COM S 327, Fall 2022
Programming Project 1.06
CSV parsing

We'll be parsing in a number of data files that describe pokémon and their moves. The database containing these files is available on Pyrite under `/share/cs327/pokedex`. You may copy this entire database to a local location if you would like using scp, for example:

`scp -r netid@pyrite.cs.iastate.edu:/share/cs327/pokedex /local/path/`

where `netid` is your net id and `/local/path/` is the local path to the location where you want your files.

When loading the database, your program should look in at least two and optionally three places for the files, only failing if none of those locations contain the database. Your program should first look under `/share/cs327`. Failing that, it should look under `$HOME/.poke327/`. And, optionally, it may look in a third place of your choosing. For the second location, use `getenv()` to resolve the value of the `HOME` environment variable.

NOTE: Your program **must** search for the database under `/share/cs327` and `$HOME/.poke327/` even if you have a local copy somewhere else. This is so it will load the TAs' copies without them having to modify your sources.

ANOTHER NOTE: DO NOT include your database under your source tree. If you submit a solution that has the database included within it, you will be docked 3 points (out of 10)! The database is too large for the teaching staff to pass around 100+ copies of it conveniently.

Your program should open the database in the first location where it is found, or print an error message and terminate upon failure if the database is not found.

The CSV files are located under `pokedex/pokedex/data/csv/`. We will need to parse the following files:

- `pokemon.csv`
- `moves.csv`
- `pokemon_moves.csv`
- `pokemon_species.csv`
- `experience.csv`
- `type_names.csv`
- `pokemon_stats.csv`
- `stats.csv`
- `pokemon_types.csv`

For each parsed file, create a struct or class to hold the data type, then create an array of those to hold all of the data.

Use the value `INT_MAX` as a placeholder for empty cells in the CSV files.

`type_names.csv` contains some non-ascii characters. You are welcome to fully parse the file—and to use non-English words—but if you are only interested in English, you can simple read whole lines and discard the irrelevant lines from this file. Only those lines wherein `local_language_id` is 9 are in English.

Modify your game to take a single command line parameter, one of `pokemon`, `moves`, `pokemon_moves`, `pokemon_species`, `experience`, `type_names`, `pokemon_stats`, `stats` or `pokemon_types`. Your program should parse the specified file, print its contents to standard output (do not initialize curses), and exit. The whole game will still be compiled and linked, we're simply exiting before running it.

The format of the files is multiple lines of values separated by commas. The first line in each file defines the fields in the file (and make good choices for field names in your data structures). Some fields will be empty, denoted by a comma followed immediately by a comma. In printing out your parsed data, do not print the `INT_MAX` that you use internally to denote empty fields.

All new code should be written in C++.