# Lab 1b Supplement: The Devil Shell by Example
# COMPSCI 210: Introduction to Operating Systems
# Due Date: Friday, 5 October 2012 at 11:59pm.

To supplement the lab1b handout, here we provide a walk-through of the *devil shell* by issuing some sample jobs. You can issue all the commands yourself on the sample shell, `dsh-example`, available in the repository.

To log the errors, you can use `dup2()` system call. If you want to log both the `stdout` and `stderr`, you can use the combination of `tee()` and `dup2()` system calls. You are required to log only the `stderr` of the `dsh`. However if log everything, you can request for extra-credit.

Note: The extra line after each job execution is added here for the readability and need not be present in the actual `dsh` output.

```
$ ls #My working directory for lab1b
dsh.c  dsh.h  Makefile

# Run make to produce the executable dsh
$ make
gcc -I. -Wall -DNDEBUG -o dsh dsh.c

#Move the executable to dsh-example; dsh-example is provided to
   you for playing around
$ mv dsh dsh-example

$ ./dsh-example #Starting the shell
#The process id (pid) for the shell is displayed by the prompt
dsh-23822$ ls
23823(Launched): ls
dsh.c  dsh-example  dsh.h  dsh.log  Makefile

dsh-23822$ sleep 50 #Issuing a sleep
23824(Launched): sleep 50
^Z    #Stopped the job by issuing ctrl-z

dsh-23822$ jobs #Checking the status of the jobs
23823(Completed): ls #The completed jobs are displayed only for
   the first time
23824(Stopped): sleep 50

dsh-23822$ jobs #Checking the status of the jobs again
23824(Stopped): sleep 50 #The completed jobs disappear


dsh-23822$ bg 23824 #Resuming my sleep job and putting it to
```

```
   background

dsh-23822$ jobs #Prompt immediately returns
23824(Running): sleep 50 #The status will be running
^Z    #Stopping again the job by issuing ctrl-z

dsh-23822$ fg 23824 #Resuming my sleep job
^C    #Terminating the job by issuing ctrl-c

dsh-23822$ jobs #Checking the status again
23824(Completed): sleep 50 #The status for sleep is now completed
    since we terminated the job by issuing ctrl-c

dsh-23822$ sleep 10 & #Launching a job in the background
23825(Launched): sleep 10

#Run the jobs within 10 seconds after issuing ‘‘sleep 10’’
dsh-23822$ jobs
23825(Running): sleep 10 #Status of the sleep is running (in the
   background)

#Now, let the job complete by waiting for 10 seconds. No message
   will be displayed on the shell until we explicity request for
   it. Wait for 11 seconds and issue ‘‘jobs’’
dsh-23822$ jobs
23825(Completed): sleep 10 #Now the status show my sleep is
   completed

#Example of multiple jobs separated by the symbol ‘‘;’’
dsh-23822$ ls; ps
23840(Launched): ls
dsh.c  dsh-example  dsh.h  dsh.log  Makefile
#The second job is launched sequentially after the first job is
   completed
23841(Launched): ps
  PID TTY          TIME CMD
 23841 pts/13   00:00:00 ps
17690 pts/13   00:00:00 tcsh
17705 pts/13   00:00:00 bash
23822 pts/13   00:00:00 dsh-example

dsh-23822$ jobs #Issue jobs to see the status
23840 (Completed): ls #The status shows as two seperate jobs were
    run: ‘‘ls’’ followed by ‘‘ps’’
23841(Completed): ps

dsh-23822$ ls | wc -l #Example with pipes
23843(Launched): ls | wc -l
5

dsh-23822$     #quit the dsh by issuing ctrl-d


#Back to our launching shell; dsh.log will contain the log info
   written to stderr
$ls
dsh.c  dsh-example  dsh.h  dsh.log  Makefile
```

```
$ cat dsh.log #examining the log file
23823(Launched): ls
23824(Launched): sleep 50
23824: Stopped by signal 127 #Singal number for ctrl-z; can be
   obtained by issuing WTERMSIG(process->status)
23823(Completed): ls
23824(Stopped): sleep 50
23824(Running): sleep 50
23824: Stopped by signal 127
23824: Terminated by signal 2 #Signal number for ctrl-c
23824(Completed): sleep 50
23825(Running): sleep 10
23825(Completed): sleep 10
23840(Launched): ls
23841(Launched): ps
23840(Completed): ls
23841(Completed): ps
23843(Launched): ls | wc -l #The completion for this job is not
   shown in the log as we did not issue ‘‘jobs’’ command
```