CST 357/457
Michael Ruth
Homework #4
Due: 10/30 (by midnight)

Undergraduates & Graduate Students:

For this assignment, you will be writing two C programs. You will create a folder which contains the programs and then zip the folder for submission to blackboard. Please be sure to document your program appropriately. You must rename the file using your last name followed by your first name followed by a dash followed by the assignment name (For instance, this is homework4, so I'd turn it in as RuthMichael-Homework4.zip)

(50%) Part I: Write a program that reads a given directory (on the command line) and searches for files that match using some or all of the following (all parameters MUST be given on the command line). Note that the output of this program should be the full path of all files that match the given criteria.

1. **(10%)** If a user enters –name use the next argument (after name) as the name of the file or directory you search for
2. **(10%)** If a user enters –R the search should be recursive (not otherwise)
3. **(10%)** If a user enters –uid use the next argument (after uid) as the UID to search for (only select files owned by the given uid)
4. **(20%)** if a user enters –type use the next argument (after type) as the type of file to search for (only include files of that given type – for instance regular files, directories, block files, etc.)

(50%) Part II: Write a program that recursively prints out all the full paths of all directories and files in a given directory that uses forks (the children) to do the recursive methods.

1. **(10%)** Each process should print out their directory and all regular files in their directory
    a. **Hint: I did this in class!**
2. **(10%)** All (children and parent) should contain a program that runs upon its exit indicating that it is dying (indicate directory).
3. **(20%)** If a directory has a subdirectory, fork a child to recurse into the subdirectory. Note that means the parent stays in the current directory and the child does the subdirectory call.
4. **(10%)** Maintain a counter and make sure not to leave any zombies around… each parent should wait for its children before exiting.

Graduate Students (or UG for extra credit + 50):
Part I: (15%) Create a mechanism for the search to use the size of the file: I want to see -lt, -gt and -eq (you can use all three at the same time). User should enter lt for less than, gt for greater than, or eq for equal to and for each of these the following parameter provides the value (file size)

Part II: (15%) Create a mechanism for search to use the depth of the search (how many subdirectories deep?). User should enter -depth followed by a depth argument (0 is non-recursive, 1 is one subdirectory deep, and so on)

Part III: (20%) Consider the performance of Part II. Create a version that doesn't fork and compare the time it takes to run both… you should run each program several times to ensure accurate results. The

major answer you want to ask “Does adding the fork make for better performance?” Explain your answer using your results in a text file “part3.txt”

Note:

- If you don’t turn in the file in the correct format (ZIP), I will take **10 points** off the total score.
- If you don’t name the files or methods correctly, I will take **10 points** off the total score.
- You SHOULD NOT need to use material learned outside this class. If you choose to do so, you will earn NO points on this homework.
- If someone else turns in your homework, you both get a zero whether or NOT you know each other.
- Do NOT try to do this last minute!