

a,b,c,d – byte

Pb.13 $a+b-c+d-(a-d)$

```
bits 32 ; assembling for the 32 bits architecture

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

;a+b-c+d-(a-d)
; 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 db 2
    b db 3
    c db 5
    d db 7
;a+b-c+d-(a-d)=2+3-5+7-(2-7)=5-5+7+5=12

; our code starts here
segment code use32 class=code
    start:
        mov AL,byte[a] ;AL=a                2
        add AL,byte[b] ;AL=a+b              5
        sub AL,byte[c] ;AL=a+b-c            0
        add AL,byte[d] ;AL=a+b-c+d          7
        mov AH,byte[a] ;AH=a                2
        sub AH,byte[d] ;AH=a-d              -5
        sub AL,AH      ;AL=a+b-c+d-(a-d)    7+5=12
        push dword 0   ; push the parameter for exit onto the stack
        call [exit]    ; call exit to terminate the program
```


Pb.18 $d - (a+b) + c$

```
bits 32 ; assembling for the 32 bits architecture

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

;d-(a+b)+c
; 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 db 2
    b db 3
    c db 5
    d db 7
;d-(a+b)+c=7-(2+3)+5=7-5+5=7

; our code starts here
segment code use32 class=code
    start:
        mov AL,byte[d]; AL=d            7
        mov AH,byte[a]; AH=a            2
        add AH,byte[b]; AH=a+b          5
        sub AL,AH; AL=d-(a+b)           2
        add AL,byte[c]; AL=d-(a+b)+c    7
        push dword 0 ; push the parameter for exit onto the stack
        call [exit] ; call exit to terminate the program
```


a,b,c,d – word

Pb.13 $(a+a-c) - (b+b+d)$

```
bits 32 ; assembling for the 32 bits architecture

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

; (a+a-c)-(b+b+d)
; 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 1022
    b dw 312
    c dw 512
    d dw 260

; (a+a-c)-(b+b+d)=(1022+1022-512)-(312+312+260)=648

; our code starts here
segment code use32 class=code
    start:
        mov AX,word[a] ;AX=a 1022
        add AX,word[a] ;AX=a+a 2044
        sub AX,word[c] ;AX=a+a-c 1532
        mov BX,word[b] ;BX=b 312
        add BX,word[b] ;BX=b+b 624
        add BX,word[d] ;BX=b+b+d 884
        sub AX,BX ;AX=(a+a-c)-(b+b+d) 648
        push dword 0 ; push the parameter for exit onto the stack
        call [exit] ; call exit to terminate the program
```


Pb18 $(a-b-c)+(a-c-d-d)$

```
bits 32 ; assembling for the 32 bits architecture

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

; (a-b-c)+(a-c-d-d)
; 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 1022
    b dw 312
    c dw 512
    d dw 260

; (a-b-c)+(a-c-d-d)=(1022-312-512)+(1022-512-260-260)=188
; our code starts here
segment code use32 class=code
    start:
        mov AX,word[a]; AX=a                1022
        sub AX,word[b]; AX=a-b              710
        sub AX,word[c]; AX=a-b-c            198
        mov BX,word[a]; BX=a                1022
        sub BX,word[c]; BX=a-c              510
        sub BX,word[d]; BX=a-c-d            250
        sub BX,word[d]; BX=a-c-d-d          -10
        add AX,BX; AX=(a-b-c)+(a-c-d-d) 188
        push dword 0 ; push the parameter for exit onto the stack
        call [exit] ; call exit to terminate the program
```


Inmultiri,impartiri

a,b,c-byte d-word

Pb13 $[(a*b)-d]/(b+c)$

```
bits 32 ; assembling for the 32 bits architecture

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

; [(a*b)-d]/(b+c)
; 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 db 20
    b db 30
    c db 5
    d dw 250
; [(a*b)-d]/(b+c)=(600-250)/(35)=350/35=10 r 0

; our code starts here
segment code use32 class=code
    start:
        mov AL,byte[a]; AL=a 20
        mul byte[b] ; AX=AL*b=(a*b) 600
        sub AX,word[d]; AX=AX-d=(a*b)-d 350
        mov BL,byte[b]; BL=b 30
        add BL,byte[c]; BL=b+c 35
        div BL ; AL=AX/BL=((a*b)-d)/(b+c) AH=AX%BL 10 r 0
        push dword 0 ; push the parameter for exit onto the stack
        call [exit] ; call exit to terminate the program
```


Pb.18 200-[3*(c+b-d/a)-300]

```
bits 32 ; assembling for the 32 bits architecture

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

;200-[3*(c+b-d/a)-300]
; 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 db 20
    b db 30
    c db 5
    d dw 250
;200-[3*(c+b-d/a)-300]=200-[3*(5+30-250/20)-300]=200-[3*(35-12)-300]=431

; our code starts here
segment code use32 class=code
    start:
        mov AX,word[d]; AX=d                250
        div byte[a] ; AL=d/a                12
        mov BL,AL ; BL=AL
        mov AL,byte[c]; AL=c                5
        add AL,byte[b]; AL=c+b              35
        sub AL,BL ; AL=c+b-d/a              23
        mov BL,3 ; BL=3
        mul BL ; AX=3*(c+b-d/a)              69
        mov BX,300 ; BX=300
        sub AX,BX ; AX=3*(c+b-d/a)-300      -231
        mov BX,200 ; BX=200
        sub BX,AX ; BX=200-3*(c+b-d/a)-300  431=1AF
        push dword 0 ; push the parameter for exit onto the stack
        call [exit] ; call exit to terminate the program
```


a,b,c,d-byte e,f,g,h-word

Pb.13 $(g+5) - a*d$

```
bits 32 ; assembling for the 32 bits architecture

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

; (g+5)-a*d
; 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 db 12
    d db 23
    g dw 276
; (g+5)-a*d=276+5-12*23=5=5h

; our code starts here
segment code use32 class=code
    start:
        mov AL,byte[a]; AL=a            12
        mul byte[d] ; AX=a*d            276
        mov BX,5 ; BX=5
        add BX,word[g]; BX=g+5          281
        sub BX,AX ; BX=(g+5)-a*d        5=5h
        push dword 0 ; push the parameter for exit onto the stack
        call [exit] ; call exit to terminate the program
```


Pb.18 $f+(c-2)*(3+a)/(d-4)$

```
bits 32 ; assembling for the 32 bits architecture

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

;f+(c-2)*(3+a)/(d-4)
; 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 db 5
    c db 12
    d db 6
    f dw 508
;f+(c-2)*(3+a)/(d-4)=508+10*8/2=508+40=548=224h

; our code starts here
segment code use32 class=code
    start:
        mov AL,byte[c]; AL=c 12
        sub AL,2 ; AL=c-2 10
        mov BL,3 ; BL=3
        add BL,byte[a]; BL=3+a 8
        mul BL ; AX=(c-2)*(3+a) 80
        mov BL,byte[d]; BL=d 6
        sub BL,4 ; BL=d-4 2
        div BL ; AL=(c-2)*(3+a)/(d-4) 40
        mov AH,0 ; AH=0
        add AX,word[f]; AX=f+(c-2)*(3+a)/(d-4) 548=224h
        push dword 0 ; push the parameter for exit onto the stack
        call [exit] ; call exit to terminate the program
```

```

1  bits 32 ; assembling for the 32 bits architecture
2
3  ; declare the EntryPoint (a label defining the very first instruction of the program)
4  global start
5
6  ; f+(c-2)*(3+a)/(d-4)
7  ; declare external functions needed by our program
8  extern exit ; tell nasm that exit exists even if we won't be defining it
9  import exit msvcrt.dll ; exit is a function that ends the calling process. It is defined in msvcrt.dll
10 ; msvcrt.dll contains exit, printf and all the other important C-runtime specific functions
11
12 ; our data is declared here (the variables needed by our program)
13 segment data use32 class=data
14     a db 5
15     c db 12
16     d db 6
17     f dw 508
18 ; f+(c-2)*(3+a)/(d-4)=508+10*8/2=508+40=548=224h
19
20 ; our code starts here
21 segment code use32 class=code
22     start:
23         mov AL,byte[c]; AL=c 12
24         sub AL,2 ; AL=c-2 10
25         mov BL,3 ; BL=3
26         add BL,byte[a]; BL=3+a 8
27         mul BL ; AX=(c-2)*(3+a) 80
28         mov BL,byte[d]; BL=d 6
29         sub BL,4 ; BL=d-4 2
30         div BL ; AL=(c-2)*(3+a)/(d-4) 40
31         mov AH,0 ; AH=0
32         add AX,word[f]; AX=f+(c-2)*(3+a)/(d-4) 548=224h
33         push dword 0 ; push the parameter for exit onto the stack
34         call [exit] ; call exit to terminate the program
35

```