Preliminary Documentation for Buffer.h

Overview:

The Buffer.h file defines a class and a struct for handling Zip Code data read from a CSV file. It includes a Buffer class that manages reading the file, storing records, and organizing them by state. The ZipCodeRecord struct stores information like Zip Code, state ID, latitude, and longitude.

Purpose:

- **Store Zip Code Data:** The ZipCodeRecord struct holds information for each Zip Code, including the Zip Code, state, and geographic coordinates.
- **Process CSV Files**: The buffer class reads and parses CSV files, storing Zip Code data.
- **Organize by State:** The data can be grouped by state using the get_state_zip_codes function, creating easy access to state-specific data.

Key Components:

ZipCodeRecord Struct

- Holds Zip code details: zip_code, state_id, latitude, and longitude

Buffer Class

- **read_csv():** Read CSV file and store data.
- **get_state_zip_codes() const:** Return a map of state IDs to their respective Zip Code records.
- **parse_csv_line(const std::string& line) const**: Parse a line from the CSV file into a ZipCodeRecord.

Preliminary Documentation for Buffer.cpp

Overview:

The Buffer.cpp file includes the implementation of the Buffer class and the ZipCodeRecord struct that were defined in Buffer.h. It handles reading and processing Zip Code data from a specific CSV file called us_postal_codes.csv. The file stores the Zip Code records and helps organize them by state.

Purpose:

- Store Zip Code Data: The ZipCodeRecord struct holds information for each Zip Code, such as the Zip Code itself, the state it belongs to, and its location (latitude and longitude).
- Read CSV Files: The Buffer class reads data from the us_postal_codes.csv file, processes it, and saves the Zip Code information in a list.
- Organize by State: The data can be organized by state using the get_state_zip_codes method, making it easy to access Zip Code records for each state.

Key Components:

- ZipCodeRecord Struct: Contains details about a Zip Code: zip_code, state_id, latitude, and longitude.
- Buffer Class: **bool read_csv();** Reads the us_postal_codes.csv file and stores the Zip Code data.
- **std::map<std::string, std::vector<ZipCodeRecord>> get_state_zip_codes() const;** Returns a list of state IDs and their corresponding Zip Code records, organized by state.
- **ZipCodeRecord parse_csv_line(const std::string& line) const;**
  Takes a line from the CSV file and turns it into a ZipCodeRecord, pulling out the important information.

Preliminary Documentation for CSVProcessing.h and CSVProcessing.cpp

Overview:

The CSVProcessing files (CSVProcessing.h and CSVProcessing.cpp) define a class responsible for processing the Zip Code data stored in the Buffer class. It handles sorting the data, creating a header for the output CSV file, and generating the final CSV output with the sorted and filtered data.

Purpose:

- **Sort and Organize Data**: The class sorts the buffer data, organizing Zip Codes by state and identifying the extreme points (easternmost, westernmost, northernmost, southernmost) for each state.

- **Generate CSV Header**: Creates a header row for the output CSV file.
- **Produce Final CSV Output**: Generates the final CSV output containing only the required data in the specified format.

Key Components:

CSVProcessing Class

- **std::map<string, std::vector<ZipCodeRecord>> sortBuffer()**:
    - Sorts the buffer data into a hashmap.
    - States are automatically sorted alphabetically by their IDs.
    - For each state, it identifies the Zip Codes with the extreme coordinates in each direction.
    - Returns a hashmap with state IDs as keys and vectors of extreme Zip Codes as values.
- **void addHeader(std::string& file_name)**:
    - Adds a header row to the specified CSV output file.
    - The header includes columns for State, Easternmost, Westernmost, Northernmost, and Southernmost Zip Codes.
- **bool csvOutput(std::string& file_name)**:
    - Parses the sorted hashmap from sortBuffer().
    - Writes the final CSV output, including only the data specified in the header.
    - Returns true if the output was successfully written, false otherwise.
- **void printZipCodeRecord(const ZipCodeRecord& record)**:
    - Utility function to print the details of a ZipCodeRecord.
    - Only used for debugging and verification.

This class serves as the main processing unit for the Zip Code data, bridging the gap between the raw data in the Buffer class and the final, organized output in CSV format.

Preliminary Documentation for main.cpp

Overview:

main.cpp is an application designed to use the methods in the other classes. It is how use of the program should be handled. It is kept simple for easy augmentation for future projects.

Purpose:

- Run **addHeader** to generate the header row for the output.
- Run **csvOutput** to display the output that is generated.

Key Components:

- **CSVProcessing.h:** Contains the **addHeader** and **csvOutput** functions that will be used.
- **buffer.h:** Dependency for **CSVProcessing.h**