ZipCode1.0

Generated by Doxygen 1.13.2

Chapter 1

ZipCode1.0

1.1 Zip Code Data Processor

1.1.1 Description

This program reads a CSV file containing ZIP code information and processes it to generate a state-wise summary of extreme ZIP codes (Easternmost, Westernmost, Northernmost, and Southernmost). The results are stored in an output file named `state_zip_summary.csv`.

1.1.2 Features

- Reads ZIP code data from a CSV file.
- Extracts relevant information such as state, latitude, and longitude.
- Determines the Easternmost, Westernmost, Northernmost, and Southernmost ZIP codes for each state.
- Saves the processed results to a CSV file (`state_zip_summary.csv`).
- Includes error handling for missing or malformed data.

1.1.3 Files in the Project

- **`main.cpp`** - The main program that initializes and runs the data processing.
- **`CSVBuffer.h`** - The header file defining the `CSVBuffer` class.
- **`CSVBuffer.cpp`** - Implementation of the `CSVBuffer` class.
- **`zip_codes.csv`** - Sample input CSV file containing ZIP code data.
- **`state_zip_summary.csv`** - Output file storing the processed data.
- **`README.md`** - This documentation file.

1.1.4 Compilation Instructions

1.1.4.1 Using g++ (Linux/macOS/Windows with MinGW)

To compile the program, run:

```
g++ -o myProgram main.cpp CSVBuffer.cpp
```

This command generates an executable file named `myProgram`.

1.1.5 Running the Program

Once compiled, you can run the program as follows:

```
./myProgram
```

On Windows (if using MinGW):

```
myProgram.exe
```

The program will prompt for a CSV filename. Enter the correct path to `zip_codes.csv`. Once the program loads the CSV file into memory, the user is prompted to choose a field from which to sort the data. An invalid choice will be sorted by default, state. The program will prompt for a CSV output filename. Enter an .csv output filename.

1.1.6 Expected Output

The program generates a CSV file name based on user input, which contains:

```
State, Easternmost, Westernmost, Northernmost, Southernmost
NY, 10001, 14905, 10598, 10002
MA, 01001, 02703, 01350, 02535
...
```

1.1.7 Error Handling

- If the CSV file is missing or unreadable, an error message will be displayed.
- If a row contains invalid data, it will be skipped, and a warning will be logged.

1.1.8 Author

Cha Vue, Sofia Hoffman, Alexander Miller, Zoljargal Enkhbayar, Yohannes Niguesse, Fatha Abdi

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Class Documentation

4.1 CSVBuffer Class Reference

`#include <CSVBuffer.h>`

Public Member Functions

- CSVBuffer (const string &filename)
  Constructor that opens a CSV file.
- const vector< ZipRecord > & getRecords () const
  Retrieves all records.
- void generateStateTable () const
  Generates and prints a table of extreme Zip Codes for each state.

Private Member Functions

- void loadRecords ()
  Loads records from the CSV file into memory.

Private Attributes

- ifstream file
- vector< ZipRecord > records

4.1.1 Constructor & Destructor Documentation

4.1.1.1 CSVBuffer()

```
CSVBuffer::CSVBuffer (
            const string & filename)
```

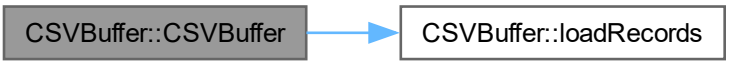
Constructor that opens a CSV file.

Constructor that opens and reads a CSV file.

Parameters

filename	Name of the CSV file.

Here is the call graph for this function:



4.1.2 Member Function Documentation

4.1.2.1 generateStateTable()

```
void CSVBuffer::generateStateTable () const
```

Generates and prints a table of extreme Zip Codes for each state.

Here is the caller graph for this function:

4.1.2.2 getRecords()

```
const vector< ZipRecord > & CSVBuffer::getRecords () const
```

Retrieves all records.

Returns

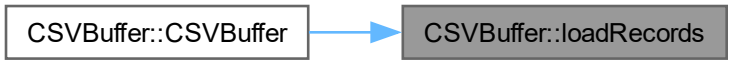
Vector of ZipRecord structures.

4.1.2.3 loadRecords()

```
void CSVBuffer::loadRecords ()  [private]
```

Loads records from the CSV file into memory.

Here is the caller graph for this function:



4.1.3 Member Data Documentation

4.1.3.1 file

```
ifstream CSVBuffer::file  [private]
```

4.1.3.2 records

```
vector<ZipRecord> CSVBuffer::records  [private]
```

The documentation for this class was generated from the following files:

- CSVBuffer.h
- CSVBuffer.cpp

4.2 ZipRecord Struct Reference

```
#include <CSVBuffer.h>
```

Public Attributes

- int zipCode
- string placeName
- string state
- string county
- double latitude
- double longitude

4.2.1 Member Data Documentation

4.2.1.1 county

```
string ZipRecord::county
```

4.2.1.2 latitude

```
double ZipRecord::latitude
```

4.2.1.3 longitude

```
double ZipRecord::longitude
```

4.2.1.4 placeName

```
string ZipRecord::placeName
```

4.2.1.5 state

```
string ZipRecord::state
```

4.2.1.6 zipCode

```
int ZipRecord::zipCode
```

The documentation for this struct was generated from the following file:

- CSVBuffer.h

Chapter 5

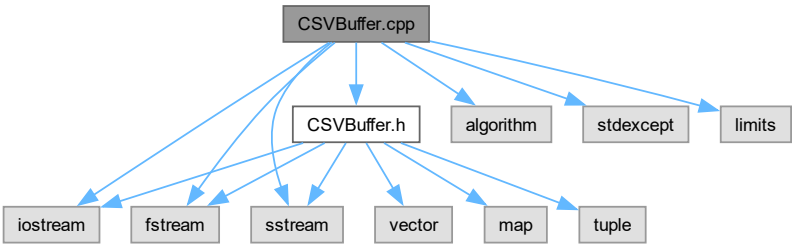
File Documentation

5.1 CSVBuffer.cpp File Reference

Implementation of CSVBuffer class.

```
#include "CSVBuffer.h"
#include <iostream>
#include <fstream>
#include <algorithm>
#include <sstream>
#include <stdexcept>
#include <limits>
```

Include dependency graph for CSVBuffer.cpp:



5.1.1 Detailed Description

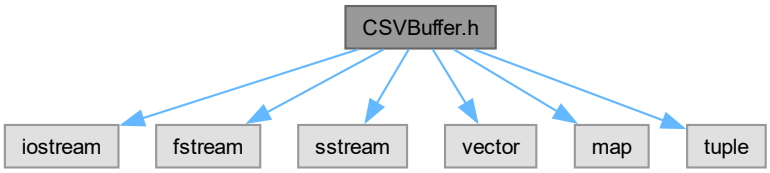
Implementation of CSVBuffer class.

5.2 CSVBuffer.h File Reference

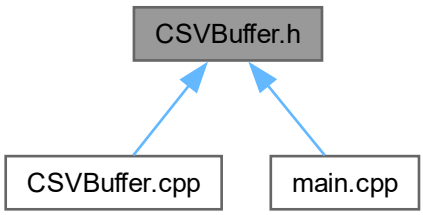
Defines a buffer class for handling CSV data.

```
#include <iostream>
#include <fstream>
#include <sstream>
#include <vector>
#include <map>
#include <tuple>
```

Include dependency graph for CSVBuffer.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct ZipRecord
- class CSVBuffer

5.2.1 Detailed Description

Defines a buffer class for handling CSV data.

This class reads a CSV file and allows extraction of structured data.

5.3 CSVBuffer.h

Go to the documentation of this file.

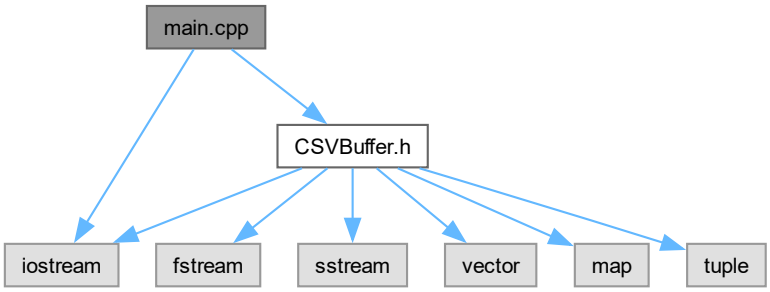
```
00001
00004
00005 #ifndef CSVBUFFER_H
00006 #define CSVBUFFER_H
00007
00008 #include <iostream>
00009 #include <fstream>
00010 #include <sstream>
00011 #include <vector>
00012 #include <map>
00013 #include <tuple>
00014
00015 using namespace std;
00016
00017 struct ZipRecord {
00018     int zipCode;
00019     string placeName;
00020     string state;
00021     string county;
00022     double latitude;
00023     double longitude;
00024 };
00025
00026 class CSVBuffer {
00027 private:
00028     ifstream file;
00029     vector<ZipRecord> records;
00030     void loadRecords();
00031 public:
00034     CSVBuffer(const string& filename);
00035
00038     const vector<ZipRecord>& getRecords() const;
00039
00041     void generateStateTable() const;
00042 };
00043
00044 #endif // CSVBUFFER_H
```

5.4 main.cpp File Reference

Main application to process Zip Code data from CSV file.

```
#include "CSVBuffer.h"
#include <iostream>
```

Include dependency graph for main.cpp:



Functions

- int main ()

5.4.1 Detailed Description

Main application to process Zip Code data from CSV file.

Uses CSVBuffer to generate a state-wise table of extreme Zip Codes.

5.4.2 Function Documentation

5.4.2.1 main()

```
int main ()
```

Here is the call graph for this function:

5.5 README.md File Reference

Index