IT-Universitetet i København 6. september 2016

Operativsystemer og C

Obligatorisk opgave 1

Rapporten - som **pdf-fil** - samt kildekode skal pakkes og navngives „BOSC-opg1-dit-navn“ og uploades til LearnIT - en per gruppe - senest onsdag den 28. september, klokken 12:00.

Baggrund

En shell er et program som tilbyder en brugergrænseflade til operativsystemet. Brugeren kan taste kommandoer ind og få dem udført på den ønskede måde.

Et eksempel kunne være en bruger, der ønsker at se filerne i en folder. Brugeren intaster `ls` i shell'en, hvorefter shell'en parser og processerer strengen „ls“ og begynder at søge efter den udførbare fil ls i filsystemet. Efter noget tid finder den filen `/usr/bin/ls`, der skal eksekveres. Til det formål starter shell'en en ny proces som sættes til at eksekvere `/usr/bin/ls`. Herefter venter shell'en på at programmet bliver færdigt. Hvis der er fejl meddeles denne, ellers forsættes med at læse den næste kommando som brugeren indtaster.

I denne opgave skal du lave din egen shell, som vi kalder `bosh` (BOSC shell), med funktionalitet svarende til en begrænset version af `bash` på Linux.

Specifikation af `bosc`

Hvad skal `bosh` - som minimum - kunne:

- `bosh` skal kunne virke uafhængigt. Du må ikke bruge andre eksisterende shells, f.eks. er det ikke tilladt at anvende et systemkald `system()` til at starte `bash`.
- Kommando-prompt'en skal vise navnet på den host den kører på.
- En bruger skal kunne indtaste almindelige enkeltstående kommandoer, så som `ls`, `cat` og `wc`. Hvis kommandoen ikke findes i operativ systemet skal der udskrives en „Command not found“ meddelelse.
- Kommandoer skal kunne eksekvere som baggrundsprocesser (ved brug af `&`) sådan at mange programmer kan køres på samme tid.
- Der skal være indbygget funktionalitet som gør de muligt at lave redirection af stdin og stdout til filer. F.eks skal kommandoen

```
wc -l < /etc/passwd > antalkontoer
```

lave en fil „antalkontoer“, der indeholder antallet af brugerkontoer.

- Det skal være muligt at anvende pipes. F.eks. skal

  ```
  ls | wc -w
  ```

  udskrive antallet af filer.

- Funktionen `exit` skal være indbygget til at afslutte shell'en.
- Tryk på Ctrl-C skal afslutte det program, der kører i `bosh` shell'en, men ikke shell'en selv.

Du er velkommen til at tilføje mere funktionalitet efter eget ønske.

Hints

På kursusbloggen er uploadet `oo1.zip`, som indeholder en ufuldstændig version af `bosh`, der kan benyttes som udgangspunkt. Hertil hører også en simpel `Makefile`, som du selv kan modificere efter behov.

Den ufuldstændige version af `bosh` giver mulighed for at indtaste kommandoer, som parses til en shellcmd `struct` med følgende definition:

```
typedef struct _cmd {
    char **cmd;
    struct _cmd *next;
} Cmd;

typedef struct _shellcmd {
    Cmd  *the_cmds;
    char *rd_stdin;
    char *rd_stdout;
    char *rd_stderr;
    int background;
} Shellcmd;
```

Programmet benytter sig af GNU readline biblioteket, hvilket betyder at man kan indtaste de samme tegn og kommandoer som i `bash`. Desuden er der implementeret en „history“ funktion som gør det muligt at browse igennem tidligere kommandoer.

Første skridt efter at du har downloadet `oo1.zip` (og evt. `apt-get install unzip`):

```
# unzip oo1.zip
# cd oo1
# make
# ./bosh
```

Bemærk, at kommandoerne parses i modsat rækkefølge af den de er indtastet i (med god grund). Du kan kigge i `print.c` og i funktionen `printshellcmd()` for at se hvordan `struct`'en tilgås.

Fremgangsmåde

Det er muligt at implementere funktioner for alle specifikationerne i den rækkefølge de er skrevet overfor. Implementér dem én af gangen. Test funktionaliteten og når det ser ud til at fungere som ønsker, så gå videre til næste punkt.

I forbindelse med at skrive C funktionerne er det nødvendigt at studere diverse systemkald. For nogle af dem, f.eks. `exec`, eksisterer der forskellige varianter. Du er nødt til at finde den der bedst passer til dit behov - nogle gange er der adskillige der er brugbare.

Systemkald som du helt sikkert skal bruge: `fork, exec, wait, pipe, dup`.

Rapport

Rapporten skal dels være dokumentation af jeres arbejde med projektet og give indblik i jeres tankegang. Samtidigt er rapporten et produkt i sig selv og ment som en træning i at skrive rapporter inden i skal til at lave bachelor rapporten senere i jeres uddannelse. Rapporten skal derfor være et selvstændigt dokument (forside, indledning, beskrivelse af hvordan hver enkelt funktionalitet er implementeret, testet, samt en konklusion). Alt relevant kildekode (+Makefile) skal inkluderes i et appendix samt zippes med afleveringen.