Name of C File: showFDtables.c

Overview:
This application offers comprehensive views of file descriptors (FD) that Linux system processes utilize.

Features
Per-Process View: Shows the file descriptors FD and PID opened by the user
System-Wide View: Displays file descriptors FD, PID, and Filename of the processes opened by the user.
Vnodes View: Lists FD and (vnodes) information.
Composite View: Combines all the above views into one comprehensive table.
Threshold Filtering: Highlights processes exceeding a specified number of file descriptors given by the user.

How did I solve the problem?
As it moves through the /proc directory, the program examines the file descriptors in /proc/[PID]/fd for each process. It collects information on inodes, pointing files, and file descriptor types, among other things and stores it in a CDT and Linked Lists. It also checks for user permissions to see which processes it can read and access, as well as skipping processes that are not accessible/relevant.

Functions Overview
getInfo(): Gathers file descriptor information (filename, FD, PID, Inode) from the system and stores it in a CDT created data.

printComposite(): Displays a comprehensive table of all file descriptors.

printPerProcess(): Shows the file descriptors FD and PID opened by the user, or if pid is given, only prints FDs with the given PID.

printSystemWide(): Outputs FD, PID, and Filename of the processes opened by the user, or if pid is given, only prints the above variables with the given PID.

printVNodes(): Lists FD and inode information, or if pid is given, only prints the above variables with the given PID.

printThreshold(): prints processes with file descriptors exceeding a set threshold stated by the user.

cleanUp(): Frees dynamically allocated memory to prevent memory leaks.

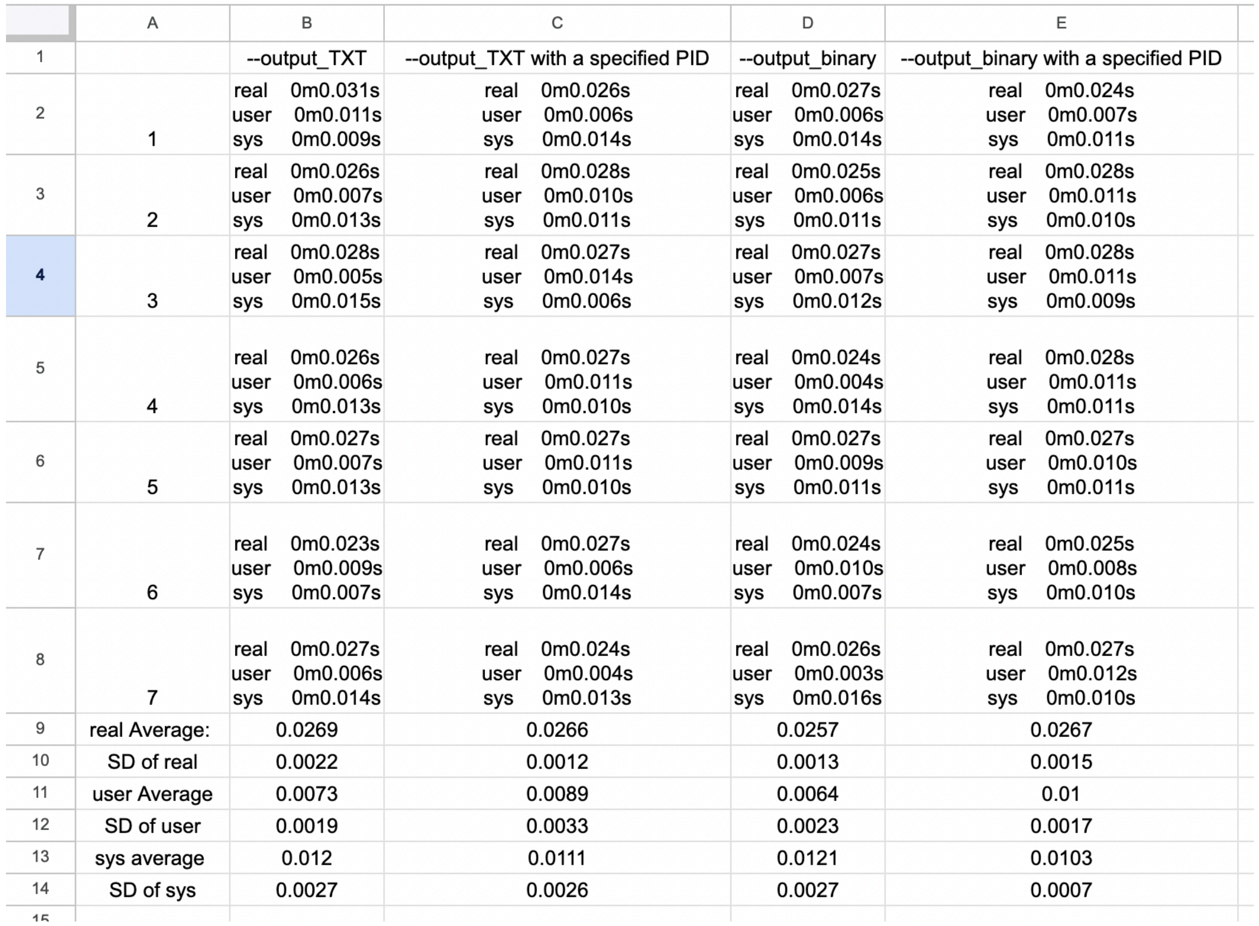
how to run the program:
Either run make OR write gcc -o showFDtables showFDtables.c in the command line,
Then, run ./showFDtables
With any of the following flags or combination of flags and it should output the expected results as shown in assignment handout and video

* Note: running ./showFDtables without any other arguments will give the same results as running ./showFDtables --composite. (a.k.a. It will only print the composite table).

Bonus Marks Section:
--output_TXT will print the composite table in text, ASCII, to compositeTable.txt and
--output_binary will print the composite table in binary to compositeTable.bin

Analysis on Binary output vs Text output:

Results from running --output_TXT for all PIDs:



	A	B	C	D	E
1		--output_TXT	--output_TXT with a specified PID	--output_binary	--output_binary with a specified PID
2	1	real 0m0.031s user 0m0.011s sys 0m0.009s	real 0m0.026s user 0m0.006s sys 0m0.014s	real 0m0.027s user 0m0.006s sys 0m0.014s	real 0m0.024s user 0m0.007s sys 0m0.011s
3	2	real 0m0.026s user 0m0.007s sys 0m0.013s	real 0m0.028s user 0m0.010s sys 0m0.011s	real 0m0.025s user 0m0.006s sys 0m0.011s	real 0m0.028s user 0m0.011s sys 0m0.010s
4	3	real 0m0.028s user 0m0.005s sys 0m0.015s	real 0m0.027s user 0m0.014s sys 0m0.006s	real 0m0.027s user 0m0.007s sys 0m0.012s	real 0m0.028s user 0m0.011s sys 0m0.009s
5	4	real 0m0.026s user 0m0.006s sys 0m0.013s	real 0m0.027s user 0m0.011s sys 0m0.010s	real 0m0.024s user 0m0.004s sys 0m0.014s	real 0m0.028s user 0m0.011s sys 0m0.011s
6	5	real 0m0.027s user 0m0.007s sys 0m0.013s	real 0m0.027s user 0m0.011s sys 0m0.010s	real 0m0.027s user 0m0.009s sys 0m0.011s	real 0m0.027s user 0m0.010s sys 0m0.011s
7	6	real 0m0.023s user 0m0.009s sys 0m0.007s	real 0m0.027s user 0m0.006s sys 0m0.014s	real 0m0.024s user 0m0.010s sys 0m0.007s	real 0m0.025s user 0m0.008s sys 0m0.010s
8	7	real 0m0.027s user 0m0.006s sys 0m0.014s	real 0m0.024s user 0m0.004s sys 0m0.013s	real 0m0.026s user 0m0.003s sys 0m0.016s	real 0m0.027s user 0m0.012s sys 0m0.010s
9	real Average:	0.0269	0.0266	0.0257	0.0267
10	SD of real	0.0022	0.0012	0.0013	0.0015
11	user Average	0.0073	0.0089	0.0064	0.01
12	SD of user	0.0019	0.0033	0.0023	0.0017
13	sys average	0.012	0.0111	0.0121	0.0103
14	SD of sys	0.0027	0.0026	0.0027	0.0007
15					

For all PIDs being printed:
We can conclude that printing the values to text/ASCII is slightly more expensive (takes more time from the real) than printing values in binary on average, by looking at the above averages.

For a specified PID:
When we specify the PID in the CLA, we can see that the averages show us the opposite results: --output_TXT takes less time to run and execute than --output_binary.