

## Harjoitusprojekti tehtävä 3

### Dokumentaatio

Harjoitustehtävässä tuli opiskella xv6 kernelin käyttöä, ja lisätä sinne oma funktio, jolla saadaan selville kuinka monta kertaa toista systeemifunktiota `read()` on käytetty. Tein systeemikutsujan, joka palauttaa tämän arvon, sekä oman komennon, jolla systeemikutsun toimivuutta voidaan testata.

Ympäristönä on käytetty Ubuntu 16.04 (Microluokan tietokone), sekä QEMU emulaattoria.

Systeemikutsun nimi on `getreadcount()`, sen käyttämä globaali muuttuja on `READCOUNT`, sekä oma komento on nimeltään `mycode`.

Koodin tuloste on kirjattu tämän dokumentin loppuun. Lopputulos vaikuttaa oikealta, se kasvaa aina hieman kun terminaalia käyttää.

Lähes kaikki oma koodi on merkattu kommentilla `// OWNCODE`. Näin esimerkiksi `grep:llä` löydetään nopeasti kaikki itse tehty koodi.

Listaan seuraavaksi tiedostot joita olen muuttanut, ja kerron miksi tein mitä tein.

```
readcounter.h
extern int READCOUNTER;
```

Itse tehty header file. Se sisältää rivin `extern int READCOUNTER;`. Tämä luo globaalin muuttujan, jota käytän read kutsujen kirjaamisessa. Globaali muuttuja ei ole paras ratkaisu, mutta tähän tehtävään hyvin soveltuva.

```
mycode.c
#include "types.h"
#include "stat.h"
#include "user.h"

/* syscall.c: syscalls[] ? */
int
main(int argc, char *argv[]) // OWNCODE
{
    int readcount;
    int i;
    readcount = getreadcount();

    for(i = 1; i < 2; i++)
        printf(1, "%d%s", readcount, i+1 < 2 ? " " : "\n");
    exit();
}
```

Kutsuu omaa funktiota `getreadcount()`, joka palauttaa `READCOUNTER:n` arvon. Tämän jälkeen arvo tulostetaan terminaaliin. Tämä funktio on apufunktio, jonka avulla systeemifunktion toimivuutta voidaan testata.

```
sysproc.c
    int
    sys_getreadcount(void) // OWNCODE
    {
        return READCOUNTER;
    }
```

Lisätty oma funktio. Funktio palauttaa globaalin muuttujan `READCOUNTER` arvon.

`syscall.c`, `syscall.h`, `user.h`, `usys.S`, `Makefile`

Lisätty tiedot omasta systeemikutsusta, sekä omasta `mycode` -komennosta. Esimerkkinä on käytetty systeemikutsuissa funktiota `getpid()`, sekä komennoissa `echo` -komentoa.

`sysfile.c`

`sys_read` funktioon on lisätty rivi: `READCOUNTER++`. Tämä lisää globaalin muuttujan `READCOUNTER` arvoa aina, kun `read()` funktio ajetaan. Tämä mahdollistaa komennon seuraamisen. Tiedostoon on myös annettu tieto muuttujasta `#include:n` avulla

`main.c`

Lisäsin rivin: `#include "readcounter.h" // OWNCODE`, jotta mainilla on tieto omasta funktiostani.

```
n7441@LUT6199:~/xv6-public$ make qemu
qemu-system-i386 -serial mon:stdio -drive file=fs.img,index=1,media=disk,format=raw
-drive file=xv6.img,index=0,media=disk,format=raw -smp 2 -m 512
xv6...
cpu1: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap start 58
init: starting sh
$ mycode
7
$ ls
.          1 1 512
..         1 1 512
README    2 2 2327
cat        2 3 13412
echo       2 4 12484
forktest   2 5 8200
grep       2 6 15236
init       2 7 13076
kill       2 8 12528
ln         2 9 12432
ls         2 10 14648
mkdir      2 11 12548
rm         2 12 12528
sh         2 13 23168
stressfs   2 14 13204
usertests  2 15 56084
wc         2 16 14064
zombie     2 17 12256
mycode     2 18 12456
console    3 19 0
$ echo asd
asd
$ mycode
59
$ mycode
66
$ echo das
das
$ mycode
82
$
```