

La ce adresa se face salt in urma executiei instructiunii **ret** stiind ca stiva arata ca in figura si ca stiva creste in sus?

* 2 puncte

```
0000| 0xffffd048 --> 0x565561b1 (<main+10>: add eax,DWORD PTR ds:0x56559008)
0004| 0xffffd04c --> 0xf7deae5 (<__libc_start_main+245>: add esp,0x10)
0008| 0xffffd050 --> 0x1
0012| 0xffffd054 --> 0xffffd0e4 --> 0xffffd2af ("/home/ruxi/Desktop/Lab5/k1")
0016| 0xffffd058 --> 0xffffd0ec --> 0xffffd2ca ("SHELL=/bin/bash")
0020| 0xffffd05c --> 0xffffd074 --> 0x0
0024| 0xffffd060 --> 0xf7fb3000 --> 0x1e6d6c
0028| 0xffffd064 --> 0xf7ffd000 --> 0x2bf24
```

- ☐ 0xffffd048
- ☒ 0x565561b1
- ☐ 0xf7deae5
- ☐ 0xf7ffd000

Fie procedura **proc** in care declaram o variabila locala **long v[100]**. Cum va arata declaratia pe stiva?

* Un punct

- ☒ subl \$400,%esp
- ☐ addl \$100,%esp
- ☐ pushl \$v
- ☐ pushl v

Fie urmatoarele variabile globale:

*

Un punct

.data

v: .space 4

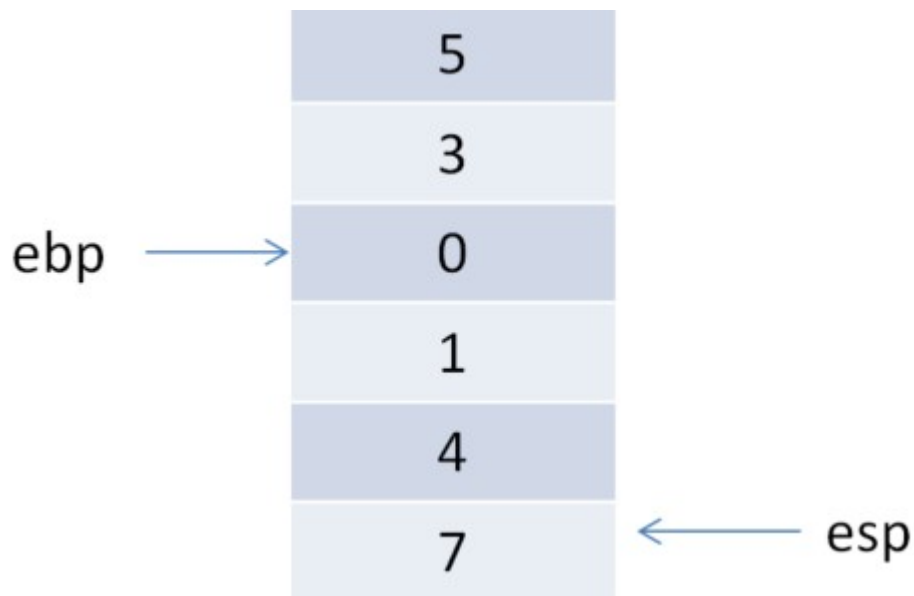
x: .long 1,2,3,4,5

Ce incarcari pe stiva va produce apelul **proc(x[0], &v)**?

- ☒ pushl \$v; pushl x;
- ☐ pushl \$v; pushl \$x;
- ☐ pushl x; pushl \$v;
- ☐ pushl \$x; pushl \$v;

Cum se realizeaza accesarea lui 3 raportat la **%ebp**? Dar a lui 4? *

2 puncte



- ☐ 0(%ebp), 8(%ebp)
- ☒ 4(%ebp), -8(%ebp)
- ☐ -4(%ebp), 8(%ebp)
- ☐ 16(%ebp), 4(%ebp)

Se da codul de mai jos. Raspundeti la urmatoarele intrebari bazate pe acesta.

.data

n: .long 3

v: .long 5, 9, 10

s: .space 4

.text

medie:

movl 8(%ebp), %esi

addl 12(%ebp), %esi

movl %esi, %eax

movl \$0, %edx

movl \$2, %ecx

divl %ecx

ret

.global main**main:**

```
mov $0, %edx
mov $1, %ecx
lea v, %esi
```

```
subl $1, n
```

loop:

```
cmp n, %ecx
jg exit
movl (%esi, %ecx, 4), %eax
movl -4(%esi, %ecx, 4), %ebx
```

```
push %eax
push %ebx
call medie
pop %eax
pop %eax
```

```
add %eax, %edx
```

```
inc %ecx
jmp loop
```

exit:

```
mov %edx, s
```

finish:

```
mov $1, %eax
xor %ebx, %ebx
int $0x80
```

In general, accesările 8(%ebp), respectiv 12(%ebp) indica primul si al doilea argument al unei proceduri. In cazul acesta accesările argumentelor vor produce **Segmentation Fault**. De ce? * Un punct

- ☐ Parametrii nu au fost corect incarcati pe stiva
- ☐ Stiva nu a fost curatata
- ☐ Accesarea elementelor din vector nu este realizata corect
- ☒ Nu s-a realizat mutarea lui %ebp in stack frame-ul curent

Ce **regiștri callee-saved trebuie** salvati pe stiva? Alegeti una sau mai multe variante de raspuns. * Un punct

- ☐ %ebx
- ☒ %esi
- ☐ %edi
- ☒ %ebp

Ce **regiștri caller-saved trebuie** salvati pe stiva astfel incat in dreptul etichetei **finish** sa fie depozitata in **s** valoarea corecta (suma dintre mediile elementelor consecutive dintr-un vector luate 2 cate 2) ? Alegeti una sau mai multe variante de raspuns. * Un punct

- ☐ %eax
- ☒ %ecx
- ☒ %edx
- ☐ niciunul

De ce valoarea returnata de procedura nu este folosita corect in cadrul acestui program * Un punct
(procedura returneaza catul mediei aritmetice a celor doua argumente)?

- ☐ Valoarea nu a fost depozitata din procedura nicaieri
- ☐ %eax nu a fost salvat pe stiva ca registru caller-saved
- ☒ Valoarea returnata a fost alterata dupa iesirea din procedura inainte de a fi folosita
- ☐ Returnarea nu se realizeaza prin registrul corect

Acest formular a fost creat în domeniul Universitatea din București.

Formulare Google