Project: BackItUp!
CS 460

Introduction

This assignment should be programmed entirely in C and must compile and run on the lab's Linux environment or similar. Copying & pasting code from anywhere is considered cheating and results in an automatic F.

WARNING: This assignment involves the creating and overwriting of files. It is very important that you are careful with your testing! It is very easy to lose your program!

Behavior

Your goal for this assignment is to create a file backup system called BackItUp! The program, named `BackItUp`, works at a directory level and should behave as follows:

1. `BackItUp` should create a directory `.backup/` in the current working directory if one does not already exist.

2. `BackItUp` should then create a copy of all the regular files in the current working directory into the `.backup/` directory. All the backup files should have `.bak` appended to their name.

3. If a backup file already exists, your program should compare the last modification times of the original file and the backup file to determine if a backup is required. If the existing `.bak` backup file is older, then your program should overwrite it. Print out a warning when doing so. Otherwise, you should not overwrite the file, and you should notify the user that the file is already the most current version.

4. Your program should allocate a new thread to copy each file. Therefore, if the current working directory contains five files, then your program should create five threads to backup each file. Of course, your program should not terminate until all the threads have finished.

5. You must recursively handle subdirectories. In each subdirectory, you must again spawn one thread per file. (Create a new instance of the `.backup/` directory in each subdirectory.)

6. If the user invokes `BackItUp` with the optional `-r` (restore) argument, then it should restore all backup files in the *.backup* directory by copying them to the current working directory. (Be careful not to clobber `BackItUp`!). Again, it should not restore any files that have a later modification time than the backed up copy. Like before, the restore function should create a separate thread for each file being copied.

Notes: You will want to check out the following Unix system calls for C: `stat()`, `opendir()`, and `readdir()`. The command line tool `touch` may also prove helpful.

The file `example.pdf` (in Canvas), provides a sample run of `BackItUp`.

Deliverables

- You must create a Makefile to compile your source and name the binary `BackItUp`
- You output should be similar to those in the sample interaction.
- Please submit in the Canvas your source code before 11:59pm on the due date.
- Please include all files necessary for a successful build as a zip file.

Grading

Your submission will be graded by compiling and running it.

- Please make sure your source code can compile. Absolutely no credit if it does not compile.
- Please make sure your programs run on standard Linux servers.
- Please don't include the binary files. Do a make clean before submission.
- Please don't leave out any files!
- Please don't modify any files you don't need to!
- Please don't send us the meta-information from your revision control system!

Scoring Data:

Category	Value
Create directory if it does not exist	5 pts
Copy all regular files	5 pts
Correct handling of old backups	10 pts
Proper threading	10 pts
Recursive subdirectory handling	10 pts
Correct restore functionality	25 pts
Makefile	5 pts
Code is free of memory / threading errors	30 pts