Tema laborator 3

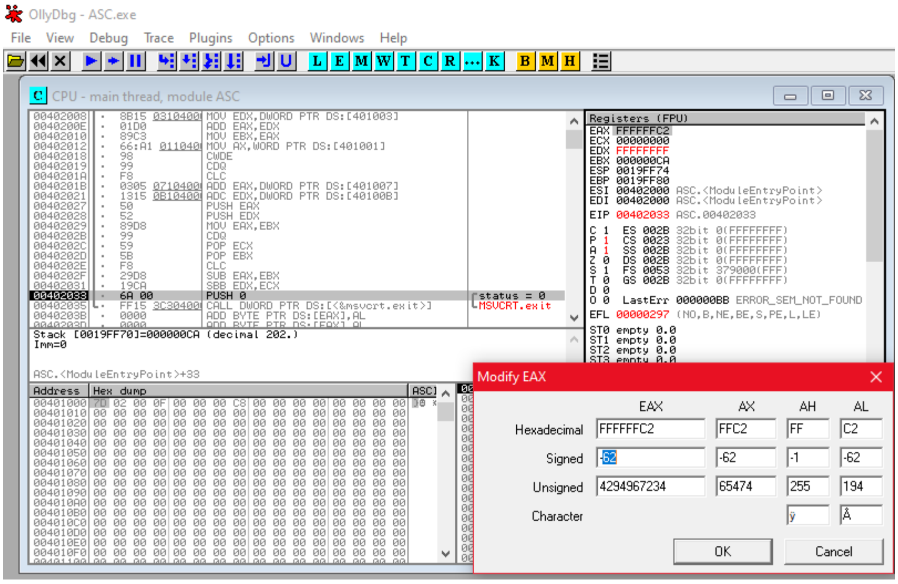
Pop Mihai-Daniel, Grupa 215/2

Adunari, scaderi – interpretare cu semn

a - byte, b - word, c - double word, d - qword

1. Ex. 27: (a+c)-(d+b)

```
; a - byte, b - word, c - double word, d - qword
; Ex. 27: (a+c)-(d+b)
; ex.1 : a=125, b=2, c=15, d=200; Rezultat: (125+15)-(2+200) = 140-202 = -62
bits 32
global  start

extern  exit ; indicam asamblorului ca exit exista, chiar daca noi nu o vom defini
import  exit msvcrt.dll; exit este o functie care incheie procesul, este definita in msvcrt.dll
        ; msvcrt.dll contine exit, printf si toate celelalte functii C-runtime importante
segment  data use32 class=data ; segmentul de date in care se vor defini variabilele
                a db 125
                b dw 2
                c dd 15
                d dq 200
segment  code use32 class=code ; segmentul de cod
start:
;pentru a calcula a+c, convertim a de la byte la doubleword pentru a-l putea aduna la doubleword-ul c
mov al, [a] ;al = a = 125
cbw ;conversie cu semn de la al la ax
cwde ;conversie cu semn de la ax la eax
;eax = a = 125
mov edx,[c] ;edx = c = 15
add eax,edx ;adunare eax cu edx
;eax = eax + edx = 15+125 = 140
mov ebx,eax ;ebx = eax = 140

mov ax, [b] ;
cwde ;conversie cu semn de la ax la eax
cdq ;conversie cu semn de la eax la edx:eax
;edx,eax = b = 2

clc ;Carry Flag = 0
add eax, dword [d]
adc edx, dword [d+4] ;edx:eax = d + b = 2+200 = 202

push eax
push edx
;am pus in stiva valoarea rezultatului (d+b)
mov eax, ebx ;
cdq ;conversie cu semn de la eax la edx:eax
;edx:eax = 140
pop ecx
pop ebx
;am scos din stiva valoarea rezultatului (d+b) = 202

clc
sub eax,ebx
sbb edx,ecx
;(a+c)-(d+b) = 140-202 = -62
push dword 0 ;se pune pe stiva codul de retur al functiei exit
call [exit] ;apelul functiei sistem exit pentru terminarea executiei programului
```



```
00402008  .  8B15 03104000  MOV EDX,DWORD PTR DS:[401003]
0040200E  .  01D0           ADD EAX,EDX
00402010  .  89C3           MOV EBX,EAX
00402012  .  66:A1 01104000 MOV AX,WORD PTR DS:[401001]
00402018  .  98             CWDE
00402019  .  99             CDQ
0040201A  .  F8             CLC
0040201B  .  0305 07104000  ADD EAX,DWORD PTR DS:[401007]
00402021  .  1315 0B104000  ADC EDX,DWORD PTR DS:[40100B]
00402027  .  50             PUSH EAX
00402028  .  52             PUSH EDX
00402029  .  89D8           MOV EAX,EBX
0040202B  .  99             CDQ
0040202C  .  59             POP ECX
0040202D  .  5B             POP EBX
0040202E  .  F8             CLC
0040202F  .  29D8           SUB EAX,EBX
00402031  .  19CA           SBB EDX,ECX
00402033  .  6A 00          PUSH 0                                   [status = 0
00402035  L. FF15 3C304000  CALL DWORD PTR DS:[<&msvcrt.exit>]       LMSVCRT.exit
0040203B  .  0000           ADD BYTE PTR DS:[EAX],AL
0040203D  .  0000           ADD BYTE PTR DS:[EAX],AL
Stack [0019FF70]=000000CA (decimal 202.)
Imm=0

ASC.<ModuleEntryPoint>+33
```

```
Registers (FPU)
EAX FFFFFFC2
ECX 00000000
EDX FFFFFFFF
EBX 000000CA
ESP 0019FF74
EBP 0019FF80
ESI 00402000 ASC.<ModuleEntryPoint>
EDI 00402000 ASC.<ModuleEntryPoint>

EIP 00402033 ASC.00402033

C 1  ES 002B 32bit 0(FFFFFFFF)
P 1  CS 0023 32bit 0(FFFFFFFF)
A 1  SS 002B 32bit 0(FFFFFFFF)
Z 0  DS 002B 32bit 0(FFFFFFFF)
S 1  FS 0053 32bit 379000(FFF)
T 0  GS 002B 32bit 0(FFFFFFFF)
D 0
O 0  LastErr 000000BB ERROR_SEM_NOT_FOUND

EFL 00000297 (NO,B,NE,BE,S,PE,L,LE)

ST0 empty 0.0
ST1 empty 0.0
ST2 empty 0.0
ST3 empty 0.0
```

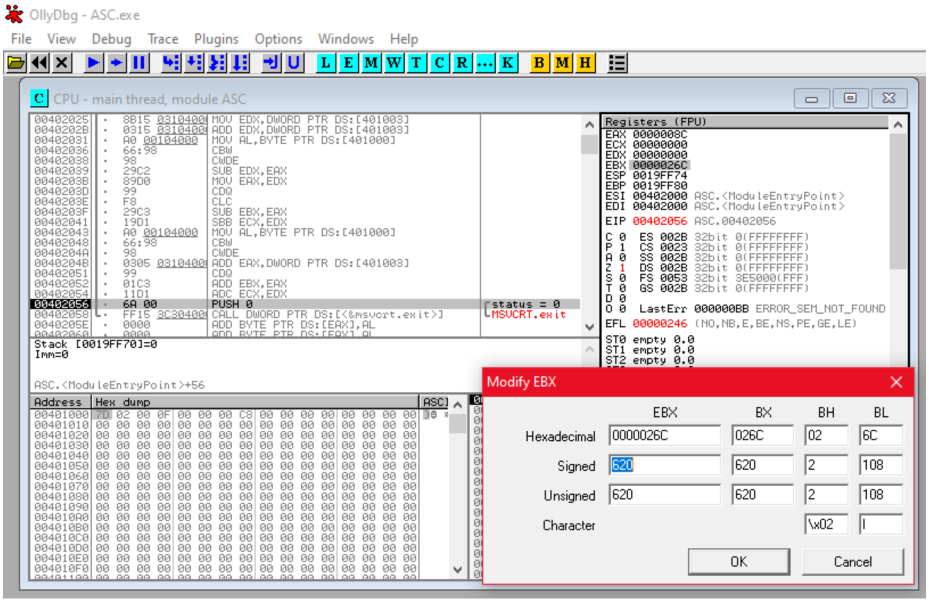
```
Address   Hex dump
00401000  7D 02 00 0F 00 00 00 C8 00 00 00 00 00 00 00 00
00401010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401080  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Modify EAX

	EAX	AX	AH	AL
Hexadecimal	FFFFFFC2	FFC2	FF	C2
Signed	-62	-62	-1	-62
Unsigned	4294967234	65474	255	194
Character			ÿ	Â

2. Ex. 27: (d+d-c)-(c+c-a)+(c+a)

```
; Adunari si scaderi in interpretarea cu semn
; a - byte, b - word, c - double word, d - qword
; Ex. 27: (d+d-c)-(c+c-a)+(c+a)
; ex.1 : a=125, b=2, c=15, d=200;
; Rezultat: (200+200-15)-(15+15-125)+(15+125) = 385-(-95)+140 = 620
bits 32
global  start
extern  exit
import  exit msvcrt.dll

segment  data use32 class=data
                a db 125
                b dw 2
                c dd 15
                d dq 200

segment  code use32 class=code ; segmentul de cod
start:

   mov eax, dword [d]
   mov edx, dword [d+4] ;edx:eax = d = 200
   add eax, dword [d]
   adc edx, dword [d+4] ;edx:eax = d+d = 200+200 = 400

   mov ebx, eax
   mov ecx, edx ;mutam rezultatul calculului d+d in perechea de registrii ecx:ebx
   mov eax, [c]
   cdq ;convertim variabila c din dword in qword
   sub ebx, eax
   sbb ecx, edx ;ecx:ebx = d+d-c = 400-15 = 385

   mov edx, [c] ;edx = c = 15
   add edx, [c] ;edx = edx+c = c+c = 15+15 = 30
   mov al, [a]
   cbw
   cwde ;eax = a = 125
   sub edx,eax ;ebx = edx-eax = c+c-a = 30-125 = -95
   mov eax, edx ;eax = edx = -95
   cdq ;convertim rezultatul din a doua paranteza din dword in qword ;edx:eax = -95
   clc
   sub ebx, eax
   sbb ecx, edx ;ecx:ebx = (d+d-c)-(c+c-a) = 385-(-95) = 385+95 = 480

   mov al, [a]
   cbw
   cwde ;eax = a = 125
   add eax, [c] ;eax = eax+c = (a+c) = 125+15 = 140
   cdq
   add ebx, eax
   adc ecx, edx ;ecx:ebx = (d+d-c)-(c+c-a)+(a+c) = 480+140 = 620

push dword 0 ;se pune pe stiva codul de retur al functiei exit
call [exit] ;apelul functiei sistem exit pentru terminarea executiei programului
```


OllyDbg - ASC.exe
File View Debug Trace Plugins Options Windows Help
L E M W T C R ... K B M H
CPU - main thread, module ASC

```
00402025  .  8B15 03104000 MOV EDX,DWORD PTR DS:[401003]
0040202B  .  0315 03104000 ADD EDX,DWORD PTR DS:[401003]
00402031  .  A0 00104000   MOV AL,BYTE PTR DS:[401000]
00402036  .  66:98         CBW
00402038  .  98            CWDE
00402039  .  29C2          SUB EDX,EAX
0040203B  .  89D0          MOV EAX,EDX
0040203D  .  99            CDQ
0040203E  .  F8            CLC
0040203F  .  29C3          SUB EBX,EAX
00402041  .  19D1          SBB ECX,EDX
00402043  .  A0 00104000   MOV AL,BYTE PTR DS:[401000]
00402048  .  66:98         CBW
0040204A  .  98            CWDE
0040204B  .  0305 03104000 ADD EAX,DWORD PTR DS:[401003]
00402051  .  99            CDQ
00402052  .  01C3          ADD EBX,EAX
00402054  .  11D1          ADC ECX,EDX
00402056  .  6A 00         PUSH 0                                  [status = 0
00402058  L. FF15 3C304000 CALL DWORD PTR DS:[<&msvcrt.exit>]      LMSVCRT.exit
0040205E  .  0000          ADD BYTE PTR DS:[EAX],AL
00402060  .  0000          ADD BYTE PTR DS:[EAX],AL
Stack [0019FF70]=0
Imm=0

ASC.<ModuleEntryPoint>+56

Address   Hex dump
00401000  7D 02 00 0F 00 00 00 C8 00 00 00 00 00 00 00 00
00401010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401080  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Registers (FPU)
EAX 0000008C
ECX 00000000
EDX 00000000
EBX 0000026C
ESP 0019FF74
EBP 0019FF80
ESI 00402000 ASC.<ModuleEntryPoint>
EDI 00402000 ASC.<ModuleEntryPoint>

EIP 00402056 ASC.00402056

C 0  ES 002B 32bit 0(FFFFFFFF)
P 1  CS 0023 32bit 0(FFFFFFFF)
A 0  SS 002B 32bit 0(FFFFFFFF)
Z 1  DS 002B 32bit 0(FFFFFFFF)
S 0  FS 0053 32bit 3E5000(FFF)
T 0  GS 002B 32bit 0(FFFFFFFF)
D 0
O 0  LastErr 000000BB ERROR_SEM_NOT_FOUND

EFL 00000246 (NO,NB,E,BE,NS,PE,GE,LE)

ST0 empty 0.0
ST1 empty 0.0
ST2 empty 0.0
```

Modify EBX

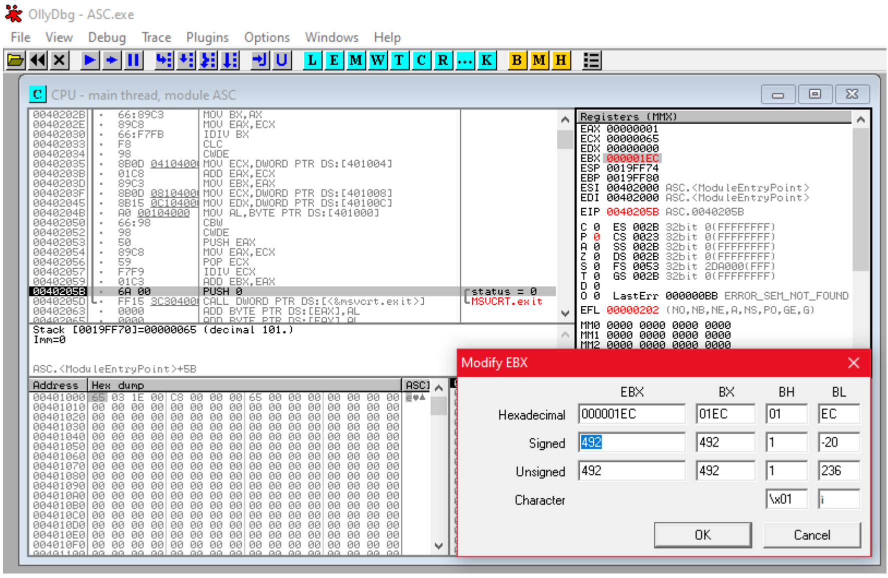
	EBX	BX	BH	BL
Hexadecimal	0000026C	026C	02	6C
Signed	620	620	2	108
Unsigned	620	620	2	108
Character			\x02	l

Inmultiri, impartiri – interpretare cu semn

a,b - byte, c - word, e - double word, x – qword

3. Ex. 27: (100+a+b*c)/(a-100)+e+x/a

```
; Inmultiri si impartiri in interpretarea cu semn
; a,b - byte; c - word; e - doubleword; x - qword
; Ex. 27: (100+a+b*c)/(a-100)+e+x/a
; ex.1 : a=101, b=3, c=30, e=200, x=101;
; Rezultat: (100+101+3*30)/(101-100)+200+101/101 = 291/1+200+1 = 492
bits 32
global  start
extern  exit
import  exit msvcrt.dll
segment  data use32 class=data
              a db 101
   b db 3
   c dw 30
   e dd 200
   x dq 101
segment  code use32 class=code ; segmentul de cod
start:
   mov al, [a]
   cbw
   cwde ;eax = a = 101
   add eax, 100 ;eax = eax+100 = 201
   mov ecx, eax ;ecx = eax = 201
   mov al, [b]
   cbw ;ax = b = 3
   mov bx, [c] ;bx = c = 30
   imul bx ;eax = ax*bx = b*c = 3*30 = 90
   add ecx, eax ;ecx = ecx+eax = (100+a+b*c) = 201+90 = 291

   mov al, [a]
   cbw ;ax = a = 101
   sub ax, 100 ;ax = ax-100 = a-100 = 101-100 = 1
   mov bx, ax ;bx = ax = 1
   mov eax, ecx ;eax = ecx = 291
   idiv bx ;ax = eax/bx = (100+a+b*c)/(a-100) = 291/1 = 291

   clc
   cwde ;ax = eax = 291
   mov ecx, [e] ;ecx = e = 200
   add eax, ecx ;eax = eax+ecx = 291+200 = 491
   mov ebx, eax ;ebx = 204

   mov ecx, dword [x]
   mov edx, dword [x+4] ;edx:ecx = x = 101
   mov al, [a]
   cbw
   cwde ;eax = a = 101
   push eax
   mov eax, ecx ;eax = ecx
   pop ecx ;ecx = a = 101
   idiv ecx ;eax = edx:eax/ecx = x/a = 101/101 = 1
   add ebx, eax ;ebx = ebx+eax = (100+a+b*c)/(a-100)+e+x/a = 491+1 = 492
push dword 0
call [exit]
```


OllyDbg - ASC.exe
File View Debug Trace Plugins Options Windows Help
L E M W T C R ... K B M H
CPU - main thread, module ASC

```
0040202B  .  66:89C3        MOV BX,AX
0040202E  .  89C8           MOV EAX,ECX
00402030  .  66:F7FB        IDIV BX
00402033  .  F8             CLC
00402034  .  98             CWDE
00402035  .  8B0D 0410400   MOV ECX,DWORD PTR DS:[401004]
0040203B  .  01C8           ADD EAX,ECX
0040203D  .  89C3           MOV EBX,EAX
0040203F  .  8B0D 0810400   MOV ECX,DWORD PTR DS:[401008]
00402045  .  8B15 0C10400   MOV EDX,DWORD PTR DS:[40100C]
0040204B  .  A0 00104000    MOV AL,BYTE PTR DS:[401000]
00402050  .  66:98          CBW
00402052  .  98             CWDE
00402053  .  50             PUSH EAX
00402054  .  89C8           MOV EAX,ECX
00402056  .  59             POP ECX
00402057  .  F7F9           IDIV ECX
00402059  .  01C3           ADD EBX,EAX
0040205B  .  6A 00          PUSH 0                                   [status = 0
0040205D  L. FF15 3C30400   CALL DWORD PTR DS:[<&msvcrt.exit>]       LMSVCRT.exit
00402063  .  0000           ADD BYTE PTR DS:[EAX],AL
Stack [0019FF70]=00000065 (decimal 101.)
Imm=0

ASC.<ModuleEntryPoint>+5B
```

```
Registers (MMX)
EAX 00000001
ECX 00000065
EDX 00000000
EBX 000001EC
ESP 0019FF74
EBP 0019FF80
ESI 00402000 ASC.<ModuleEntryPoint>
EDI 00402000 ASC.<ModuleEntryPoint>

EIP 0040205B ASC.0040205B

C 0  ES 002B 32bit 0(FFFFFFFF)
P 0  CS 0023 32bit 0(FFFFFFFF)
A 0  SS 002B 32bit 0(FFFFFFFF)
Z 0  DS 002B 32bit 0(FFFFFFFF)
S 0  FS 0053 32bit 2DA000(FFF)
T 0  GS 002B 32bit 0(FFFFFFFF)
D 0
O 0  LastErr 000000BB ERROR_SEM_NOT_FOUND
EFL 00000202 (NO,NB,NE,A,NS,PO,GE,G)

MM0 0000 0000 0000 0000
MM1 0000 0000 0000 0000
MM2 0000 0000 0000 0000
```

```
Address   Hex dump                                         ASCI
00401000  65 03 1E 00 C8 00 00 00 65 00 00 00 00 00 00 00  e♥▲
00401010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401080  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

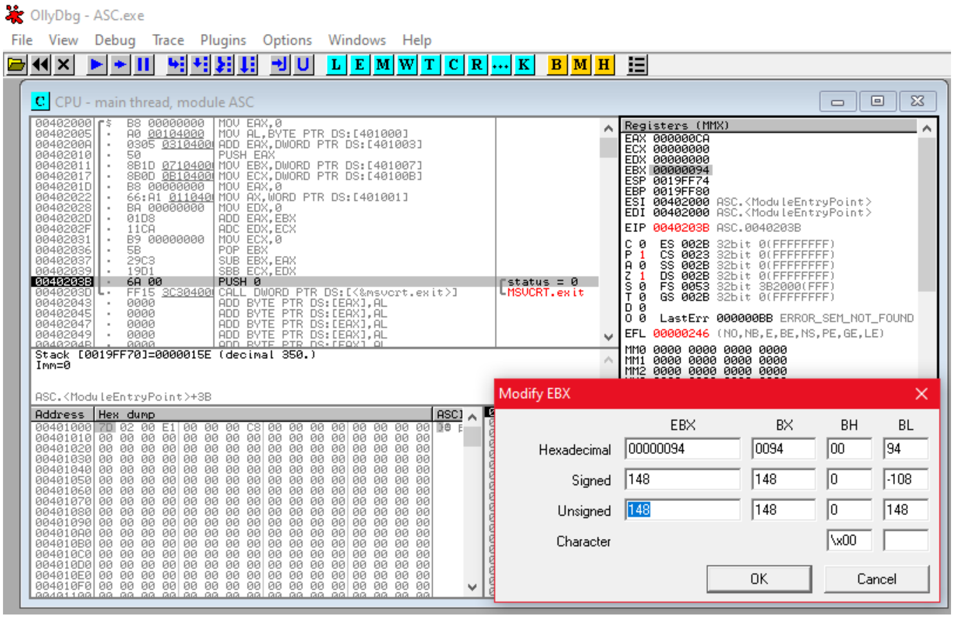
Modify EBX

	EBX	BX	BH	BL
Hexadecimal	000001EC	01EC	01	EC
Signed	492	492	1	-20
Unsigned	492	492	1	236
Character			\x01	i

a - byte, b - word, c - double word, d - qword

4. Ex. 27: (a+c)-(d+b) interpretare fara semn

```
; Adunari si scaderi in interpretare fara semn
; ; a - byte, b - word, c - double word, d - qword
; Ex. 27: (a+c)-(d+b)
; ex.1 : a=125, b=2, c=225, d=200; Rezultat:(125+225)-(200+2) = 350-202 = 148

bits 32
global  start
extern  exit
import  exit msvcrt.dll

segment  data use32 class=data
            a db 125
            b dw 2
            c dd 225
            d dq 200


segment  code use32 class=code ; segmentul de cod
start:
    mov eax, 0 ;eax = 0
    mov al, [a] ;eax = a = 125
    add eax, [c] ;eax = eax+c = (a+c) = 125+225 = 350
    push eax ;punem in stiva rezultatul, eliberand registrul

    mov ebx, dword [d]
    mov ecx, dword [d+4] ;ecx:ebx = d = 200
    mov eax, 0 ;eax = 0
    mov ax, [b] ;eax = b = 2
    mov edx, 0 ;edx = 0
    ;edx:eax = b = 2
    add eax, ebx
    adc edx, ecx
    ;edx:eax = d+b = 200+2 = 202

    mov ecx, 0 ;ecx = 0
    pop ebx ;ebx = 350
    sub ebx, eax
    sbb ecx, edx
    ;ecx:ebx = (a+c)-(d+b) = 350-202 = 148

            push dword 0
            call [exit]
```



OllyDbg - ASC.exe
File View Debug Trace Plugins Options Windows Help
L E M W T C R ... K B M H
CPU - main thread, module ASC

```
00402000  $  B8 00000000   MOV EAX,0
00402005  .  A0 00104000   MOV AL,BYTE PTR DS:[401000]
0040200A  .  0305 0310400  ADD EAX,DWORD PTR DS:[401003]
00402010  .  50            PUSH EAX
00402011  .  8B1D 0710400  MOV EBX,DWORD PTR DS:[401007]
00402017  .  8B0D 0B10400  MOV ECX,DWORD PTR DS:[40100B]
0040201D  .  B8 00000000   MOV EAX,0
00402022  .  66:A1 011040  MOV AX,WORD PTR DS:[401001]
00402028  .  BA 00000000   MOV EDX,0
0040202D  .  01D8          ADD EAX,EBX
0040202F  .  11CA          ADC EDX,ECX
00402031  .  B9 00000000   MOV ECX,0
00402036  .  5B            POP EBX
00402037  .  29C3          SUB EBX,EAX
00402039  .  19D1          SBB ECX,EDX
0040203B  .  6A 00         PUSH 0                                  status = 0
0040203D  .  FF15 3C30400  CALL DWORD PTR DS:[<&msvcrt.exit>]      MSVCRT.exit
00402043  .  0000          ADD BYTE PTR DS:[EAX],AL
00402045  .  0000          ADD BYTE PTR DS:[EAX],AL
00402047  .  0000          ADD BYTE PTR DS:[EAX],AL
00402049  .  0000          ADD BYTE PTR DS:[EAX],AL
Stack [0019FF70]=0000015E (decimal 350.)
Imm=0

ASC.<ModuleEntryPoint>+3B
```

```
Registers (MMX)
EAX 000000CA
ECX 00000000
EDX 00000000
EBX 00000094
ESP 0019FF74
EBP 0019FF80
ESI 00402000 ASC.<ModuleEntryPoint>
EDI 00402000 ASC.<ModuleEntryPoint>

EIP 0040203B ASC.0040203B

C 0  ES 002B 32bit 0(FFFFFFFF)
P 1  CS 0023 32bit 0(FFFFFFFF)
A 0  SS 002B 32bit 0(FFFFFFFF)
Z 1  DS 002B 32bit 0(FFFFFFFF)
S 0  FS 0053 32bit 3B2000(FFF)
T 0  GS 002B 32bit 0(FFFFFFFF)
D 0
O 0  LastErr 000000BB ERROR_SEM_NOT_FOUND

EFL 00000246 (NO,NB,E,BE,NS,PE,GE,LE)

MM0 0000 0000 0000 0000
MM1 0000 0000 0000 0000
MM2 0000 0000 0000 0000
```

```
Address  Hex dump
00401000 7D 02 00 E1 00 00 00 C8 00 00 00 00 00 00 00 00
00401010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401040 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401080 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00401090 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010A0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010D0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004010F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Modify EBX

	EBX	BX	BH	BL
Hexadecimal	00000094	0094	00	94
Signed	148	148	0	-108
Unsigned	148	148	0	148
Character			\x00	