For each of the following questions, the answer should be given as a single sequence of piped Unix commands, except of course if a script is explicitly asked for, or if the answer must be given in words.

1. Rewrite the following command by using egrep instead:

```
grep '*\(\[\{ab\}\]\){2,4}\' filename
```

Here is the equivalent egrep command:

```
egrep '*\(\[^\{ab\}\]\){2,4}\' filename
```

2. Which words of the websters dictionary does this command print out?

```
grep '\(\.{1}\)\\\\\\\1\\2' websters
```

The words containing three consecutive identical characters, e.g., `bossship`, `demigoddessship`, `goddessship`, `headmistressship`, `patronessship`, `wallless`, `whenceeer`.

3. An HTML comment correctly parsed by most browsers begins with `<!--`, ends with `-->`, and does not contain `--` or `>` anywhere in the comment.

   (a) Use egrep to find all lines containing an HTML comment in the files of your working directory whose names end with `.html`.

```
egrep '<!--\[^>\-\]*\{-0,1\}\[^>\-\]*-->\.' html
```

(b) Use sed to replace in each file the comment written using the HTML syntax by the same comment using the syntax of C (that is using `/*` and `*/` for the beginning and end of comment) everywhere except from the last line of a file.

```
sed '$!s/\<!--\([-\]\{-0,1\}\)[-\]*\-->\/*/\*\1*/\g/` html
```
4. The command yes of Unix simply prints out y on each line, forever. For example,

```bash
$ yes | head -2
y
y
```

What does the following generate?

```bash
$ yes | head -10 | cat -n | \ sed -n -e '/1/,/7/ p' -e '/5/,/9/ p'
```

```bash
$ yes | head -10 | cat -n | \ sed -n -e '/1/,/7/ p' -e '/5/,/9/ p'
```

1 y
2 y
3 y
4 y
5 y
6 y
7 y
8 y
9 y
10 y

5. On i5, the command ps produces an output such as the following where the fields are tab-separated:

```
 możesz ps -delaf | head
          FS UID PID PPID C PRI NI ADDR SZ WCHAN STIME TTY TIME CMD
m02  0 S nobody 13280 13269 0 40 20 ? 1197 ? Aug 12 ? 5:01 /usr/apache/bin/httpd
m02  0 S nobody 13402 13269 0 98 20 ? 1190 ? Aug 12 ? 4:14 /usr/apache/bin/httpd
m02  0 S daemon 13138 13024 0 41 20 ? 335 ? Aug 12 ? 0:00 /usr/sbin/rpcbind
m02  0 S nobody 17351 13269 0 40 20 ? 1179 ? Aug 12 ? 4:07 /usr/apache/bin/httpd
m02  0 S nobody 13805 13269 0 98 20 ? 14368 ? Aug 12 ? 11:33 /usr/apache/bin/httpd
m02  0 S root 13043 13024 0 40 20 ? 1174 ? Aug 12 ? 0:04 /lib/svc/bin/svc.configd
m02  0 S mm007 26138 26133 0 98 20 ? 1074 ? 17:23:07 ? 0:00 /usr/lib/sdh
m02  0 S smxap 17246 13024 0 98 20 ? 913 ? Aug 18 ? 0:32 /usr/lib/sendmail -Ac -q15m
m02  0 S nobody 13282 13269 0 40 20 ? 1180 ? Aug 12 ? 4:16 /usr/apache/bin/httpd
```

(a) Write a bash script called pgrep that returns the process IDs of the processes whose name matches the regular expression provided as argument (e.g., pgrep http*).
#!/bin/bash

USAGE="Usage: $0 regexp
options:
    -? help"

while getopts ?\? c
do
    case $c in
        \?) echo "$USAGE" 1>&2 ; exit 0 ;;
        esac
done

ps -delaf | \
    gawk '$3!="UID" && $3 !~ /nobody|root/' | \
    egrep $* | \
    gawk '{ print $4 }'

(b) Similarly, write a bash script pkill that can be used to kill all
processes whose name matches the regular expression provided
as argument.

#!/bin/bash

USAGE="Usage: $0 regexp
options:
    -? help"

while getopts ?\? c
do


3
case $c in
  \?) echo "$USAGE" 1>&2 ; exit 0 ;;
esac
done
shift 'expr $OPTIND - 1'

if
[ $# -eq 0 ]
then
echo "$USAGE" 1>&2
exit 1
fi

for proc in
  'pgrep $*'
do kill -9 $proc
done

(c) Print all user IDs running more than four processes.

i5$ ps -delaf | nawk '{ np[$3]++ }\nEND{ for (u in np) if(np[u]>4) print u }'

(d) Write a bash script that sends email to user IDs other than nobody and root running more than 20 processes.

!/bin/bash

for uu in
  'ps -delaf | nawk '{ np[$3]++ }\nEND{ for (u in np) if(np[u]>20) print u }'\ndo mail $uu << EOM
You are running more 20 processes.
EOM
done

(e) gawk has a special function, strftime, for creating strings based on the current time, e.g.,
Use that to show all the processes that have started in the last two hours, assuming that this is not done around midnight.

```bash
ps -delaf | \ 
gawk '$21 ~ /:/ { h=strftime("%H"); m=strftime("%M"); \ 
split($21, arr,":"); 
print $21, h":"m }'
```