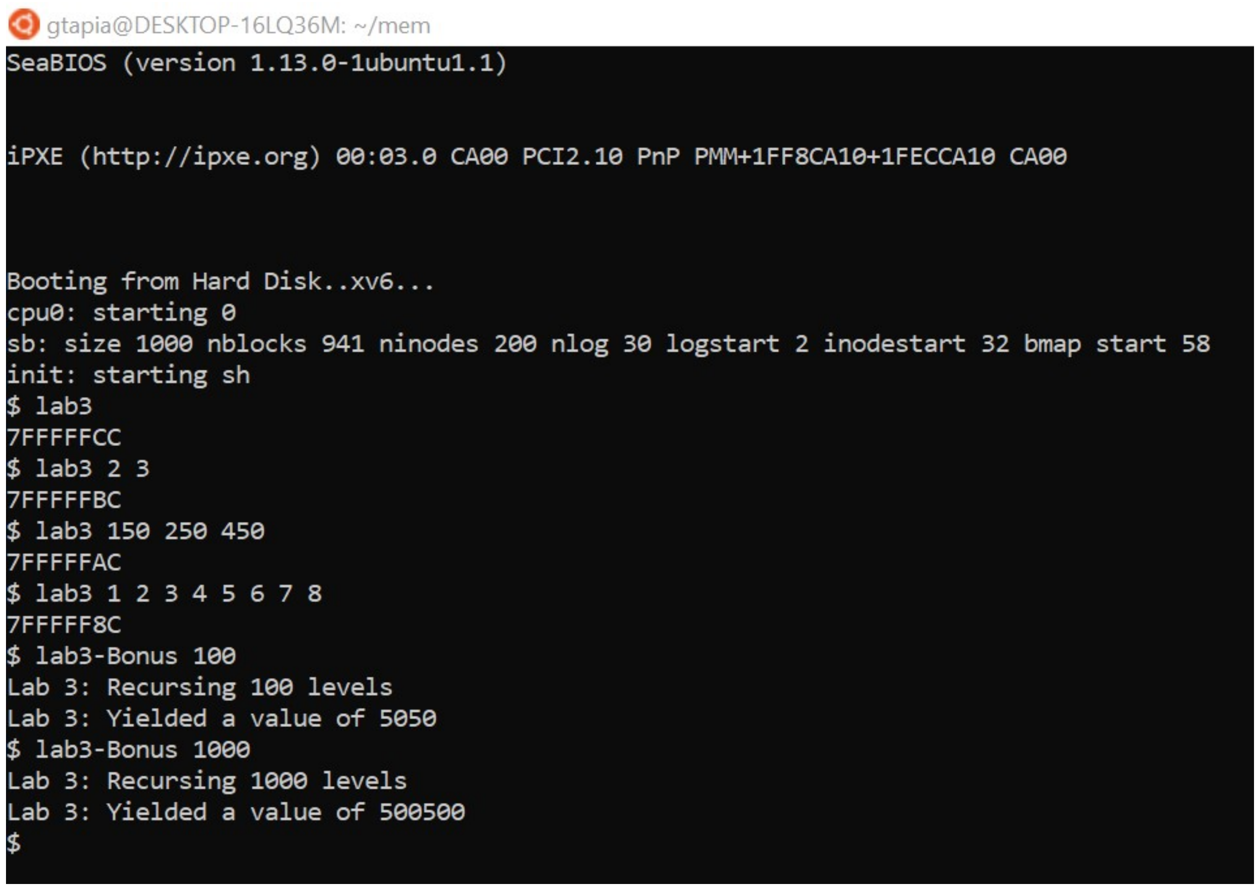


gtapia@DESKTOP-16LQ36M: ~/mem

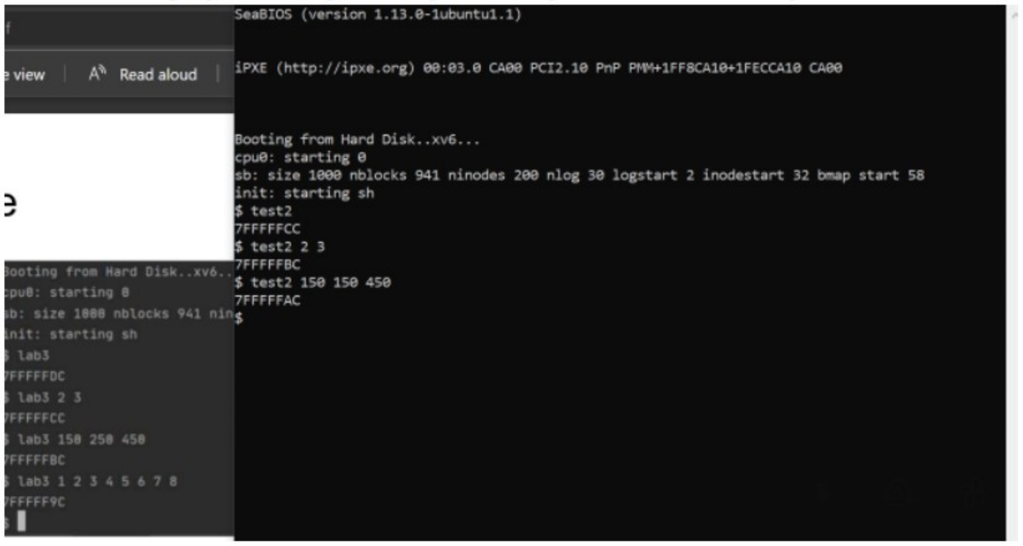
```
SeaBIOS (version 1.13.0-1ubuntu1.1)


iPXE (http://ipxe.org) 00:03.0 CA00 PCI2.10 PnP PMM+1FF8CA10+1FECCA10 CA00



Booting from Hard Disk..xv6...
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap start 58
init: starting sh
$ lab3
7FFFFFCC
$ lab3 2 3
7FFFFFBC
$ lab3 150 250 450
7FFFFFAC
$ lab3 1 2 3 4 5 6 7 8
7FFFFF8C
$ lab3-Bonus 100
Lab 3: Recursing 100 levels
Lab 3: Yielded a value of 5050
$ lab3-Bonus 1000
Lab 3: Recursing 1000 levels
Lab 3: Yielded a value of 500500
$
```

Passed all test cases. I was off by one letter but the professor said it's ok, that he got similar results. Email attached below.

George Tapia <gtapia@hawk.iit.edu>
to Yue

Good afternoon professor,

I was wondering if you had any idea why I am off by a letter when I run my test case?



```
Booting from Hard Disk..xv6..
cpu0: starting 0
sb: size 1000 nblocks 941 nin
init: starting sh
$ lab3
7FFFFFDC
$ lab3 2 3
7FFFFFCC
$ lab3 150 250 450
7FFFFFBC
$ lab3 1 2 3 4 5 6 7 8
7FFFFF9C
$
```

```
SeaBIOS (version 1.13.0-1ubuntu1.1)


iPXE (http://ipxe.org) 00:03.0 CA00 PCI2.10 PnP PMM+1FF8CA10+1FECCA10 CA00



Booting from Hard Disk..xv6...
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap start 58
init: starting sh
$ test2
7FFFFFCC
$ test2 2 3
7FFFFFBC
$ test2 150 150 450
7FFFFFAC
$
```

sincerely,
George T

Yue Duan
to me

I got similar results from my end. I think the slides may have some issues. Anyway, not a big deal.

```
Binary files mem/.git/index and org/.git/index differ
diff -r mem/.git/logs/HEAD org/.git/logs/HEAD
< 0000000000000000000000000000000000000000
1496296c01bd50765654e61c227599ede447194d gtapia
<gtapia@DESKTOP-16LQ36M.localdomain> 1650727538 -0500   clone: from
https://github.com/naelag/lab2-f17.git

diff -r mem/Makefile org/Makefile
180,181d179
<        _lab3\
<        _lab3-Bonus\
```

diff -r mem/exec.c org/exec.c
33,34c33
<   if(readi(ip, (char*)&elf, 0, sizeof(elf)) != sizeof(elf)) {
   Testing purposes
<   // cprintf("bad happened 0, %x\n");

>   if(readi(ip, (char*)&elf, 0, sizeof(elf)) != sizeof(elf))
36,37d34
<   }
<
43c40
<

>
46,48d42
>   // Allocate page at the next page boundary.
>   // Make the first inaccessible. Use the second as the user stack.
<   if((sz=allocuvm(pgdir,sz, sz + PGSIZE)) == 0)
<   goto bad;
<   clearpteu(pgdir, (char*)(sz - PGSIZE));
68a63,64
70,71c66
//setting pointer to point to the rounded lower multiple of page size(KERNBASE-1)
<   uint stack_pointer = PGROUNDDOWN(KERNBASE-1);
//allocating a page, but if the page is null, then this means it fails and so goto bad handles it.
<   if((stack_pointer = allocuvm(pgdir, stack_pointer, stack_pointer + 1 *PGSIZE-1)) == 0) {

>   if((sz = allocuvm(pgdir, sz, sz + 2*PGSIZE)) == 0)
73,74c68,69
<   }
<   sp = stack_pointer;   **// init stack pointer to the top byte of address**

>   clearpteu(pgdir, (char*)(sz - 2*PGSIZE));
>   sp = sz;
78,79c73
<   if(argc >= MAXARG) {
//added the cprintf for testing purposes
<   cprintf("bad happened 2\n");

>   if(argc >= MAXARG)
81d74
<   }

```
83,84c76
<     if(copyout(pgdir, sp, argv[argc], strlen(argv[argc]) + 1) < 0) {
<       cprintf("bad happened 3\n");
---
>     if(copyout(pgdir, sp, argv[argc], strlen(argv[argc]) + 1) < 0)
86,87d77
<     }
<
97,98c87
```
Testing purposes, print statement
```
<   if(copyout(pgdir, sp, ustack, (3+argc+1)*4) < 0) {
<     cprintf("bad happened 4\n");
---
> if(copyout(pgdir, sp, ustack, (3+argc+1)*4) < 0)
100c89<   }
---
>
115,117d103
<
<   curproc->stack_end = PGROUNDDOWN(sp); // size of stack of process
<
129d114
<

diff -r mem/memlayout.h org/memlayout.h
5a6
>

diff -r mem/proc.c org/proc.c
202,203d201
<   //assigning the parents process stack size to child
<   np->stack_end = curproc->stack_end;
Only in mem: proc.d
diff -r mem/proc.h org/proc.h
52d51
<   uint stack_end;     // End of process stack

diff -r mem/syscall.c org/syscall.c
22,24c22,23
```
//added the if statements to check if it is within the bounds specified when fetching the value
```
<   if((addr >= KERNBASE || addr+4 > KERNBASE) && (addr > curproc->stack_end)){
<     cprintf("Bad adrress\n");
<     return -1;}
```

```
---
>   if(addr >= curproc->sz || addr+4 > curproc->sz)
>     return -1;
41c40
<   ep = (char*)KERNBASE-1;
---
>   ep = (char*)curproc->sz;
67,69c66
<   if(size < 0)
<      return -1;
<  if(( (uint)i >= KERNBASE || (uint)i+size > KERNBASE) && (uint)i < curproc->stack_end)
---
>   if(size < 0 || (uint)i >= curproc->sz || (uint)i+size > curproc->sz)

diff -r mem/trap.c org/trap.c
39d38
<   uint fault_addr;
51,62d49
```

rcr2() reads the address that caused the page fault from register CR2, which should be from the page directly below the current bottom of the stack

```
<   case T_PGFLT:
<   fault_addr = rcr2();
<   if(fault_addr < myproc()->stack_end && fault_addr >= myproc()->stack_end - 4096){
<       uint stack_pointer = PGROUNDDOWN(myproc()->stack_end - 1);
<   if((stack_pointer = allocuvm(myproc()->pgdir, stack_pointer, stack_pointer + PGSIZE)) == 0){
<       goto bad;}
<
<   myproc()->stack_end = PGROUNDDOWN(stack_pointer - 1);
< }
< else{
<   goto bad;}
<   break;
96d82
<     bad:

diff -r mem/vm.c org/vm.c
338,357d337
```

//Added extra handling to cover the stackpage and the pageguard situation

```
<   for(i = PGROUNDDOWN(KERNBASE - 1); i >= myproc()->stack_end; i-= PGSIZE){
<    if((pte =walkpgdir(pgdir,(void *) i, 0)) == 0) {
<      panic("copyuvm: pte should exist");
<    }
<    if(!(*pte & PTE_P)){
<      panic("copyuvm: page not present");
<    }
<    pa = PTE_ADDR(*pte);
<    flags = PTE_FLAGS(*pte);
<
<    if((mem = kalloc()) == 0) {
<     goto bad;
<   }
<
<   memmove(mem,(char*)P2V(pa), PGSIZE);
<   if(mappages(d, (void*)i, PGSIZE, V2P(mem), flags) < 0) {
<      goto bad;
<    }
< }
```