Lab 4

1. (E) **AWK:** Print only the first 4 fields from each even-numbered line from a file, considering that the fields are separated by whitespaces. If a line has fewer than 4 fields, print all of them.

```
awk 'NR % 2 == 0 {print $1,$2,$3,$4}' file
```

2. (E) **GREP:** Print all the lines that contain only non-alphanumeric characters from a file. (any character that isn't a letter or a digit).

```
grep -E -v "[a-zA-Z0-9]" file
```

3. (E) **SED:** Duplicate each occurrence of an integer number from a file. We will consider that an integer number is a sequence of neighboring base 10 digits.
   - Ex: line "This 1234 is a number" will become "This 12341234 is a number"
   - Ex: line "56.34" will become "5656.3434"

```
sed -E 's/([0-9]+)/\1\1/' file
```

4. (E) **SED:** Swap field number 2 with field number 3 from a file where the fields are separated by the "**:**" character (Ex. /etc/passwd or passwd.fake if available, but any file where fields are separated by the : character should do)

```
sed -E "s/^([^:]*):([^:]*):([^:]*):/\1:\3:\2:/" /etc/passwd
```

5. (M) **GREP:** Display all the lines from a file that contain between 2 and 4 occurrences of the letter i, not necessarily consecutive.
   - this line contains 4 **i**'s and that's ok -> match
   - this line has only 2 -> match
   - this one has only one -> no match
   - this line contains five or more **i**'s -> no match

```
grep -E "^([^i]*i[^i]*){2,4}$" file
```

6. (M) **SED:** Delete all characters after the last whitespace from each line from a file.
   - Ex: line *"A regular, boring line"* will become "*A regular, boring "*
   - Ex: line *"A less regular ;;&*^line"* will become *"A less regular "*

```
sed -E "s/ [^ ]*$/ /" file
```

7. (M) **AWK:** Print the line number and the field from the middle of the line from each line that contains an odd number of fields from a file. Consider that the fields are separated by whitespaces. Note: division in awk is by default float division. If you need the integer part of a division use the int function. Ex: int(5/2) = 2.

```
awk 'NF % 2 == 1{i=int(NF/2)+1; print NR,$i}' file
```

8. (M) **SED:** Remove the first word containing only lowercase letters from each line of a file. (We will consider words as being any string of consecutive letters)

```
sed -E "s/\<[a-z]+\>//" file
sed -E "s/[[:<:]][a-z]+[[:>:]]\b//" file - !! this maybe works on macOS, it will not work on the exam server !!
```

9. (H) **GREP:** Print all lines that contain at most 5 vowels, not necessarily consecutive, situated between 2 **^** signs from a file.
   - Ex: line *"aei^, still works^"* satisfies the condition
   - Ex: line *"abc^, way too many vowels here ^"* has too many vowels between the two **^**
   - Ex: line *"^here there are too many vowels^but not here^"* satisfies the condition because there are 4 vowels between the second and third occurrences of the **^** character

```
grep -E "\^([^aeiou]*[aeiou][^aeiou]*){0,5}\^" file
```

10. (H) **AWK:** Print the processes from the system (use the ps.fake file or run ps -ef) that have a cumulated CPU time greater than 10 minutes. (See the TIME column of the ps command)
    If you test on *ps.fake* the output should be:

```
root         78     2  0  2013 ?        10:33:01 [kipmi0]
root       1315     1  0  2013 ?        00:15:21 /sbin/rsyslogd -i /var/run/syslogd.pid -c 5
mongodb    1630     1  0  2013 ?        02:47:37 /usr/bin/mongod --quiet -f /etc/mongodb.conf run
clam       3299     1  0 Feb01 ?        00:10:06 clamd
mysql     11312 11210  0 Feb13 ?        00:17:40 /usr/libexec/mysqld --basedir=/usr --datadir=/va
r/lib/mysql --user=mysql --log-error=/var/log/mysqld.log --pid-file=/var/run/mysqld/mysqld.pid -
-socket=/var/lib/mysql/mysql.sock
```

```
ps -ef | awk 'NR > 1 {split($7,s,":"); time=60*60*int(s[1]) + 60 * int(s[2]) + int(s[3]); if(time > 600) {print $0}}'
```

11. **(H) [GREP] + SED + AWK:** For each regular file from the current directory, display only the name of the file and the permissions for the user. (not the permissions for the *group* or for *other*; you can use **ls -l** to get information about files and folders from the current directory)

    Example:

    Consider that we have the following files in the current directory:

    drwxrwxr-x. 2 horeb horeb 6 Mar 16 13:34 dir1

    -rwxrw-r--. 1 horeb horeb 0 Mar 16 13:32 file1

    -rw-rw-r--. 1 horeb horeb 0 Mar 16 13:33 file2

    The expected output is:

    rwx file1

    rw- file2

```
ls -l | awk 'NR > 1 && $1 ~ /^-/{print $1,$NF}' | sed -E 's/.(...).{7}/\1/
```

The first 2 students who finish all problems from any one category will receive for the next test:

E -> + 0.5p

M -> + 1p

H -> + 1.5p

You can only get the bonus from one single category.