Process Information Script

This homework will require you to write a bash script that provides information related to a specific process. By default, the script will simply indicate if the process id currently belongs to a process that is executing. Additional options will provide information about the process if it exists. Your script must complete all of the shalls listed below. You man **not** use `lsof`, `ps`, or any other native Linux command that completes these requirements. You should only need to use `ls`, `grep`, `find`, or `cat`. You can use other commands as well (anything we talked about in class), but don't try to get too creative. You can always ask if you are unsure.

1. The script **shall** accept a process id via command line.
   (a) The process id **shall** be entered as an argument to the script[1].
   (b) If no process id is given, the script **shall** print usage information. This is only if the *-f* option was not given. If the *-f* option is used, see 2c.
   (c) If the value entered for the process id does not refer to a running process, the script **shall** indicate that the process does not exist and exit. Other command options (besides help) should not be performed.
   (d) If the value entered for the process id refers to a running process, the script **shall** indicate that the process exists.
2. The script **shall** accept 5 command options:
   (a) When given the *h* command option, the script **shall** print usage and option information and exit. The script should not print anything else regardless of the other command options.
   (b) When given the *c* command option, the script **shall** print the command-line used to start the program. This should have all the necessary spaces and appear exactly as the user typed it in (ignoring extraneous white space).
   (c) When given the *e* command option, the script **shall** print executable information related to the selected process id. The information should include the path to the executable that was used to start the process.
   (d) When given the *s* command option, the script **shall** print status information related to the selected process id.
   (e) When given the *s* command option, the script **shall** expect an argument for the option to be used in filtering the status information. Only status information that patches the supplied pattern should be printed. This filtering should be done via the *grep* command.
   (f) When given the *f* command option, the script **shall** print all process ids that might have a given file opened. No process id will be given if this option is specified. Instead, a file must be provided for the option argument.

[1] As opposed to a command option or command switch.

3. The script **shall** handle notable error conditions:

   (a) If no argument is provided for the *s* option, the script **shall** display an error and exit.

   (b) If no argument is provided for the *f* option, the script **shall** display an error and exit.

Example

```
sam@sam-VirtualBox:~$ ./hw4.sh 1234
Process 1234 is running
sam@sam-VirtualBox:~$ ./hw4.sh 1234 -e -c
Process 1234 was started by /bin/cat
Process 1234 was started as cat somefile.txt
sam@sam-VirtualBox:~$ ./hw2 -f samsfile
Process 1305 /bin/cat is using /home/sam/samsfile.txt
```

Figure 1: Example output for the program. Your output does not have to match exactly.