Operating Systems — Exercise Sheet 2

2016-04-13

Task 1

Write a program which forks itself a random number of times (1–10). Each child process prints following message upon creation:

```
Hello I am a child process, my PID is $CHILD_PID and my parent has the PID $PARENT_PID.
```

After this, the child process should wait a random number of seconds (1–30) and terminate after printing following message:

```
Hello I am a child process $CHILD_PID, I will terminate now.
```

The parent process should print following message before creating children:

```
Hello I am a parent process, my PID is $PARENT_PID and I will now create $n children.
```

Finally the parent process waits for each child process to finish and prints a message for each finished child process:

```
Child process $CHILD_PID has finished.
```

After all children have finished their work, the parent process should also terminate with a message.

Task 2

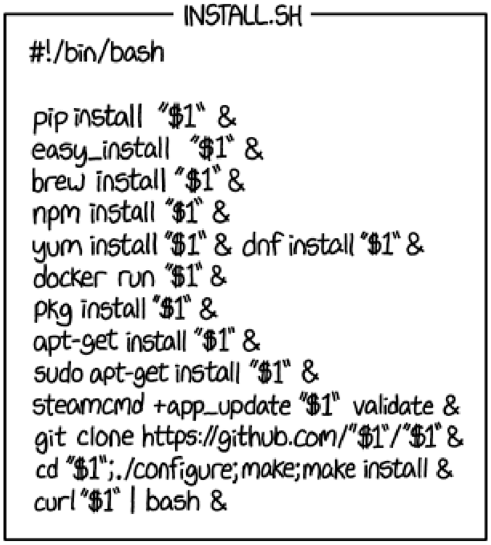
Write a program which reads commands from a file and executes them. The file contains one command per line (with arguments). It may also contain empty lines and comments (starting with a `#`). Similar to bash it should be possible to run a command *in background* by adding an `&` after it. An example file:

```
# This is a comment followed by an empty line

/usr/bin/whoami
/bin/echo hello world
/bin/false
/usr/bin/id &
/bin/date
```

Your program should take a filename and an optional flag `-e` as argument. If the flag is specified your program should terminate as soon as one command terminates with an exit code $\neq 0$, continue otherwise. In the example file the command `/usr/bin/id` should not be executed (given `-e`) since `/bin/false` will yield an exit code of 1.

Task 3



INSTALL.SH

```
#!/bin/bash

pip install "$1" &
easy_install "$1" &
brew install "$1" &
npm install "$1" &
yum install "$1" & dnf install "$1" &
docker run "$1" &
pkg install "$1" &
apt-get install "$1" &
sudo apt-get install "$1" &
steamcmd +app_update "$1" validate &
git clone https://github.com/"$1"/"$1" &
cd "$1";./configure;make;make install &
curl "$1" | bash &
```

Figure 1: Universal Install Script

Figure 1, a recent XKCD strip[1], shows an *Universal Install Script.* Its caption states the following:

> The failures usually don't hurt anything, and if it installs several versions, it increases the chance that one of them is right. (Note: The `yes` command and `2>/dev/null` are recommended additions.)

Explain *each* line of the script and also comment on the caption.

[1] https://xkcd.com/1654/