# Operating Systems 2024-25 (Lab exercise II - 15%)

*__Deliverable:__* Your answers (linux scripts) in a word file and their results (Prt.Sc.)

**A.** Write a script called **searching** that (a) accepts two integers as arguments and (b) asks the user for a directory name, and then displays the following (for items 1-3 use the **find** command, for items 4-5 use **ls** and **grep** commands in pipeline):

1. The files of the tree of the given directory with permissions equal (as an octal equivalent) to the first number (argument).
2. The files of the tree of the given directory that changed contents during the last 'x' days, where 'x' is the second number (argument).
3. The subdirectories of the tree of the given directory that were accessed during the last 'x' days, where 'x' is the second number (argument).
4. The files of the given directory for which all the users have read access.
5. The subdirectories of the given directory for which only the owner has complete rights (create, rename, delete, access files); all the others can access files only if they know the name of the file.

Before printing each list (1 to 5) above, print an appropriate heading indicating (among other things), the number of files (or subdirectories) that will be printed. The script should execute iteratively (as many times as the user wishes - for different directories) and at the end (before the final exit) summarize the total number of files (or subdirectories) found in each case (from 1 to 5), for all directories searched.

**B.** Write a script called **teldb** that will manage a telephone catalogue that will be implemented in a file named **catalog**. The script should make the following:

- With the -a parameter it will add a new entry to the catalogue (i.e. to the **catalog** file). The addition will take place after the user asks for a name, surname, city, and the listing will be placed in a line (eg John Markou Peristeri 2105546789).
- With the -l parameter it will print the contents of the catalogue (with its numbered lines and omitting any blank lines).
- With the -s parameter followed by a number it will print the contents of the catalogue sorted by the corresponding column number (e.g. telcat -s 3 will display the contents sorted by city).
- With the -c parameter followed by a keyword it will only show us the directory lines that contain the keyword.
- With the -d parameter followed by a keyword and -b or -r, it will delete the directory lines containing the keyword. If the third parameter is -b it will insert in the position of each deleted line a blank line otherwise (-r) no.
- With the -n parameter it will print the number of blank lines in the catalogue, ask the user if he wants to delete these lines or not and act accordingly.

In any other case, an appropriate 'Usage' (indicating the correct usage of the script) message will be printed. Also, in the case of –c and –d parameters, if there are no lines containing the keyword the user should be notified with the appropriate message.

**C.** Write a script called **cmpdir** that compares the contents of two directories (whose names are given as arguments - and first the script should check if they are actually directories) with regard to the files they contain. As a result it will initially print for each directory separately, how many and which files are not in the other directory, and what their total size is. It will then print how many and what are the two directories' common files and their total size. Finally, it will move all common files of the two directories to a third directory (which will also be given / checked as an argument), and will create appropriate *hard links* from the two directories to them.

**D.** Write a script called **bck** that will get for a specific user (whose username is given as the *first argument*) a ***backup copy*** of one area of his account to another. The script should accept a directory (or file) as the *second argument*, create a *backup copy* of the area specified by this argument (with use of ***tar*** command), and copy it to the directory given as the *third argument*. If however the third argument is a file (and not a directory) then it should simply append the *backup copy* to that file. Next, modify script **bck** properly (naming it **bck1**) to perform the requested backup scheduled (with use of **at** command) at a specific *time* of your choice (try to give the *time* as an argument as well).

*__Note:__* Don't forget to make the necessary checks *for all the three arguments* (as well as for the total number of arguments given in the execution).