The company you are working for, Insurance Aggregator Inc, currently has multiple CSV files that contain information about partners that we work with. These files are sent to us on a daily basis from our partners, and we need a way to combine all of that information into one place to help out our friends in marketing and sales.

Given an input list of CSV files, convert the multiple files into a single CSV that matches the output schema provided below. Some of the data we get from these partners is missing or doesn't fit with the schema, so in those cases we want to print out that there has been an error along with the offending row instead of including the row in the output file. You should try to convert as many of the rows as possible (including ones where it's possible to convert a valid string to a number for instance), leaving out any columns our partners give us that are not in our schema.

Our partners are nice enough that they made their column names on the input files match those in our schema, and they always send the columns in the same order. We have provided two example input sheets to help with testing and debugging. As an additional note, the maximum number of rows we will ingest at once is in the tens of thousands.

Your final output should be a well formatted, unified CSV file written to the local directory. It should contain all of the rows from the input files that matched the expected schema.

What we are looking for

- A github repo with your solution and instructions on how to run it
- Well written, readable code that follows standard programming practices
- Some unit tests so that we know we are deploying good code
- Python is preferred, but we will accept submissions in comparable backend-oriented (Java, Ruby, C# etc) languages.
- Your thoughts on your solution in a README as well as anything you would do in the future to expand on the project (more tests, nice to haves, etc), and any other thoughts you may want to share.

Output Schema

Provider Name	CampaignID	Cost Per Ad Click	Redirect Link	Phone Number	Address	Zipcode
Type: String, Non-Nullable	String, Non-Nullable	Type: Float, Non-Nullable	Type: String, Non-Nullable	Type: String, Nullable	Type: String, Non-Nullable	Type: String, Non-Nullable

Notes on our CSV format

- The goal of this assignment is not to catch you out on file format technicalities. We are being specific in this section to help answer any questions you might have, not to be pedantic about small mistakes.
- All input files will have valid UTF-8 encoding
- The input files will generally conform to the CSV standard as defined in https://tools.ietf.org/html/rfc4180.html, with one exception:
    - We use Unix style line endings, i.e., rows are delimited with a single newline (**\n**) character instead of CRLF (**\r\n**)
- Your program should accept files in this input format, but it MAY be less strict with the files that it accepts. We are only concerned with validating the contents of the input files, not their encodings.
- Your program MAY quote fields in the output file even if they could be parsed without quotes, as long as the output remains valid CSV.