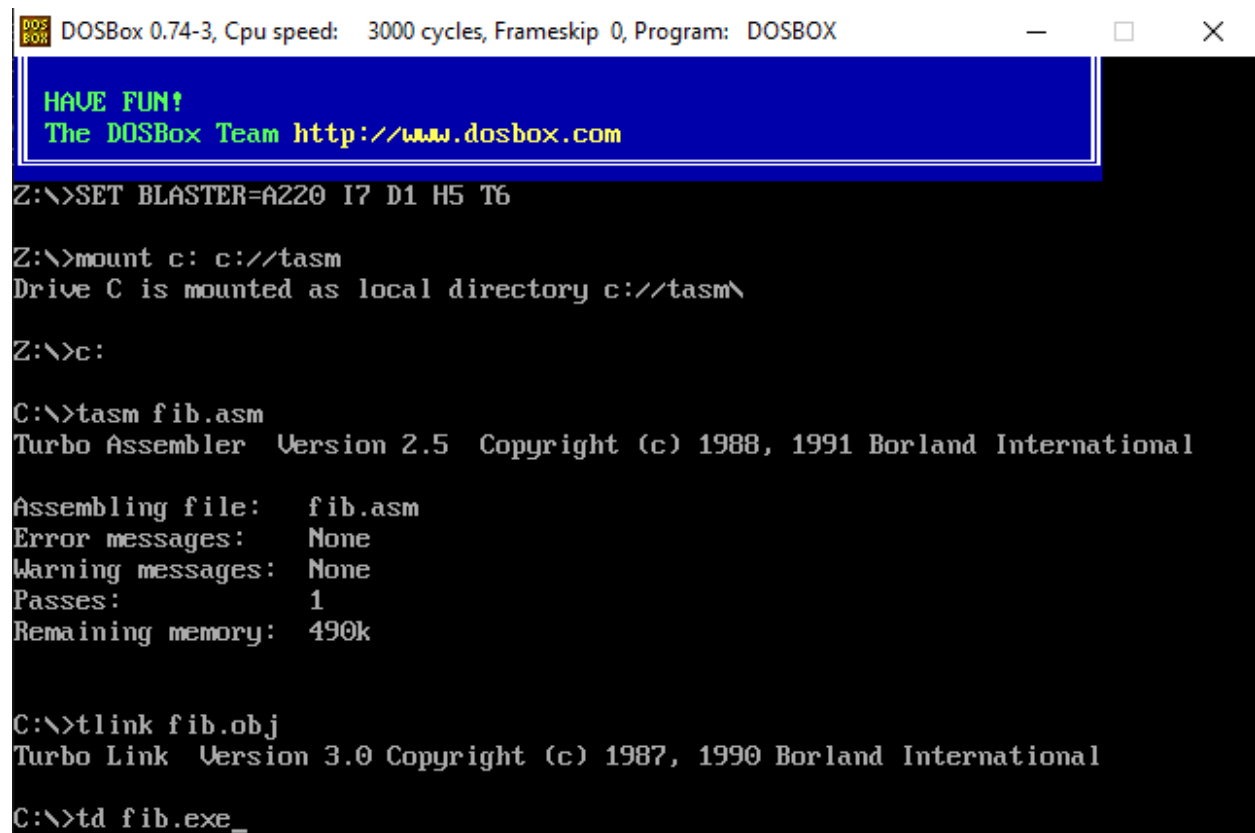


1. Enunțarea scopului programului

Programul calculează primele 10 elemente ale sirului lui Fibonacci prin introducerea de la tastatură a poziției dorite. (pentru prima cifră a sirului se apasă tasta 0 iar pentru ultima tasta 9)

2. Obținerea fișierului executabil



```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX
HAVE FUN!
The DOSBox Team http://www.dosbox.com
Z:\>SET BLASTER=A220 I7 D1 H5 T6
Z:\>mount c: c://tasm
Drive C is mounted as local directory c://tasm\
Z:\>c:
C:\>tasm fib.asm
Turbo Assembler Version 2.5 Copyright (c) 1988, 1991 Borland International
Assembling file: fib.asm
Error messages: None
Warning messages: None
Passes: 1
Remaining memory: 490k
C:\>tlink fib.obj
Turbo Link Version 3.0 Copyright (c) 1987, 1990 Borland International
C:\>td fib.exe_
```

3. Codul sursă explicat

.model small ; se specifică modelul de memorie SMALL

.stack 100h ; se definește o stivă de 256 octeți

.data ; marchează începutul segmentului de date

mesaj db 13,10,'Introduceti cifra: \$' ; ; se definește și se inițializează mesajul care va fi afișat de program pentru introducerea cifrei

.code ; marchează începutul segmentului de cod al programului

start: ; eticheta de început a programului

| | |
|---|--|
| mov ax, @data ; mov ds, ax | ; încarcă în registrul DS adresa de bază a segmentului de date |
|---|--|

mov dx, offset mesaj ; se încarcă în registrul DX deplasamentul (offset-ul) mesajului pentru ;
introducerea cifrei

| | |
|--------------------------------------|--|
| mov ah, 09h int 21h | ; afișează pe ecran mesajul Introduceți o cifră: DS:DX conține adresa de început a șirului de caractere |
|--------------------------------------|--|

| | |
|--------------------------------------|---|
| mov ah, 01h int 21h | ; citește de la tastatură cu ecou un caracter (cifra introdusă) după execuție în registrul AL se află codul ASCII al caracterului tastat |
|--------------------------------------|---|

sub ax, 130h ; deoarece în registrul ax se afla 130h + cifra introdusă, se scade valoarea 130h

mov cx, ax ; se muta în registrul cx valoarea registrului ax (pentru funcția loop de mai jos)

mov dx, cx ; se muta în registrul dx valoarea registrului cx (Fibonacci 0,1,1)

cmp dl, 0 ; se compară valoarea introdusă cu '0'

je sfarsit0 ; dacă val. introdusă e '0' instrucțiunea je (jump equal) realizează saltul la eticheta
sfarsit0

cmp dl, 1 ; se compară valoarea introdusă cu '1'

je sfarsit1 ; dacă val. introdusă e '1' instrucțiunea je (jump equal) realizează saltul la eticheta
sfarsit1

cmp dl, 2 se compară valoarea introdusă cu '2'

je sfarsit2 ; daca val introdusa e '2' instructiunea je (jump equal) realizeaza saltul la eticheta sfarsit2

mov ax, 0 ; daca tasta apasata nu este 0, 1 sau 2, registrul ax ia valoarea primului termen al sirului

mov bx, 1 ; registrul bx ia valoarea celui de-al doilea termen al sirului

sub cx, 2 ; se scad cei doi termini din loop

repetă:

| | | |
|--------------------|---|---|
| add ax, bx | } | ; instructiuni folosite pentru determinarea valorilor din sir |
| mov dx, ax | | |
| mov ax, bx | | |
| mov bx, dx | | |
| loop repeta | | |

add dx, ax ; in registrul dx se afla valoarea dorita in hexazecimal\

| |
|--|
| mov ax, 4Ch ; întrerup execuția programului int 21h |
|--|

sfarsit0:

| |
|--|
| mov ax, 4Ch ; întrerup execuția programului int 21h |
|--|

sfarsit1:

| |
|--------------------------------------|
| mov ax, 4Ch int 21h |
|--------------------------------------|

sfarsit2:

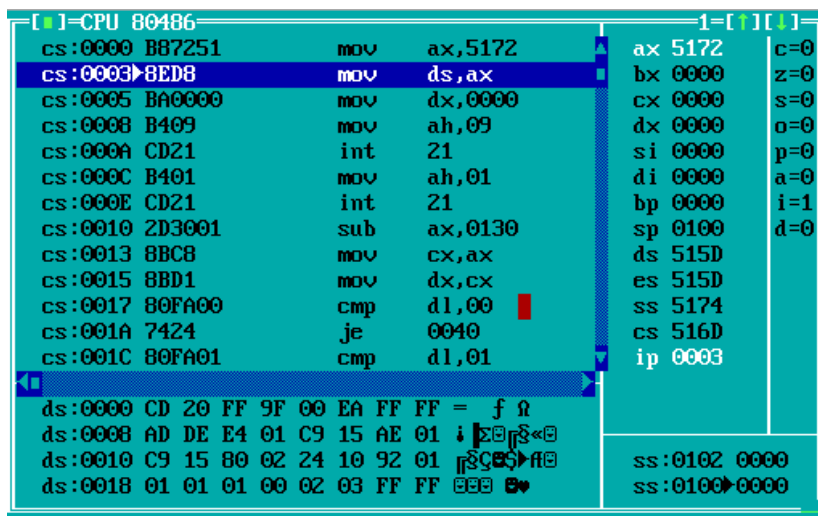
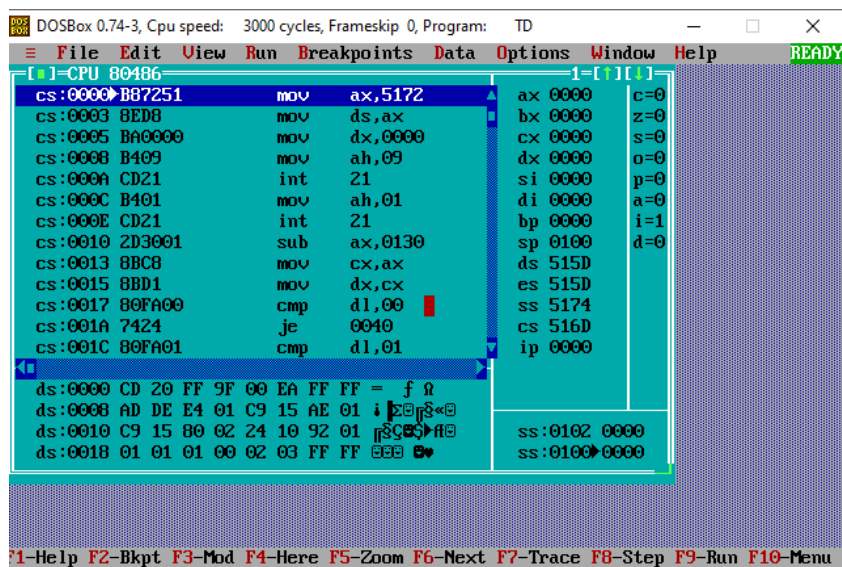
dec dx ; in acest caz, decrementez registrul dx pentru afisarea valorii

mov ax, 4Ch ; întrerup execuția programului

int 21h

end start ; t ; directiva de terminare a codului programului

4. Programul in Debugger



Registrul ax ia valoarea 5172h (a tuturor datelor)

```

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: TD
Z:\>SET BLASTER=A220 I7 D1 H5 T6

Z:\>mount c: c://tasm
Drive C is mounted as local directory c://tasm\

Z:\>c:

C:\>tasm fib.asm
Turbo Assembler Version 2.5 Copyright (c) 1988, 1991 Borland International

Assembling file: fib.asm
Error messages: None
Warning messages: None
Passes: 1
Remaining memory: 490k

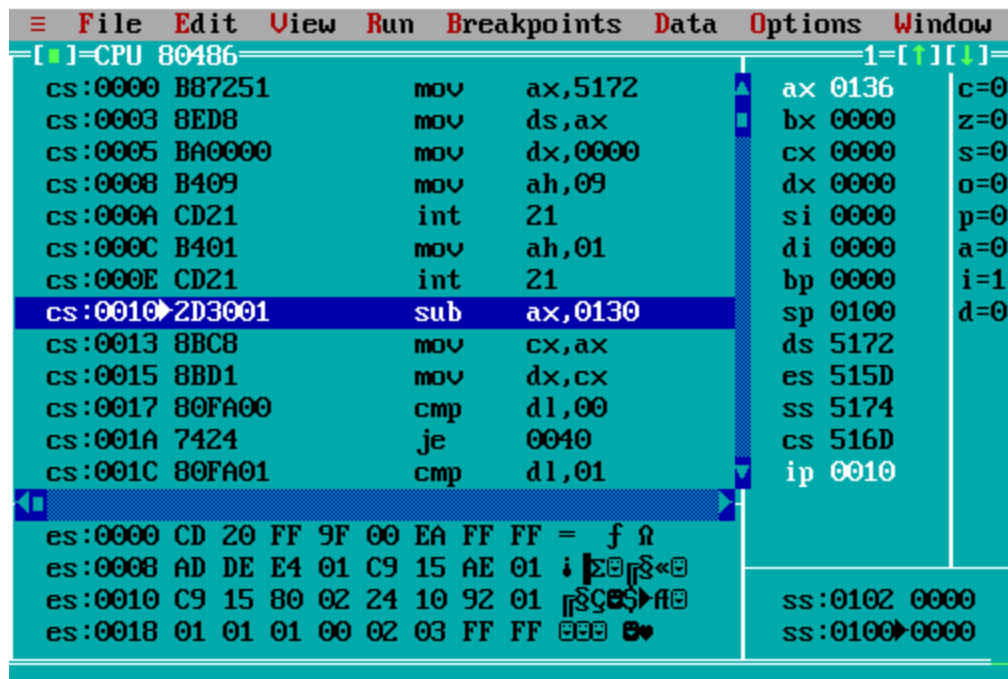
C:\>tlink fib.obj
Turbo Link Version 3.0 Copyright (c) 1987, 1990 Borland International

C:\>td fib.exe
Turbo Debugger Version 3.2 Copyright (c) 1988,92 Borland International

Introduceti cifra: _

```

Dupa instructiunile: "mov ah, 01 ; int 21" va aparea consola unde putem introduce cifrele (Exemplu: cifra 6)



Cifra apare in registrul ax (= 130h + 6h)

| []-CPU 80486 | | | | 1=[↑][↓] | | |
|--|--------|-----|---------|------------------------------|------|-----|
| cs:0000 | B87251 | mov | ax,5172 | ax | 0006 | c=0 |
| cs:0003 | 8ED8 | mov | ds,ax | bx | 0000 | z=0 |
| cs:0005 | BA0000 | mov | dx,0000 | cx | 0000 | s=0 |
| cs:0008 | B409 | mov | ah,09 | dx | 0000 | o=0 |
| cs:000A | CD21 | int | 21 | si | 0000 | p=1 |
| cs:000C | B401 | mov | ah,01 | di | 0000 | a=0 |
| cs:000E | CD21 | int | 21 | bp | 0000 | i=1 |
| cs:0010 | 2D3001 | sub | ax,0130 | sp | 0100 | d=0 |
| cs:0013 | 8BC8 | mov | cx,ax | ds | 5172 | |
| cs:0015 | 8BD1 | mov | dx,cx | es | 515D | |
| cs:0017 | 80FA00 | cmp | dl,00 | ss | 5174 | |
| cs:001A | 7424 | je | 0040 | cs | 516D | |
| cs:001C | 80FA01 | cmp | dl,01 | ip | 0013 | |
| es:0000 CD 20 FF 9F 00 EA FF FF = f Ω es:0008 AD DE E4 01 C9 15 AE 01 i Σ ⊞ S < ⊞ es:0010 C9 15 80 02 24 10 92 01 SCS ⊞ f ⊞ es:0018 01 01 01 00 02 03 FF FF ⊞ ⊞ ⊞ | | | | ss:0102 0000 ss:0100 0000 | | |

Dupa instructiunea sub ax, 0130; registrul va avea cifra tastata in valoare hexazecimala

| []-CPU 80486 | | | | 1=[↑][↓] | | |
|--|--------|-----|---------|------------------------------|------|-----|
| cs:0000 | B87251 | mov | ax,5172 | ax | 0006 | c=0 |
| cs:0003 | 8ED8 | mov | ds,ax | bx | 0000 | z=0 |
| cs:0005 | BA0000 | mov | dx,0000 | cx | 0006 | s=0 |
| cs:0008 | B409 | mov | ah,09 | dx | 0000 | o=0 |
| cs:000A | CD21 | int | 21 | si | 0000 | p=1 |
| cs:000C | B401 | mov | ah,01 | di | 0000 | a=0 |
| cs:000E | CD21 | int | 21 | bp | 0000 | i=1 |
| cs:0010 | 2D3001 | sub | ax,0130 | sp | 0100 | d=0 |
| cs:0013 | 8BC8 | mov | cx,ax | ds | 5172 | |
| cs:0015 | 8BD1 | mov | dx,cx | es | 515D | |
| cs:0017 | 80FA00 | cmp | dl,00 | ss | 5174 | |
| cs:001A | 7424 | je | 0040 | cs | 516D | |
| cs:001C | 80FA01 | cmp | dl,01 | ip | 0015 | |
| es:0000 CD 20 FF 9F 00 EA FF FF = f Ω es:0008 AD DE E4 01 C9 15 AE 01 i Σ ⊞ S < ⊞ es:0010 C9 15 80 02 24 10 92 01 SCS ⊞ f ⊞ es:0018 01 01 01 00 02 03 FF FF ⊞ ⊞ ⊞ | | | | ss:0102 0000 ss:0100 0000 | | |

Apoi cifra este mutata in registrul cx iar ulterior in dx

```

[ ]=CPU 80486
cs:000C B401      mov     ah,01
cs:000E CD21      int      21
cs:0010 2D3001    sub      ax,0130
cs:0013 8BC8      mov      cx,ax
cs:0015 8BD1      mov      dx,cx
cs:0017 80FA00    cmp      dl,00
cs:001A 7424      je        0040
cs:001C 80FA01    cmp      dl,01
cs:001F 7424      je        0045
cs:0021 80FA02    cmp      dl,02
cs:0024 7424      je        004A
cs:0026 B80000    mov      ax,0000
cs:0029 BB0100    mov      bx,0001
es:0000 CD 20 FF 9F 00 EA FF FF = f Ω
es:0008 AD DE E4 01 C9 15 AE 01 i 20 13 <
es:0010 C9 15 80 02 24 10 92 01 13 05 > ff
es:0018 01 01 01 00 02 03 FF FF 00 00

ax 0006  c=0
bx 0000  z=0
cx 0006  s=0
dx 0006  o=0
si 0000  p=0
di 0000  a=0
bp 0000  i=1
sp 0100  d=0
ds 5172
es 515D
ss 5174
cs 516D
ip 0026

ss:0102 0000
ss:0100 0000

```

Intruciunile incercuite compara cifra introdusa cu 0, 1 si 2 iar daca valoarea este egala cu acestea, programul sare prin intruciunea "je" la etichetele sfarsit0, sfarsit1 si sfarsit2

```

[ ]=CPU 80486
cs:0026 B80000    mov      ax,0000
cs:0029 BB0100    mov      bx,0001
cs:002C 83E902    sub      cx,0002
cs:002F 03C3      add      ax,bx
cs:0031 8BD0      mov      dx,ax
cs:0033 8BC3      mov      ax,bx
cs:0035 8BDA      mov      bx,dx
cs:0037 E2F6      loop     002F
cs:0039 03D0      add      dx,ax
cs:003B B84C00    mov      ax,004C
cs:003E CD21      int      21
cs:0040 B84C00    mov      ax,004C
cs:0043 CD21      int      21
es:0000 CD 20 FF 9F 00 EA FF FF = f Ω
es:0008 AD DE E4 01 C9 15 AE 01 i 20 13 <
es:0010 C9 15 80 02 24 10 92 01 13 05 > ff
es:0018 01 01 01 00 02 03 FF FF 00 00

ax 0000  c=0
bx 0001  z=0
cx 0004  s=0
dx 0006  o=0
si 0000  p=0
di 0000  a=0
bp 0000  i=1
sp 0100  d=0
ds 5172
es 515D
ss 5174
cs 516D
ip 002F

ss:0102 0000
ss:0100 0000

```

Daca valoarea nu este 0 1 sau 2, registrii ax si bx iau primele doua valori ale sirului iar din registrul cx, se scad (sub cx, 0002)

| CPU 80486 | | | Registers | |
|--|--------|-------------|------------------------------|------|
| cs:0026 | B80000 | mov ax,0000 | ax | 0003 |
| cs:0029 | BB0100 | mov bx,0001 | bx | 0005 |
| cs:002C | 83E902 | sub cx,0002 | cx | 0000 |
| cs:002F | 03C3 | add ax,bx | dx | 0005 |
| cs:0031 | 8BD0 | mov dx,ax | si | 0000 |
| cs:0033 | 8BC3 | mov ax,bx | di | 0000 |
| cs:0035 | 8BDA | mov bx,dx | bp | 0000 |
| cs:0037 | E2F6 | loop 002F | sp | 0100 |
| cs:0039 | 03D0 | add dx,ax | ds | 5172 |
| cs:003B | B84C00 | mov ax,004C | es | 515D |
| cs:003E | CD21 | int 21 | ss | 5174 |
| cs:0040 | B84C00 | mov ax,004C | cs | 516D |
| cs:0043 | CD21 | int 21 | ip | 0039 |
| es:0000 CD 20 FF 9F 00 EA FF FF = f 0 es:0008 AD DE E4 01 C9 15 AE 01 i 0 0 0 0 es:0010 C9 15 80 02 24 10 92 01 0 0 0 0 es:0018 01 01 01 00 02 03 FF FF 0 0 0 0 | | | ss:0102 0000 ss:0100 0000 | |

Instructiunile incercuite se realizeaza de cx ori iar dupa parcurgerea instructiunii loop 002F registrii vor avea valorile finale

| CPU 80486 | | | Registers | |
|--|--------|-------------|------------------------------|------|
| cs:0026 | B80000 | mov ax,0000 | ax | 0003 |
| cs:0029 | BB0100 | mov bx,0001 | bx | 0005 |
| cs:002C | 83E902 | sub cx,0002 | cx | 0000 |
| cs:002F | 03C3 | add ax,bx | dx | 0008 |
| cs:0031 | 8BD0 | mov dx,ax | si | 0000 |
| cs:0033 | 8BC3 | mov ax,bx | di | 0000 |
| cs:0035 | 8BDA | mov bx,dx | bp | 0000 |
| cs:0037 | E2F6 | loop 002F | sp | 0100 |
| cs:0039 | 03D0 | add dx,ax | ds | 5172 |
| cs:003B | B84C00 | mov ax,004C | es | 515D |
| cs:003E | CD21 | int 21 | ss | 5174 |
| cs:0040 | B84C00 | mov ax,004C | cs | 516D |
| cs:0043 | CD21 | int 21 | ip | 003B |
| es:0000 CD 20 FF 9F 00 EA FF FF = f 0 es:0008 AD DE E4 01 C9 15 AE 01 i 0 0 0 0 es:0010 C9 15 80 02 24 10 92 01 0 0 0 0 es:0018 01 01 01 00 02 03 FF FF 0 0 0 0 | | | ss:0102 0000 ss:0100 0000 | |

Dupa apelarea functiei "add dx,ax", valoarea Dorita se va afla in registrul dx sub forma hexazecimala (In acest caz, 0008 in hexazecimal este 8, a 6a valoare a sirului lui Fibonacci (am luat 0 ca a 0-a valoare a sirului)

| [CPU 80486] | | | 1-[↑][↓] | |
|-------------|--------|-------------|----------|-----|
| cs:0026 | B80000 | mov ax,0000 | ax 004C | c=0 |
| cs:0029 | BB0100 | mov bx,0001 | bx 0005 | z=0 |
| cs:002C | 83E902 | sub cx,0002 | cx 0000 | s=0 |
| cs:002F | 03C3 | add ax,bx | dx 0008 | o=0 |
| cs:0031 | 8BD0 | mov dx,ax | si 0000 | p=0 |
| cs:0033 | 8BC3 | mov ax,bx | di 0000 | a=0 |
| cs:0035 | 8BDA | mov bx,dx | bp 0000 | i=1 |
| cs:0037 | E2F6 | loop 002F | sp 0100 | d=0 |
| cs:0039 | 03D0 | add dx,ax | ds 5172 | |
| cs:003B | B84C00 | mov ax,004C | es 515D | |
| cs:003E | CD21 | int 21 | ss 5174 | |
| cs:0040 | B84C00 | mov ax,004C | cs 516D | |
| cs:0043 | CD21 | int 21 | ip 003E | |

| | | | |
|---------|-------------------------|-------------------|--|
| es:0000 | CD 20 FF 9F 00 EA FF FF | = f Ω | |
| es:0008 | AD DE E4 01 C9 15 AE 01 | ! 2 3 4 5 6 7 8 9 | |
| es:0010 | C9 15 80 02 24 10 92 01 | 1 2 3 4 5 6 7 8 | |
| es:0018 | 01 01 01 00 02 03 FF FF | 0 1 2 3 4 5 6 7 | |

| | |
|---------|------|
| ss:0102 | 0000 |
| ss:0100 | 0000 |

Dupa apelul int 21h, programul se termina.

```

ds:0000 0D 0A 49 6E 74 72 6F 64 J Introd
ds:0008 75 63 65 74 69 20 63 69 uceti ci
ds:0010 66 72 61 3A 20 24 00 00 fra: $
ds:0018 00 00 00 00 00 00 00 00
ds:0020 00 00 00 00 00 00 00 00

```

Harta memoriei dupa comanda Ctrl + G si tastarea ds:0000