ID2090 : Introduction to Scientific Computing
Summer-2021

Assignment – 2

Instructions:

1. Each question carries 5 marks. Solutions to these questions can make use of shell commands, scripts, awk and sed.
2. Make a consolidated report of your submission as a tar file named after your roll number eg., ME20B001-Assn2.tar. When we untar this file, it should create a folder MM20B001-Assn2 which should contain your stuff including any images, screenshots, scripts, PDF files etc., Please try this yourself once and confirm before you upload.
3. Upload the tar file on moodle as per the deadline for the assignment. Late submissions are accepted but with a small penalty proportional to the delay. I shall decide the extent of this based on the actual submissions of the class.

[1] Use the $RANDOM shell variable that provides a random number between 0 and 32767 to generate a csv file that will contain marks for a set of 50 students. Scale or use mod operation to get the range correct. The roll number can be a serial number from 1 to 50. The marks in percentage shall be in three columns, namely, Q1 (max 25), Q2 (max 25) and EndSem (max 50). Name this csv file as "marks.csv". Using a separate script, run through these 50 rows to normalize the marks (topper is scaled to 100) and to assign grades in the following rubric: S for marks > 90; A for >80; B for >70; C for >60; D for >50; E for >40 and U for 40 or below. Print the output showing the original mark break up, normalized mark and the grade against each student.
PS: Since the marks are randomly generated, I expect that each of you will have a unique csv file and reasonably different grade sequence. I plan to check that out.

Output required: The two codes – one to create the csv and one to analye it to assign grades; the csv file itself; the output file showing the mark break up, normalized marks and the grades.

Application: When the calculations are simple but the number of rows is large (say >1 million), spreadsheets don't work. Scripts like this come of use and are fast as well as efficient. One doesn't always need a database or an expensive GUI for every task that involves processing a sequence of numbers.

[2] Create a make file that has the following behavior when invoked as given below.

make	Output the usage pattern as help
make list	Recursively list all files in the current directory larger than a certain size ie., n kilobytes
make small	Compress all those files listed as above.
make restore	Uncompress all those files listed as above back to their original form.

The value of n for the threshold size of files should be configurable using a shell variable $MODSIZE. Default value can be taken as 10000 (i.e., 10MB). You are welcome to store the list of files being processed in a temporary file to ease the process. Use a shell variable like $MODTMPFILE to hold the name of the file. Your script should exit elegantly if such a variable is not defined or if such a temporary file is not present or readable. You can test this script in a sample directory and you can have any random files from the internet to test the functionality. For the

compression, you can try commands like “compress”, “gzip” or “bzip2”. You need not include the test directory while uploading.

Output required: The code, screenshots showing its behavior.

Application: Our accounts in a supercomputer often come with a quota on the total amount of space used. We may need to reduce the footprint time to time to stay below the quota limit. Scripts like this will help in such situations readily. Also, such scripts can help identify the space hoggers in our system to keep our disc clean.

[3] Pick five commands or tools that people use in non-free environment and identify a suitable alternative from the free and open source ecosystem. Give documentary references (links) for someone to verify the information you provide. You should prefer those tools that are typically used by engineering community in the broad area of scientific computing.

PS: Please don’t try copying from others. I have scripts to report me percentage overlap.

Output: A text file containing the mapping of non-free to free tool and links to verify the same

Application: One should be aware of free and open source tools that can save the day when we need to solve a problem at hand.

--oo**oo--