EXPLAINATION

In summary, program reads market data from multiple files, divides the data into five-minute intervals, processes the data concurrently using multiple threads, sorts the selected data, and writes it to an output file.

STEPS

Code is a C++ program that processes market data from multiple files and writes the selected data to an output file. Here's a breakdown of its functionality:

1. The code includes necessary header files for input/output, string manipulation, threading, and time-related operations.
2. It defines a structure called **MarketDataEntry**, which represents a single entry of market data, including fields like timestamp, symbol, price, size, exchange, and type.
3. The function **ReadMarketDataFromFile** reads market data from a file. It opens the file, reads each line, and parses the data to create **MarketDataEntry** objects. The parsed data is stored in a vector and returned.
4. The function **WriteMarketDataToFile** writes market data to a file. It opens the file in append mode and iterates over the input vector of **MarketDataEntry** objects, writing each entry to a new line in the file.
5. The function **CompareMarketDataEntries** is a comparator function used for sorting market data entries based on their timestamp and symbol. It compares two entries and returns a boolean value indicating their relative order.
6. The function **ConvertTimestampToMillis** converts a timestamp in the format "YYYY-MM-DD HH:MM:SS.SSS" to milliseconds since the Unix epoch.
7. The **main** function is the entry point of the program.
    - It defines a vector **fileNames** that stores the names of the input files to be processed.
    - It specifies the maximum number of threads to be created (**maxThreads**) and the chunk size for dividing the files (**chunkSize**).
    - It creates a mutex (**mtx**) for thread synchronization and sets the output file name (**outputFile**).
    - The input file names are divided into chunks, stored in a vector of vectors (**fileChunks**), to be processed concurrently.
    - The start and end timestamps for intervals are defined using **startTime** and **endTime**. We can safely assume that the data vendor provides data for a specific duration which is known
    - The start and end timestamps are converted to milliseconds using **ConvertTimestampToMillis**.

- Five-minute intervals are created using a loop and stored in a vector (**intervals**). Interval duration can be changed according to data diatribution
- For each interval, the program creates a vector **marketData** to store the selected market data.
- It iterates over the file chunks and creates threads to process the market data concurrently. Each thread calls the **ProcessMarketData** function with the appropriate parameters.
- The **ProcessMarketData** function reads the market data from each file in the chunk, selects the data within the given interval, and adds it to the **marketData** vector.
- After all threads have finished processing, the **marketData** vector is sorted based on the timestamp and symbol using the **CompareMarketDataEntries** function.
- The sorted market data is written to the output file using the **WriteMarketDataToFile** function.

8. Finally, the program returns 0 to indicate successful execution.

DIAGRAM: