# TEST CASES

| | Bash/Dash | simpsh |
|---|---|---|
| **Test Case 1** | cat pg98.txt \| \<br>sort \| \<br>tr A-Z a-z > output.txt 2> outerr.txt | ./simpsh --rdonly pg98.txt --pipe --pipe --wronly outputS.txt --wronly outerrS.txt --command 3 5 6 tr A-Z a-z --command 0 2 6 cat --command 1 4 6 sort --close 1 --close 2 --close 3 --close 4 --profile --wait |
| **Test Case 2** | cat pg98.txt \| \<br>uniq \| \<br>sort > output1.txt 2> outerr1.txt | ./simpsh --rdonly pg98.txt --pipe --pipe --wronly output1S.txt --wronly outerr1S.txt --command 3 5 6 sort --command 0 2 6 cat --command 1 4 6 uniq --close 1 --close 2 --close 3 --close 4 --profile --wait |
| **Test Case 3** | cat pg98.txt \| \<br>uniq \| \<br>tr A-Z a-z > output2.txt 2> outerr2.txt | ./simpsh --rdonly pg98.txt --pipe --pipe --wronly output2S.txt --wronly outerr2S.txt --command 3 5 6 tr A-Z a-z --command 0 2 6 cat --command 1 4 6 uniq --close 1 --close 2 --close 3 --close 4 --profile --wait |

For testing purposes, I am using the pg98.txt file the TA has posted, but I added a few more lines (wc -l pg98.txt :returns=> 71008) so that differences would show up more.

# RUN TIMES

| Test case | Bash | Dash | simpsh |
|---|---|---|---|
| **Test Case 1** | 0m0.001s 0m0.002s<br>0m0.471s 0m0.058s | 0m0.000s 0m0.000s<br>0m0.453s 0m0.053s | 0.00000s 0.00017s<br>0.47008s 0.05404s |
| **Test Case 2** | 0m0.001s 0m0.001s<br>0m1.331s 0m0.096s | 0m0.001s 0m0.001s<br>0m1.320s 0m0.087s | 0.00082s 0.00023s<br>1.32671s 0.08834s |
| **Test Case 3** | 0m0.001s 0m0.001s<br>0m0.885s 0m0.106s | 0m0.000s 0m0.000s<br>0m0.883s 0m0.090s | 0.00000s 0.00017s<br>0.87857s 0.08337s |

Note: the simpsh times should be prefaces with a 0m

# CONCLUSIONS

Generally speaking, they are all around the same in terms of performance. For larger files, based on the trends shown above, it is likely that simpsh is the most efficient shell, however, dash rivals it closely in many ways. For larger files, simpsh would be an efficient choice.