std::unordered\_map<std::string, BSTSet<FlightSegmentWrapper>> m\_flightMap;

load\_flight\_data

If there are N airports, and if each airport is associated with F flights on average, then the file contains NF lines, and each line on average has C characters. For each line:

* reading: O(C)
* finding the source airport of a flight segment in the unordered\_map: O(1)
* inserting the flight segment into the BSTSet of that airport: O(logF)

And this is executed NF times.

So the big-O of loading is O(C+1+logF)\*NF=O(NFC+NFlogF)

find\_flights()

If there are N airports, and if each airport is associated with F flights on average, then:

look up the source\_airport in the unordered\_map: O(1)

call find\_first\_not\_smaller() in the associated BSTSet: O(logF)

call get\_and\_advance(), at worst it visits each node once: O(F)

push the flight segment into the vector: O(1)

SO the big-O of finding the flights is O(1+logF+F)=O(F)