# Algorithms

* With 50,000,00 record at most, the file’s size maximum is 1,5GB
* With data’s size is 1,5GB, we can load all data to memory
* With a larger file will have another approach ( using merge sort for sorting file’s data)
* After loading data, with each phone number, we need to know the whole record assigned with this phone number. To do this, we can use HashMap to map a record with key is phone number.
* After separate each record into sets with key is phone number, with each phone number set, we sort the set with order of active date.
* We will find active date with each phone number by setting the set’s first record to “result”. Go thought set and update “result”:
* If result is the last record, return result.activeDate
* If result.deactiveDate=nextRecord.activeDate, update result. deactiveDate=nextRecord. deactiveDate
* If result.deactiveDate < nextRecord.activeDate, update result.activeDate=nextRecord. activeDate

# Time complexity and memory complexity

## Time complexity

* To load data we have to go through every record, read data and store to array. It takes O(n) time.
* After having array of record, we will separate records to different sets with key is phone number, because we use HashMap and we need iterate through each record so we need O(N) time.
* With each set of phone number, we sort data by activation date, it takes O(milogmi) time with mi is the number record of mth phone number.
* We have to go through each sorted array and update result, it will take O(mi ) time
* Write data to output file take O(m) time where m is number of phone number

Totally, time complexity:

O(n) + O(n) +(milogmi) + O(mi) ) + O(m)

## Memory complexity

* To load record into memory it’ take O(n)
* Mapping record to HashMap take O(n)
* Sorting record data by using quicksort take O(mj) with mj is the largest set
* With each set, it’s take O(1) to find “result”
* Storing result take O(m)

Totally, memory complexity:

O(n) + O(n) + O(mi) + O(1) + O(m)