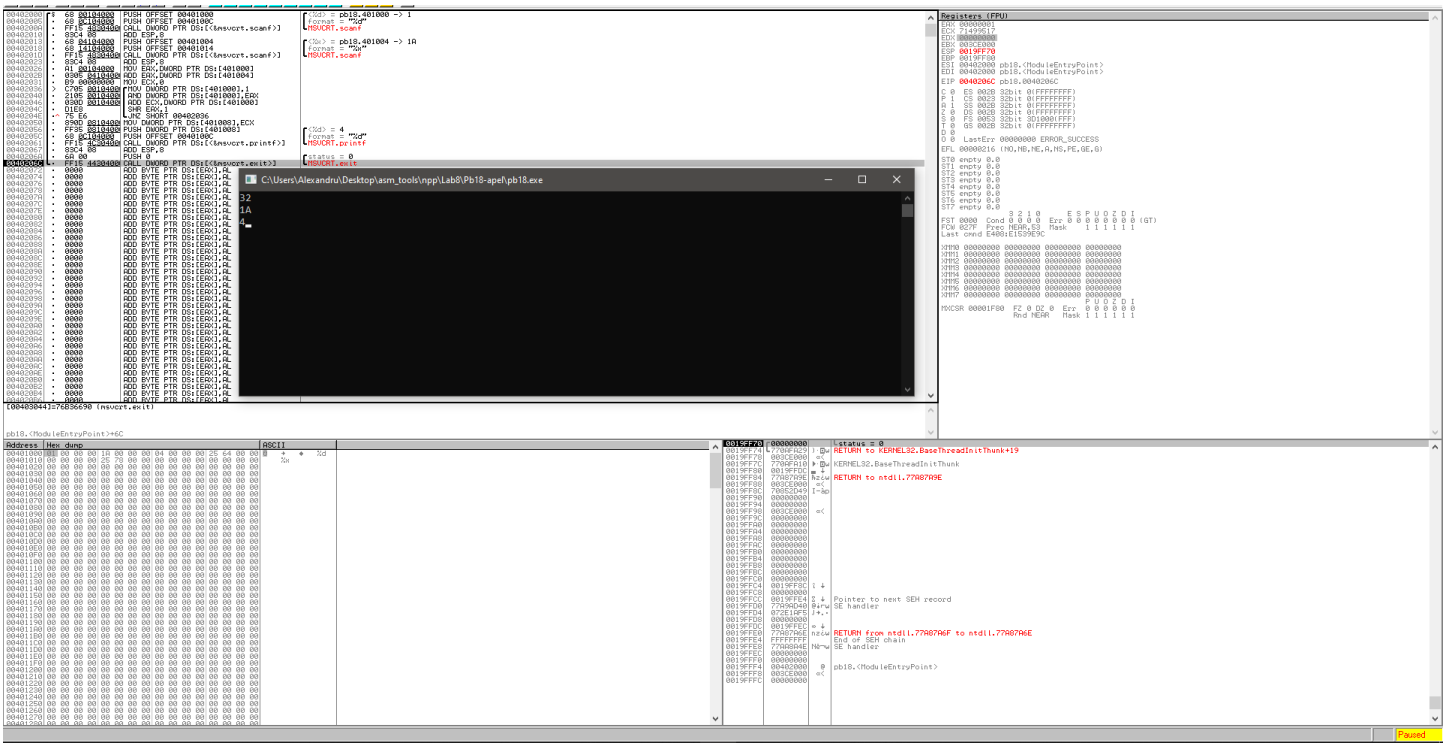


```
    cld ;parcurgem in de la stanga la dreapta
    mov ESI,text ;in SOURCE INDEX punem adresa textului initial
    mov EDI,nw_text ; in DESTINATION INDEX punem adresa textului de construit
    mov ECX,l ;pune in ECX lungimea sirului
    mov byte[dif],'A'-'a';
    st_loop:
        lodsb ;luam in AL caracterul din text
        ;verificam if('a'<=c && c<='z')
        cmp AL,'z'
        ja et1
        cmp AL,'a'
        jb et2
            add AL,[dif] ;in caz afirmativ adaugam cont
        et1:
        et2:
        stosb ;pune in sirul destinatie pe AL
    loop st_loop

    mov AL,0 ; adaugam 0-ul final pentru a putea afisa in fisier
    stosb

    ;apelam fopen pentru a deschide fisierul
    ;EAX=fopen(nume_fisier,acces_mode)
    push dword acces_mode
    push dword nume_fisier
    call [fopen]
    add ESP,4*2

    mov [description_fis],EAX ;salvam valoare returnata in EAX in description_fis

    cmp EAX,0
    je final

        ;scriem testul in fisier
        ;fprintf(description_fis,nw_text)
        push dword nw_text
        push dword [description_fis]
        call [fprintf]
        add ESP,4*2

        ;inchidem fisierul
        ;fclose(description_fis)
        push dword [description_fis]
        call [fclose]
        add ESP,4

    final:
```

```
bits 32 ; assembling for the 32 bits architecture
; declare the EntryPoint (a label defining the very first instruction of the
program)
global start
; declare external functions needed by our program
extern exit,fopen,fprintf,fclose
import exit msvcrt.dll    ; exit is a function that ends the calling process.
import fopen msvcrt.dll   ; msvcrt.dll contains exit, printf
import fprintf msvcrt.dll
import fclose msvcrt.dll

; our data is declared here (the variables needed by our program)
segment data use32 class=data
    nume_fisier db "output.txt",0
    acces_mode db "w",0
    dif db 0
    description_fis dd -1

    text db "Tata are 5/*-2 feciori",0
    l equ $-text-1
    nw_text times l+1 db 0

; our code starts here
segment code use32 class=code
    start:
        ;pentru modificarea textului vom folosi operatii pe siruri
        cld ;parcurgem in de la stanga la dreapta
        mov ESI,text ;in SOURCE INDEX punem adresa textului initial
        mov EDI,nw_text ; in DESTINATION INDEX punem adresa textului de construit
        mov ECX,l ;pune in ECX lungimea sirului
        mov byte[dif],'A'-'a';
        st_loop:
            lodsb ;luam in AL caracterul din text
            ;verificam if('a'<=c && c<='z')
            cmp AL,'z'
            ja et1
            cmp AL,'a'
            jb et2
                add AL,[dif] ;in caz afirmativ adaugam cont
            et1:
            et2:
            stosb ;pune in sirul destinatie pe AL
        loop st_loop

        mov AL,0 ; adaugam 0-ul final pentru a putea afisa in fisier
        stosb

        ;apelam fopen pentru a deschide fisierul
        ;EAX=fopen(nume_fisier,acces_mode)
        push dword acces_mode
        push dword nume_fisier
        call [fopen]
        add ESP,4*2
```

```
mov [description_fis],EAX ;salvam valoare returnata in EAX in description_fis

        cmp EAX,0
        je final

            ;scriem testul in fisier
            ;fprintf(description_fis,nw_text)
            push dword nw_text
            push dword [description_fis]
            call [fprintf]
            add ESP,4*2

            ;inchidem fisierul
            ;fclose(description_fis)
            push dword [description_fis]
            call [fclose]
            add ESP,4

        final:

        push    dword 0      ; push the parameter for exit onto the stack
        call    [exit]       ; call exit to terminate the program
```

18. Sa se citeasca de la tastatura un numar in baza 10 si un numar in baza 16. Sa se afiseze in baza 10 numarul de biti 1 ai sumei celor doua numere citite. Exemplu:
    a = 32 = 0010 0000b
    b = 1Ah = 0001 1010b
    32 + 1Ah = 0011 1010b
    Se va afisa pe ecran valoarea 4.



```
segment data use32 class=data
    a dd 0
    b dd 0
    cont dd 0
    format_dec dd "%d",0
    format_hex dd "%x",0

; our code starts here
segment code use32 class=code
    start:

        ;scanf("%d",&a)

        push dword a
        push dword format_dec
        call [scanf]
        add ESP,4*2

        ;scanf("%x",&b)

        push dword b
        push dword format_hex
        call [scanf]
        add ESP,4*2

        mov EAX,[a]
        add EAX,[b]

        mov ECX,0 ;in ECX se numara bitii de 1
        ;numaram biti de 1 ai rezultatului
        cnt:
            mov dword[a],1
            and [a],EAX
            add ECX,[a]
            shr EAX,1
        jnz cnt

        mov [cont],ECX ;punem in cont ce vrem sa afisam
        ;printf("%d",cont)

        push dword [cont]
        push dword format_dec
        call [printf]
        add ESP,4*2

        push    dword 0      ; push the parameter for exit onto the stack
        call    [exit]       ; call exit to terminate the program
```

```
bits 32
global start
extern exit,scanf,printf
import exit msvcrt.dll    ; exit is a function that ends the calling process.
import scanf msvcrt.dll   ; msvcrt.dll contains exit, printf
import printf msvcrt.dll

; our data is declared here (the variables needed by our program)
segment data use32 class=data
    a dd 0
    b dd 0
    cont dd 0
    format_dec dd "%d",0
    format_hex dd "%x",0

; our code starts here
segment code use32 class=code
    start:

        ;scanf("%d",&a)

        push dword a
        push dword format_dec
        call [scanf]
        add ESP,4*2

        ;scanf("%x",&b)

        push dword b
        push dword format_hex
        call [scanf]
        add ESP,4*2

        mov EAX,[a]
        add EAX,[b]

        mov ECX,0 ;in ECX se numara bitii de 1
        ;numaram biti de 1 ai rezultatului
        cnt:
            mov dword[a],1
            and [a],EAX
            add ECX,[a]
            shr EAX,1
        jnz cnt

        mov [cont],ECX ;punem in cont ce vrem sa afisam
        ;printf("%d",cont)

        push dword [cont]
        push dword format_dec
        call [printf]
        add ESP,4*2

        push    dword 0      ; push the parameter for exit onto the stack
        call    [exit]       ; call exit to terminate the program
```