**Exerciții tutoriat 2-3**

**Vor fi 2 mini exercitii, lab3.1 test si cateva exercitii in plus facute de noi.**

Recomand daca nu intelegeti, sa rulati programul si sa faceti debugg si daca nu intelegeti nici atunci, sa va auitati pe rezolvarile trimise de noi.

Ce este important pentru test lab3: sa stiti bine jumpurile, operatiile artimetice, cum functioneaza hexa si complementul fata de 2, shiftarile, cum se afiseaza ceva, cum merg registrii (mai ales ah, al, bh, bl, etc..), mare grija la overflow, tipurile de date si sa va ma uitati un pic pe materie la lab (flaguri, ce inseamna „$” si chestii la care ar putea sa va ceara definitie). Multa baftaaa!

**Mini exercitiu :**

Ce valoare va avea eax dupa a doua operatie de mai jos?

.data

.text

.globl main

main:

    movl $1, %eax

    not %eax

    movl $1, %eax

    movl $0, %ebx

    int $0x80

**Mini exercitiu 2:**

Ce valoare va avea eax dupa a doua operatie de mai jos?

.data

.text

.globl main

main:

    movl $0x7fffffff, %eax

    addl $1, %eax

    movl $1, %eax

    movl $0, %ebx

    int $0x80

Mai jos o sa urmeze lab3.1 test, fata de acel file, aici puteti copia codul si nu aveti raspunsurile date.

**Ex 1:**

Fie codul de mai jos. Care este valorea lui s cand executia ajunge la et\_exit?

.data

s: .long 0

.text

.globl main

main:

    mov $1, %edx

    mov $0, %eax

    movl $0xffffffff, %ebx

    divl %ebx

    mov %eax, %ecx

    movl %eax, s

    movl $1, %eax

    movl $0, %ebx

    int $0x80

1. 1
2. 0
3. 0xffffffff
4. 15

**Ex 2:**

Se stocheaza in registrul eax valoarea 0x40000000, in ebx 0x8, in ecx 0x1 si in edx 0x8.

Ce valori vor avea registrii eax si edx dupa executarea instructiunii mul %edx?

1. Eax = 0, edx = 2
2. Eax = 32, edx = 0
3. Eax = 32, edx = 2
4. Eax = 2, edx = 0

**Ex 3:**

Fie codul de mai jos. Care este valoarea depozitata la final in ecx (in dreptul etichetei exit) ?

.data

x: .long 0x80000000

y: .long 0x70000000

.text

.globl main

main:

mov x, %eax

cmp y, %eax

jge label

mov $5, %ecx

jmp exit

label:

mov $6, %ecx

exit:

mov $1, %eax

mov $0, %ebx

int $0x80

1. 5
2. 6

Ex 4: (original in lab3.1 penultima poza, aici este schimbat)

Fie urmatorul program. Ce valoare vom obtine daca vom rula cu debuggerul

urmatoarele comenzi?

b et\_exit

run

i r ebx i r edx

.data

.text

.globl main

main:

mov $11, %eax

shr $1, %eax

mov $3, %ebx

mul %ebx

mov $0, %edx

xor $7, %eax

mov $8, %ebx

div %ebx

sub %eax, %ebx

et\_exit:

mov $1, %eax

mov $0, %ebx

int $0x80

**Ex 5 :** Care este ordinea de trecere prin etichete?

.data

.text

.global main

main:

mov $1, %eax

mov $2, %ebx

mov $3, %ecx

mov $4, %edx

cmp %ebx, %eax

je etx

ety:

    cmp %ecx, %edx

    jg etz

    jmp ett

etx:

    jmp ety

etz:

    mov %ebx, %edx

    jmp ety

ett:

    mov $1, %eax

    mov $0, %ebx

    int $0x80

**Optional:** (se va duce pe ramura ceva sau continua? De ce?)   
 [aici gasiti toate jumpurile](https://faydoc.tripod.com/cpu/jnp.htm)

.data

.text

.global main

main:

    movl $0x2, %eax

    test $1, %eax

    jp ceva

    movl $1, %eax

    movl $0, %ebx

    int $0x80

ceva:

    movl $1, %eax

    movl $0, %ebx

    int $0x80

**Optional 2:** (se va duce pe ramura ceva sau continua? De ce?)

.data

.text

.globl main

main:

    movl $0xfffffff2, %eax

    test %eax, %eax

    js ceva

    movl $1, %eax

    movl $0, %ebx

    int $0x80

ceva:

    movl $1, %eax

    movl $0, %ebx

    int $0x80

**Exercitiu greut:** Aflati daca un numar este prim.