Assignment

This assignment in Systems Programming is designed to prepare you for the real time scenarios you might face in future. Each group is assigned with two tasks to be implemented. The implemented script should be efficient and well documented.

Each task is designed based on some topic taught in systems programming. You may also require to go through on line resources available. Total weightage for this assignment is 10 Marks.

Submission: You will wrap all your script files and a document file in a folder named with your batch number as a zip files. Document file should contain the names of the student, assumption made, and detailed description of the script. Submit this zip file on or before _____ (will be shared with you once I get an reply from Avinash).

Caution!!!

I expect you to submit your original work. You are not allowed to share your code / document / other details about your assignment with other batches implementing the same problem. Any evidence of such practice will attract severe penalty. I will not distinguish plagiarism as less or more.

2020HS70015@wilp.bits-pilani.ac.in	Shlok Sanjay Kamat	Group 5
2020HS70018@wilp.bits-pilani.ac.in	Shraddha Gulati	
2020HS70004@wilp.bits-pilani.ac.in	Rajdeep Biswas	
2020HS70010@wilp.bits-pilani.ac.in	Hussain Madraswala	
2020HS70001@wilp.bits-pilani.ac.in	Sugata Kar	

Part 1:

Write a shell script (list.sh) to display a directory listing based on the command line argument. Command line options are : File name : F, Date : D, Time: T and Permission (P)

Sample Input and Output:

$ ./list.sh F D T P

File name	date	time	permission
---	---	---	---
Filename1	date	time	permission
Filename2	date	time	permission

Total no. of files : <total number>
Total no of normal file : <number>
Total no of directory : <number>

$ ./list.sh D T P F

date	time	permission	File name

------	-----	----------	----------
date	time	permission	Filename1
date	time	permission	Filename2

Total no. of files : <total number>
Total no of normal file : <number>
Total no of directory : <number>

Part 2:

Find duplicate files. Suppose you're working in a project where software (or people) create lots of files, many of them duplicates. You don't want the duplicates: you want just one copy of each, to save disk space. Write a shell script "sameln" that takes a single argument naming a directory D, finds all regular files immediately under D that are duplicates, and replaces the duplicates with hard links. Your script should not recursively examine all files that are in subdirectories of D; it should examine only files that are immediately in D.

If your script finds two or more files that are duplicates, it should keep the file whose name is lexicographically first (for example, if the duplicates are named X, A, and B, it should keep A and replace X and B with hard links to A); however, it should prefer files whose name start with "." to other files (for example, if the duplicates are named .Y, .X, A, and B, it should keep .X and replace the others with hard links to .X).

If your script finds a file in D that is not a regular file, it should silently ignore it; for example, it should silently ignore all symbolic links and directories. If your script has a problem reading a file (for example, if the file not readable to you), it should report the error and not treat it as a duplicate of any file.

You need to worry about the cases where your script is given no arguments, or more than one argument and also be prepared to handle files whose names contain special characters like spaces, "*", and leading "–".