Group 4 Preliminary Design Document | Project 2 | Created by: Tristan Adams, Andrew Clayton

Part I:

- Ensure Project 1 works on row-randomized CSV file
- Create length-indicated CSV files (for both randomized and ordered)
    - We used ChatGPT to create a C++ program to convert files from CSV to length-indicated. We had to prompt ChatGPT a couple of times to get it right, since it originally did not count a comma added after the record length and thus all the counts were off.
- Modify Buffer class to accept length-indicated CSV files
- Create a new Buffer class data file header record
- Repeat Project 1 with length-indicated files
    - This was not true, as project 2 did not end up involving sorting by extrema at all

Part II

- Use a hashmap to handle the primary key index that can be used to process and display the zip code data.
    - Key: zipcode Value: Byte-offset for that zipcode's data
        - Hashmap can then just access the specified zip code, and go to offset
    - The input zip code may not be in the file
    - We ended up spending 3 hours trying to get the byte offset to work and ended up having to just store instances of the buffer class in the index to get the output we needed. This is horribly space inefficient and breaks one of the rules for the project but we could not figure out what was wrong with our algorithm in time.
- Program needs to be able to run with command-line arguments to display more data, and to specify the zipcodes to provide information on
    - This ended up being easy and was just baked into the main() function call.
- Write doxygen tags using Chat GTP and write hand written comments as the code is made.
- Once all of the other parts are done, write a user guide.

  Overall we got very close to completing all of the requirements but could not get the index's algorithm to find the right spot in the data file. Since we could not get the index to work properly the code just creates a new index every time as the data being stored in the index is not a type that can be read back to a file. We will need to fix this runtime error before the next project so this can be used during project 3.

Header record Architecture:

- Work on this method alongside the rest of the code being developed. Work on it in iterative stages.
    - This was not a method as I originally thought and we ended up making a whole new class to handle the more complex header record.

- Start out hardcoding data but make it more flexible as the program is developed.
    - All of the data fields for the header record were flexible and editor with the new class that was made. The only thing it could not handle is the addition of more fields.

Submission requirements:

- User guide (.txt)
- Design document
- Test document
- CSV files (standard CSV, randomized, length-indicated)
- Length-indicated data-files files (.txt)
- Primary key index files
- Doxygen PDF
- .cpp and .h source code files *(zip the source files)*
- (type)script files generated to demonstrate the testing and running of the application program for Part I and Part II respectively

ChatGPT:

- This was used to bug fix small things when we were stumped(moderate success)
- Attempted to get it to fix our index run time error(no success)
- Used to generate the file that converted the CSV's to length indicated(100% success)