Lab 10
SED

Read carefully the instructions before performing!!!
Submission in pairs : 03/02/25 for the groups of Monday
06/02/25 for the groups of Thursday

SED (Stream Editor) *The primary purpose of sed is to automate text processing tasks such as searching, replacing, inserting, and deleting text. It is especially useful for batch editing of files.*

Syntax: sed [options] 'command' file(s)

1. Replacing with sed

sed 's/old_word/new_word/' <input_file >out_file

Here:
- s means “substitution”
- / are 3 delimeters between the fields
- old_word = word which will be replaced by a new_word
- input_file = file in which substitution will be done. This file will not be changed (!)
- out_file = file where the result file will be written

Example:
>cat try
I like milk
I like milk because my cat likes milk
We do not like this milk
The sun is white
The house is black
milkmilk

> sed 's/milk/boo/' <try >resTry

>cat resTry

I like **boo**
I like **boo** because my cat likes milk
We do not like this **boo**
The sun is white
The house is black
boomilk

Put your attention that only the first appearance of the string (milk) was changed.
Additional possibilities:

For a global changing (will change all the srings in the file)
sed 's/old_word/new_word/**g**' <input_file >out_file
To change specific position of the string:
sed 's/old_word/new_word/**num_of_pos**' <input_file >out_file

To change the string in a specific line number :
sed 'line_number s/boo/milk/' try
To write in the input file
sed **-i** 's/old_word/new_word/' input_file
Ignore cases
sed 's/old_word/new_word/**I**' input_file

Example:
>cat try
I like milk
I like milk because my cat likes milk
We do not like this milk
The sun is white
The house is black
milkmilk

> sed 's/milk/boo/**2**' <try >resTry
>cat resTry
I like milk
I like milk because my cat likes **boo**
We do not like this milk
The sun is white
The house is black
milk**boo**

> sed 's/milk/boo/**g**' <try >resTry
>cat resTry
I like **boo**
I like **boo** because my cat likes **boo**
We do not like this **boo**
The sun is white
The house is black
booboo

> sed '**2** s/milk/boo/' <try >resTry
>cat resTry
I like milk
I like **boo** because my cat likes milk
We do not like this milk
The sun is white
The house is black
milkmilk
> sed '**1,3** s/milk/boo/' <try >resTry #Change from line 1 to 3

> sed **'5,$** s/milk/boo/' <try >resTry #Change from line 5 to the end of the file

---

Delimeters:

We used / as a delimeter. However you can choose whatever you want, like :, |, + and others.

Example:

sed '**s:**old_word**:**new_word**:**' <input_file >out_file

Using multiply commands

sed **-e** 's/old/new/' **-e** '/pattern/d' file = Replace old by new **and** delete lines with the given pattern

2. Delete with sed

sed '/line_pattern/d' = Delete the line which contains the line_pattern

sed '/line_pattern/Id' = two flags: Delete the line which contains the line_pattern, ignoring the cases.

sed 'nd' filename = Delete the line number n.

Example
sed '5d' try = Delete the 5th line of try.
sed '2,4d' try = Delete the 2-4 lines of try.

Example:

>cat try
I like milk
I like milk because my cat likes milk
We do not like this milk
The sun is white
The house is black
milkmilk

> sed '/The/d'

I like milk

I like milk because my cat likes milk

We do not like this milk

milkmilk

3. Insert with sed

Insert a line after a match:

***sed* '/patern/a\new line' filename**

Example:

>cat try

I like milk

I like milk because my cat likes milk

We do not like this milk

The sun is white

The house is black

milkmilk

my cat and the cat

>sed '/We/**a** hello to you' try

I like milk

I like milk because my cat likes milk

We do not like this milk

hello to you

The sun is white

The house is black

milkmilk

my cat and the cat

4. Print specific lines

sed **-n** '/pattern/**p**' filename #Prints all the line contain the pattern

sed **-n** '<line_number>p' <file_name> #Prints specific numbers of lines

Examples:

sed -n '2p' try #Prints second line of the file try

sed -n '2,5p' try #Prints lines 2-5 of try

sed -n -e '2p' -e '5p' try #Prints lines 2 and 5 of try

5. Using variables in bash scripts

To use the values of the variables, use doubles quotes instead of single quotes

Example:

```
#!/bin/bash
VAR1="milk"
sed "s/$VAR1/foo/" try
```

Procedures

1. Create directory Lab10.

2. In script **sedCommands.sh**:

a). Write function **change** which uses sed to replace all occurrences of the word "cat" with "dog" in a file named animals.txt. And saves the output to a new file called updated_animals.txt.

b). Write function **print** which prints only the 3rd line of a file given as a parameter.

c). Write function **replace** which replaces "apple" with "orange" and "banana" with "grape" in the file fruits.txt. Save changes in place (in the file fruits.txt).

d). Write function **replaceBak** which replaces the word "linux" with "Linux" in manual.txt, while creating a backup of the original file with a .bak extension.

e). Write function **delete** which deletes all lines from line 10 to 20 in the file log.txt.

3. Write a script **sedRec.sh** which uses recursion and sed command to replace "2024" with "2025" in all .txt files within the current directory and its subdirectories **recursively**.

4. Every student who registers for an Operating Systems (OS) course at the college is required to complete courses that are prerequisites for this course. However, in practice the students do not fulfill this requirement. The lecturer of the OS course decided to enforce the requirement. She asked the secretary of the department for a list of all the students registered for the course with details of all the courses studied by each student so far and their grades in the course.

The lecturer did receive a file with the following format (the fields are

separated by spaces):

- in the first column appears id of the student.

- the student's name appears in the second column.

- columns three-seven list all the courses, according to the department's

recommended program and the student's grades in that course.

For example:

Student_ID	Student_Name	Hedva1	Python	English	Linux	Algebra
5245617890	Ido_Raziel	78	92	2	57	87
3451236748	Shon_Ruev	20	22	100	58	16
2455566767	Smadar_Cohen	98	58	67	51	89
6666678768	Alan_Goldshtein	80	76	48	95	88

The lecturer asks you to write a script named **students.sh** that receives a file according to the format shown above.
The list of prerequisites for the operating systems course is as follows:
Python and **Linux.**

The script will check the file and prepare a new file called **OSStudents.data** that will contain the names and IDs of the students with a score of at least 56 in each of the pre-courses. The output file will be sorted according to the average grades of the students (from highest to lowest) in the prerequisite courses and will contain in its last line the total number of students who are allowed to take the course.

For example, for the input file named **data** that appears above, after running the command
./students.sh data

A new file named OSStudents.data will be created containing the following information:

6666678768 Alan_Goldshtein

5245617890 Ido_Raziel

Sum: 2 students

Notes:
1). Each student can appear only once in an input file.
2). There is no need to check the correctness of the input.
3). Do not use temporary files.
4). **data** file is given as an example. Work on general files in the specified format.
5). The list of prerequisites is fixed and consists of two courses: Python and Linux.
6). Do not use awk to solve the question.

Submission:
Create one compressed file named Linux_Lab10_id1_id2.tar,
where id1 and id2 must be changed to the id numbers of the two partners.

This file will include the files **sedCommands.sh**, **sedRec.sh,** **students.sh.**

Submit only the file Linux_Lab10_id1_id2.tar in Moodle and only by one of the partners.

Appeals:

Can be submitted by e-mail to Elad mail@eladhuttner.net within a week from the date of publication of the grades.