Problem 635: Design Log Storage System

Problem Information

Difficulty: Medium
Acceptance Rate: 59.23%
Paid Only: Yes
Tags: Hash Table, String, Design, Ordered Set

Problem Description

You are given several logs, where each log contains a unique ID and timestamp. Timestamp is a string that has the following format: `Year:Month:Day:Hour:Minute:Second`, for example, `2017:01:01:23:59:59`. All domains are zero-padded decimal numbers.

Implement the `LogSystem` class:

* `LogSystem()` Initializes the `LogSystem`**** object. * `void put(int id, string timestamp)` Stores the given log `(id, timestamp)` in your storage system. * `int[] retrieve(string start, string end, string granularity)` Returns the IDs of the logs whose timestamps are within the range from `start` to `end` inclusive. `start` and `end` all have the same format as `timestamp`, and `granularity` means how precise the range should be (i.e. to the exact `Day`, `Minute`, etc.). For example, `start = "2017:01:01:23:59:59"`, `end = "2017:01:02:23:59:59"`, and `granularity = "Day"` means that we need to find the logs within the inclusive range from **Jan. 1st 2017** to **Jan. 2nd 2017** , and the `Hour`, `Minute`, and `Second` for each log entry can be ignored.

Example 1:

Input ["LogSystem", "put", "put", "put", "retrieve", "retrieve"] [[], [1, "2017:01:01:23:59:59"], [2, "2017:01:01:22:59:59"], [3, "2016:01:01:00:00:00"], ["2016:01:01:01:01:01", "2017:01:01:23:00:00", "Year"], ["2016:01:01:01:01:01", "2017:01:01:23:00:00", "Hour"]] **Output** [null, null, null, null, [3, 2, 1], [2, 1]] **Explanation** LogSystem logSystem = new LogSystem(); logSystem.put(1, "2017:01:01:23:59:59"); logSystem.put(2, "2017:01:01:22:59:59"); logSystem.put(3, "2016:01:01:00:00:00"); // return [3,2,1], because you need to return all logs between 2016 and 2017. logSystem.retrieve("2016:01:01:01:01:01", "2017:01:01:23:00:00", "Year"); // return [2,1], because you need to return all logs between Jan. 1, 2016 01:XX:XX and Jan. 1, 2017

23:XX:XX. // Log 3 is not returned because Jan. 1, 2016 00:00:00 comes before the start of the range. logSystem.retrieve("2016:01:01:01:01:01", "2017:01:01:23:00:00", "Hour");

Constraints:

* `1 <= id <= 500` * `2000 <= Year <= 2017` * `1 <= Month <= 12` * `1 <= Day <= 31` * `0 <= Hour <= 23` * `0 <= Minute, Second <= 59` * `granularity` is one of the values `["Year", "Month", "Day", "Hour", "Minute", "Second"]`. * At most `500` calls will be made to `put` and `retrieve`.

Code Snippets

C++:

```
class LogSystem {
public:
LogSystem() {

}

void put(int id, string timestamp) {

}

vector<int> retrieve(string start, string end, string granularity) {

}
};

/**
* Your LogSystem object will be instantiated and called as such:
* LogSystem* obj = new LogSystem();
* obj->put(id,timestamp);
* vector<int> param_2 = obj->retrieve(start,end,granularity);
*/
```

Java:

```
class LogSystem {

public LogSystem() {
```

```
}

public void put(int id, String timestamp) {

}

public List<Integer> retrieve(String start, String end, String granularity) {

}
}

/**
* Your LogSystem object will be instantiated and called as such:
* LogSystem obj = new LogSystem();
* obj.put(id,timestamp);
* List<Integer> param_2 = obj.retrieve(start,end,granularity);
*/
```

Python3:

```
class LogSystem:

def __init__(self):


def put(self, id: int, timestamp: str) -> None:


def retrieve(self, start: str, end: str, granularity: str) -> List[int]:



# Your LogSystem object will be instantiated and called as such:
# obj = LogSystem()
# obj.put(id,timestamp)
# param_2 = obj.retrieve(start,end,granularity)
```