#####READ ME#####

Project 1: A Multiprocess Sorter

-sorter.c

-mergesort.c

-Sorter.h

This program takes in 0, 1, or 2 directories as input, and sorts all properly formatted .csv files inside.

All inputs must be in the form:

./sorter -c FIELD [-d INPUT] [-o OUTPUT]

./sorter -c FIELD [-d INPUT]

./sorter -c FIELD [-o OUTPUT]

./sorter -c FIELD

where field may be:

color

director_name

num_critic_for_reviews

duration

director_Facebook_likes

actor_3_facebook_likes

actor_2_name

actor_1_facebook_likes

gross

genres

actor_1_name

movie_title

num_voted_users

cast_total_facebook_likes

actor_3_name

facenumber_in_poster

plot_keywords

movie_imdb_link

num_user_for_reviews

langauge

country

content_rating

budget

title_year

actor_2_facebook_likes

imdb_score

aspect_ratio

movie_facebook_likes

And

INPUT and OUTPUT may be relative or direct paths, with quote circumfixes optional.

All outputs are marked with a "-sorted-FIELD".

If an output directory does not exist, a new directory is created to fit them.

--PROJECT 1 INFO---

We use a loop to handle forking. The loop works by opening a directory, and going to the next item (skipping . and ..), and forking on each file to process it. In the case of directories, the forked child reopens the new directory and runs through the loop. In the case of .csv files, they are sent into the sorter.

The output of our sorter was slightly changed, instead of using STDOUT, we instead write to a newly created file buffer.

Waits are handled by counting the number of valid children of the starting directory, and the children of every directory below that, then waiting for each child. These subdirectories also return their number of children, which is how we count the value of PIDS.

We reject all improperly formatted arguments.

We skip all files with a "-sorted-" in the name.

Sorter.h contains the struct needed to store our movie data,

as well as a number of function declarations. The structs and

functions are the ones that are called in both files, sorter.c and mergesort.c

like mergesort or mergesortHelper, or ones that could be useful for

other projects, such as trim, compareStrings, or printCSV.

---PROJECT 0 INFO---

Design:

We chose to use an array of struct pointers to store our

data, as this made for an easier time swapping the arrays.

We chose to use strcasecmp, as using strcmp put lowercase

letters after all uppercase letters, which put them

out of dictionary order place.

Memory leak free according to valgrind.

We reject all improperly formatted arguments.

All empty number fields are set as "-1", to differentiate

between an empty input value, and a value of 0.

All empty strings are left as is.