HW 1: Unix Basics Name: Francesconi

100 Points

2023 Spring

Questions

(9 points) What are the three main components of a Unix system, according to the notes? Give a short description of each.

|  |
| --- |
| 1. User Space: Defines the interfaces of the OS that are user accessible, programs are written with this space in mind without needing to understand deeper parts of the OS 2. Kernel Space: Operations of the OS that manage the interface between user actions and the hardware. 3. Hardware: Physical parts of the computer including input/output devices, peripherals, the CPU, memory, and network interface. |

(14 points) Consider the ls -l output below. Label each section indicated.

.- Directory?

| .------- Permissions .- Directory Name

| \_\_\_|\_\_\_ .----------- Owner |

v/ \ V (group) V

drwxr-xr-x 2 door scs 4096 2013-12-22 10:57 demo/

-rw-r--r-- 1 door scs 13454 2013-12-22 10:56 text.dat

^ \\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ ^

File size in bytes-----' | '- File Name

|

|

Last Modified -----------'

(16 points) Look up the which command in the manual pages (man which at the terminal), using either the WSL terminal or a lab machine terminal. Reading the description, what is the purpose of this command?

|  |
| --- |
| which prints the path to where a given command is stored. It does this by searching the PATH directory for executable files matching the given command. |

(10 points) Look up the tac command in the man pages. Describe its operation and give an example usage.

|  |
| --- |
| tac allows the user to enter text in the terminal or a file. tac writes the file to standard output in reverse so the last line in the file comes first. The user can add options for the command to interpret whitespace differently.  Ex:  File letters.txt contains:  A  B  C  D  E  F  tac letters.txt  F  E  D  C  B  A |

(15 points) What are the three guiding principles of the Unix design philosophy, according to the notes?

|  |
| --- |
| Write programs that do one thing and do it well. |
| Write programs to work together. |
| Write programs to handle text streams, because that is a universal interface |

(12 points) What are the primary purposes of *standard input*, *standard output*, and *standard error*?

|  |  |
| --- | --- |
| Standard Input (stdin) | The primary input stream for reading information printed to the terminal |
| Standard output (stdout) | The primary output stream for printing information & program output to the terminal |
| Standard error (stderr) | The primary error stream for printing information to the terminal that resulted from an error in processing |

(15 points) Consider the following command line with redirects:

$ grep PA < sample-db.csv 2> oops > sample-db.PA.csv

|  |  |
| --- | --- |
| What is the *output* file? | sample-db.PA.csv |
| What is the *input* file? | sample-db.csv 2 |
| What is the *error* file? | oops |

(9 points) Why is it necessary to have both *standard error* and *standard output*? Consider this example pipeline in your answer (goodfile exists but badfile does not):

$ cat badfile goodfile | wc

|  |
| --- |
| By separating standard error and standard output you ensure that errors are not piped into the next command as input. In this case, this ensures that the error produced by badfile is not run as input for wc. |