Last updated: Feb 18, 2019

粉🚗

>> 面经时间:

面经时间: 最早 ~ 2019.2.09

>> 系统设计

- 设计一个donation 系统，要求实现信用卡支付的时候要exactly-once
- music sharing system
- 设计一个Lyft 的coupon 系统，对系统的并发要求不高，但要搞明白use case 和合理的数据存储方式。
- 设计一个drivers的实时监控的dashboard, 基本思路也是GeoHashing， 要求设计API给前端数据render 一个heat map
- implement price surge, 这一类的题还包括什么实时计算每个地区有多少driver / rider, 之类的，换汤不换药.(Three哥面试官，大概就是设计一个能够监控lyft rider driver的dashboard，比如有多少个driver在线，available还是已经接了订单在路上，有多少rider在等车。我先是把用户打车的流程化出来，然后再说每个流程中的component需要什么服务（payment，pricing，matching，rider，driver，route，transaction，navigation等等），然后再回答需要从哪几个service里面要数据，怎么用message queue等)
- design lyft
- tiny url
- Design Lyft driver and rider matching system
- design job scheduler
- 实现yelp的typeahead recommendation
- 设计delay queue
- distributed lock
- design an online KaraOk system(说API设计不够user friendly)

爬虫

Given 1000 slave machines, crawl the wikipedia
这题不是传统意义上的crawler，不需要多次crawl，只求download所有link里的内容一次，并保存.
这些slave machine要和你自己设计的一套系统communicate，他们只负责download东西，并且执行你自己给定的api
至于存储和deduplicate之类的事情，需要你自己另外设计

推荐阅读材料

- geohash

http://www.bigfastblog.com/geohash-intro
https://dzone.com/articles/designing-spacial-index
https://www.youtube.com/watch?v=cSFWlF96Sds

- uber talk

https://www.youtube.com/watch?v=AzptiVdUJXg
https://www.infoq.com/presentations/uber-flink-stream
https://www.slideshare.net/g9yuayon/streaming-analytics-in-uber

最后面两个link，如果面data infra看看，不面的话就不用看了吧，别看乱脑子。。

Save Copy to Evernote

>> 编程

首先建议大家，一定要提前把parse file，然后input output io的code 都提前写好了，input format不一定很准确，但是准备个大概的都会很有用
不然光是parse file可能就要很长时间。。。。

- 一个数组， 里面的元素可能是int或者另一个数组， 设计iterator. 设计个composite iterator使得可以同时iterate两个数组， 数组里面的元素类型如上所诉
- Given 2 sorted Arrays, Implement an iterator that will output the elements in 2 arrays in order.
- minStack, maxStack
- 汉诺塔，打印移动的方法。
- Find second largest element in a binary search tree
- . 设计一个司机和乘客的mapping 系统，一个m*n的二维数组，有很多司机和乘客（乘客远多于司机），int【x,y]代表所处的坐标，任务是给设计算法，保证每个司机都拉到一个乘客，并且所有的pickup话费的路程和最小

```
function to determine whether the driver is allowed to enter driver mode
* drivers are not allowed to drive a total of 12 hours without an 8 hour
break
* the function inputs are:
- a list of driver shifts as start/end integers, the integer is relative to
lyft launch
- the current time since lyft launch as integer


def can_drive(history, current_time):
    """
    Returns true if the driver has driven less than 12 hours since their
last 8 hour break


    history|array - Shifts, e.g. [(0, 12), (13, 19)]
    current_time|int - Current timestamp as hour since Lyft launch, e.g. 50


    can_drive = True example
        # 9 hour break, 1 hour shift, 2 hour break, 10 hour shift
        history = [(9, 10), (12, 22)]
        current_time = 24


    can_drive = False example:
        history = [(0, 4), (5, 9), (10, 14), (15, 19), (20, 24)]
        current_time = 24
不会答，大家看看，求解答。
```

```
前半:
給兩個排序過的 array
l1 = [1,2,4,8]
l2 = [2,8,9]
回傳兩個 array 都有的 element, e.g. [2, 8]
```

後半:
實作一個 CommenElement class ，constructor input 是兩個 sorted iterator，
要求實作 hasNext 和 next，假設還有下一個 common element hasNext return tr
next 則是回傳 next common element

白哥小哥，给了两个函数接口，需要实现针对不同的signalId可以注册执行函数，然后调用signal函数的时候，执行signalId对应的callable。
void register(int signalId, Callbable cb);
void signal(int signalId);
void remove(int signalID, Callable cd);
刚开始的时候问题不难，直接用hashmap和hashset，但后面各follow-up。比如Callable里面调用了signal和remove会有什么问题，怎么办。和小哥不断讨论，研究就好，还挺有意思的

给两个sorted integer array
int[] a = {1,2,3,4,4,5,5}
int[] b = {1,1,2,2,3}

要求返回其中重叠的部分 {1,2,3}。

follow up 是写一个iterator， 有两个方法 hasnext(), 和 next().
调用 next()， 依次返回 1-2-3.

*** 支持commit, rollback的nested transactions

地理其他面经也有过，给了API calls，分别为注册属于signal ID的callable，以及调用和移除 callable
void register(int signalId, Callbable cb);
void execute(int signalId);
void remove(int signalID, Callable cd);
基本概念就是用个map存放signal ID and Callable， 不难。后面有很多follow up，包括如果Cb A裡面有Cb B and Cb B又有Cb A的cycle情况，以及concurrency要怎麽处理。

Versioned key-value store

实现以下 几个 api set(k, value), get(k), getValueWithVersion(k, version)
就是每set一次key（不管哪个key），version就increment一次。
given a key and version, find the value, or if not version given, find the latest value
use a hashtable + treeMap

小行星LT735

给定了 Asteroid class ：

```
class Asteroid {
  public final int mass;
  public final int direction;
  public Asteroid(int mass, int direction) {
    this.mass = mass;
    this.direction = direction;
  }


  public int getMass() {
```

```
    return this.mass;
  }

  public int getDirection() {
    return this.direction;
  }


  public String toString() {
    return "Asteroid(" + mass + ", " + direction + ")";
  }
}
```

followup:
如果慧星有不同的速度，之间有不同的距离，结果有什么不同，如何设计算法?
好像没有好的办法，只能一步步更新。先算每个慧星碰撞到另外的时间，得到最小的碰撞时间，更新慧星的距离或坐标；重复直到没有慧星或只有反方向的慧星。

For example,

```
                space station   A1 A2 A3
mass          0                   1    2  3
distance     0                   6    8   10
speed        0                    -1    2   -2


at time 0.5, A2 and A3 collide, and update status
                space station   A1 A3
mass          0                   1    3
distance     0                  5.5  9
speed        0                    -1  -2. check 1point3acres for more.


at time 4, A1 and A3 collide, and update status
                space station   A3
mass          0                   3
distance     0                  2
speed        0                  -2


finally at time 5, A3 hits space station, and only 1 asteroid hits space station.
```

和leetcode原题不一样的是，题目有一个Space station在最右边，最后计算的是到达space station的astroids数量。
Follow up是如果给定陨石的质量，速度和到空间站的距离，问有几个能撞上空间站，这时候如果同方向的陨石相遇，大的会把小的毁掉

```
put('A1', 1)
put('A2', 'A1 + 4')
put('A3', 'A1 + A2 + 10')
put('A1', 'A3 - 5') (Throw exception, because of circular reference. Remember: A3 was A1
+ A2 + 10)

Get('A1') =>1
Get('A2') =>5
Get('A3') =>16
```

我大概能够想明白，用topological sort来判断是否 冲突 比如说第四个put，但是感觉这个题处理string会很麻烦，请问有没有战友见过这道题还记得 input的格式的，或者能提供更详细一点的细节，求分享


```
vector<int> fetch(int pageId): this function returns the items in given page, page size is fixed (say 5).


Implement a Fetcher class which has a function:
    vector<int> fetch(numOfItems): returns number of items.
using above api.


sample:
  fetch(0): return [0,1,2,3,4]
  fetch(1): return [5,6]
Fetcher class:
  fetch(2): return [0,1]
  fetch(1): return [2]
  fetch(4): return [3,4,5,6]
```

Coding: 3*3的矩阵是True或False的值， 有个api Toggle(row, col)可以把本行本列的所有值 x 变成 !x. 给个input matrix 和一个output matrix， 问最少需要多少次调用Toggle API可以达成转换。 挺tricky的， 不过本质上就是个找符合条件的subset的问题
Design: 设计个系统可以evaluate quality of AV maps

leetcode:

LRU
Word ladder 2
Read 4k
max Stack
rain trapping water
5, 8, 23, 35,42, 44, 54, 63,68,
128
173
200(follow up， 如果上边界下边界联通，左边界右边界联通，求问有几个岛。),
212, 238, 239, 279, 283, 287
347,
415
501
642(上机), 671,
735
935

759,

每个任务有开始时间，和完成它需要的时间，每个worker只能同时做一个任务。输入是标准输入，第一行是任务数量，下面每一行是任务标号，开始时间，和完成所需时间，求，共需要多少个worker，和每一个任务是由哪个worker完成的