Prove that it´s thread safe:

For our event class, we use a final internal state so They are immutable, and can´t be changed.

For our EventStore we use the try catch available in java to avoid concurrent modifications.

Code explenation:

For this implementation we use the LinkedList and iterator that’s already available in java to store and run over our events. I Decided to use the java list and iterator because lot of the functions that we had to use in this challenge are like the ones that java give to us, or very similar, so I just need to adjust somethings. Like the current function, that only uses the get function that java give to us to return the current event plus a integer with the index of the current position.

MoveNext:

In this function we update the index if the maxsize is bigger than zero

Remove:

We check if there’s event´s to remove, if true we remove the current event and we update maxsize and indexc

Query:

We create a LinkedList copy , a event to receive the next element on the list and a iterator to run over the list and check if there’s any event with the same parameter type as the one passed with the function and if the timestamp is bigger than startime and smaller than endTime. If all those conditions are true we add these event to the copy list and return a Iterator from the copy list

Current:

We pass the indexc which is the position where we are on the list now to the get function already available in java that return us the event in that position

RemoveAll:

In this function I create an iterator to run over the list and an event, to receive the next event on the list, so I check if the event type is equal the type passed as a parameter to the function, if true remove the event on the list and update the indexc and maxsize