Module 1: Bash scripting for parsing text files

Project Brief

In this project you will develop two short bash scripts to automate downloads and parse files.

Successful completion will demonstrate competence in basic bash scripting and awk programming. These skills are valuable in automating and simplifying manifold text and data processing tasks in scientific computing.

Deliverables

You should submit your scripts by creating a subdirectory called
`/class/mse404ela/sp26/<your_net_id>/Project1`
and copying your two completed scripts into that directory by **11:59pm on 2 February 2026.** ***Scripts must run on the EWS Linux machines.*** **Late submissions will not be accepted; let me know in advance if you will have difficulty with completion.**

I will give you feedback on the expectations listed below and for the overall script *usability, performance, clarity, and presence of useful in-script comments/documentation.*

It is **strongly advised** that students write the script using vi/emacs or a simple text editor on a machine running Linux or Mac OS X or via ssh to EWS). Scripts written on Windows machines often contain extraneous characters that make them non-portable to other environments.

Script 1: Parsing Quantum Espresso output files.

The output from a series of Quantum Espresso runs, at different energy cutoffs (we'll talk about that in the next module) from 10 Ryd to 80 Ryd in steps of 5 (10, 15, 20, etc.) are available at the URL:
`http://courses.engr.illinois.edu/mse404ela/Project1/qe-out.[Ecut]`
where `[Ecut]` is a number. Your script will need to do the following:

1. Download files using `wget` or `curl`; (*hint:* a `for` loop combined with `seq` can generate the sequence of numbers you'll need)
2. Output a table with the following columns: energy-cutoff, total-energy, CPU-time.
   The total energy is found on a line "! total energy", while the total CPU time is on a line starting `PWSCF`. The time should *just* be a number, not include the unit ("s"). (*hint:* `awk` will be helpful for parsing; you could either have an awk script that extracts both the total energy and CPU-time and outputs it at the end, or have two single scripts that save their output to variables and use `echo` to write out each line).

Script 2: Text analysis of a file.

You will write a script that takes a text file, and outputs a list of the 100 most common words, sorted by their appearance in the file. This can be done with a *single line* of script, appropriately piped together. You do not need to write your script on a single line, but you do not need to create any temporary files. There is an example text file, `/class/mse404ela/Bash-example/magnesium-alloying.txt` that will produce output that looks like this:

```
601 the
224 of
207 and
175 in
152 to
136 solute
131 for
115 a
 94 with
 84 is
```

and continues to

```
 14 not
 14 mn
 14 induced
 13 yield
 13 use
 13 pyramidal
 13 into
 13 directly
 13 crss
 13 approach
```

You will need to keep a few things in mind for this:

1. Split words with a hyphen into two words: "solute-fault" should be counted as "solute" and "fault".
2. Treat words with the same case identically ("Solute" and "solute").
3. Do not include digits. ("2", "50,000" etc.)
4. Your script should run as `./word-count.sh magnesium-alloy.txt` for the above example.

Hint: This can be done entirely *without* using `awk` or another programming language; you will need to find a program that translates characters, one for sorting, and another for counting up how often something occurs. With these daisy-chained together, the analysis is very fast.