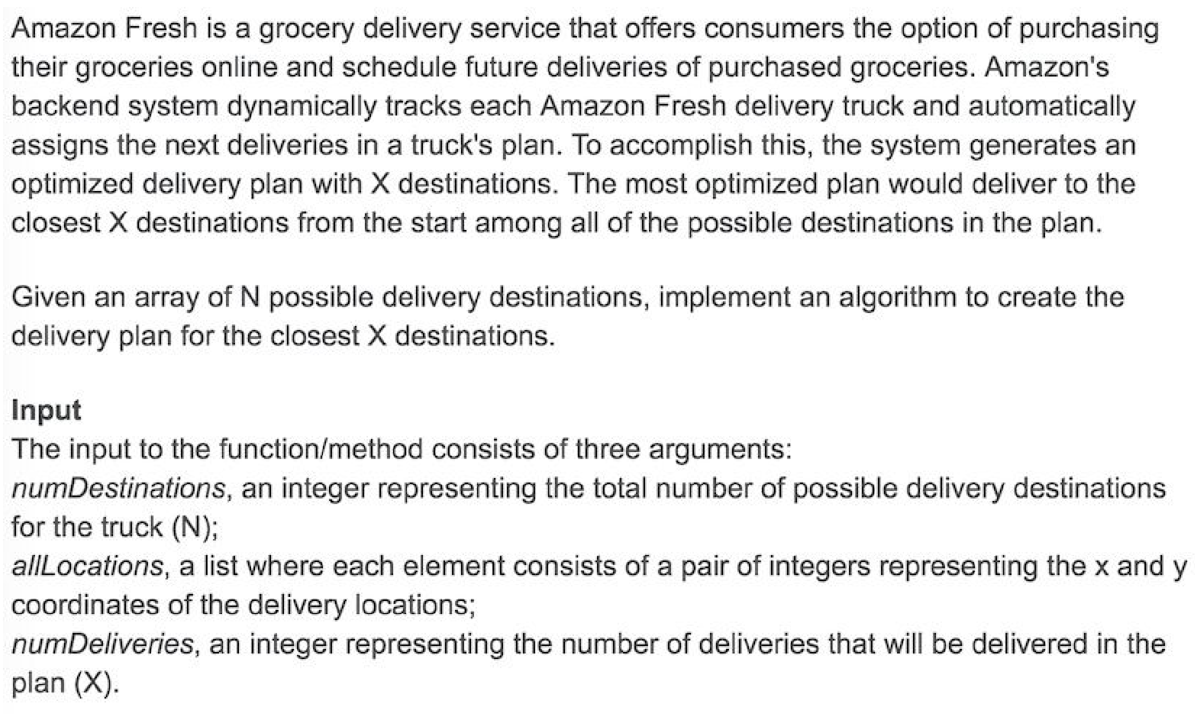
第一题是Amazon Fresh送货的



Amazon Fresh is a grocery delivery service that offers consumers the option of purchasing their groceries online and schedule future deliveries of purchased groceries. Amazon's backend system dynamically tracks each Amazon Fresh delivery truck and automatically assigns the next deliveries in a truck's plan. To accomplish this, the system generates an optimized delivery plan with X destinations. The most optimized plan would deliver to the closest X destinations from the start among all of the possible destinations in the plan.

Given an array of N possible delivery destinations, implement an algorithm to create the delivery plan for the closest X destinations.

Input
The input to the function/method consists of three arguments:
numDestinations, an integer representing the total number of possible delivery destinations for the truck (N);
allLocations, a list where each element consists of a pair of integers representing the x and y coordinates of the delivery locations;
numDeliveries, an integer representing the number of deliveries that will be delivered in the plan (X).

Output
Return a list of elements where each element of the list represents the x and y integer coordinates of the delivery destinations.

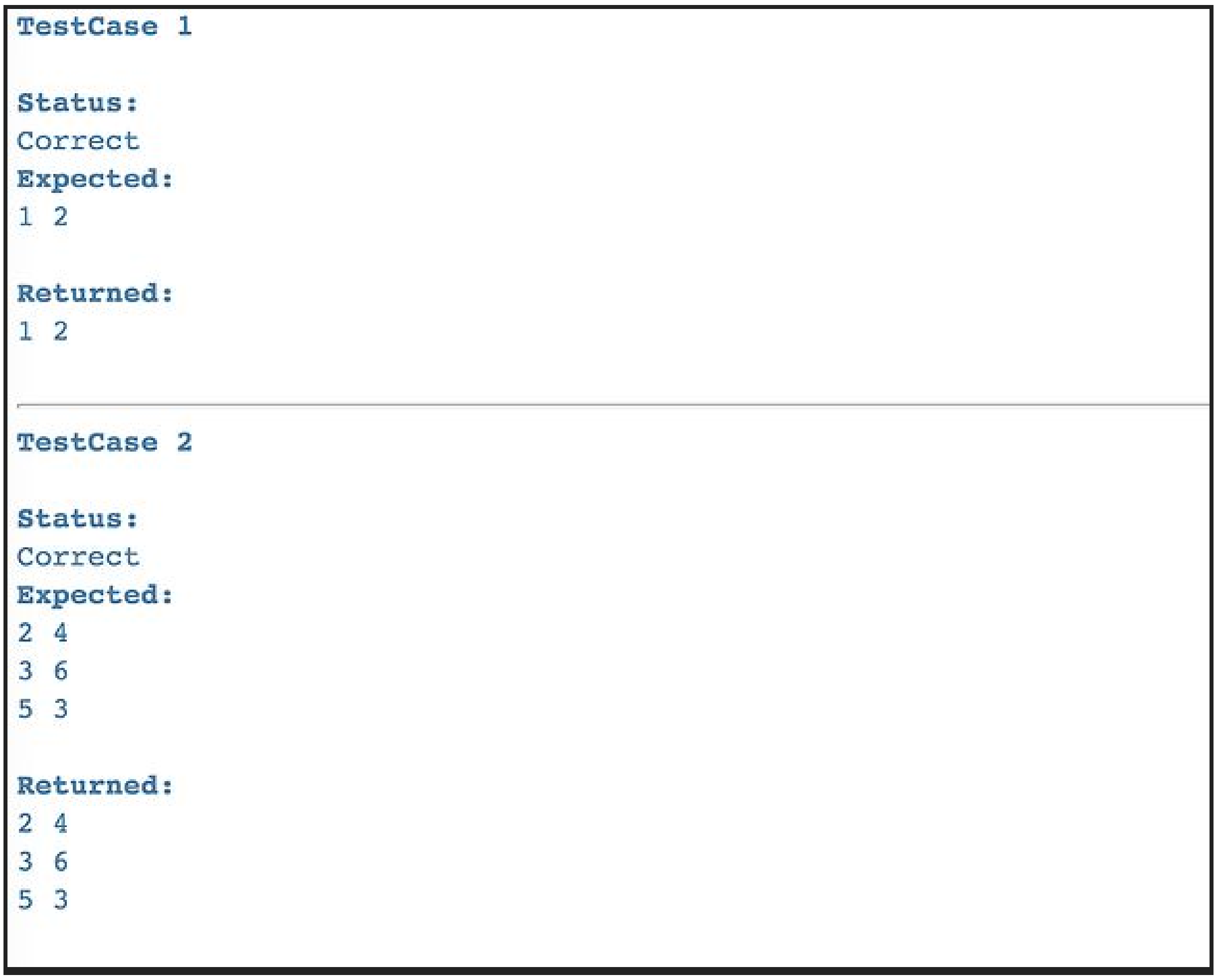
Constraints
numDeliveries ≤ *numDestinations*

Note
The plan starts from the truck's location [0, 0]. The distance of the truck from a delivery destination (x, y) is the square root of $x^2 + y^2$. If there are ties then return any of the locations as long as you satisfy returning X deliveries.

Example
Input:
numDestinations = 3
allLocations = [[1, 2], [3, 4], [1, -1]]
numDeliveries = 2

Output:
[[1, -1], [1, 2]]

Explanation:
The distance of the truck from location [1, 2] is square root(5) = 2.236
The distance of the truck from location [3, 4] is square root(25) = 5
The distance of the truck from location [1, -1] is square root(2) = 1.414
numDeliveries is 2, hence the output is [1, -1] and [1, 2].



```
TestCase 1

Status:
Correct
Expected:
1 2

Returned:
1 2
```

```
TestCase 2

Status:
Correct
Expected:
2 4
3 6
5 3

Returned:
2 4
3 6
5 3
```

第二题是无人机送货

Amazon Prime Air is developing a system that divides shipping routes using flight optimization routing systems to a cluster of aircraft that can fulfill these routes. Each shipping route is identified by a unique integer identifier, requires a fixed non-zero amount of travel distance between airports, and is defined to be either a forward shipping route or a return shipping route. Identifiers are guaranteed to be unique within their own route type, but not across route types.

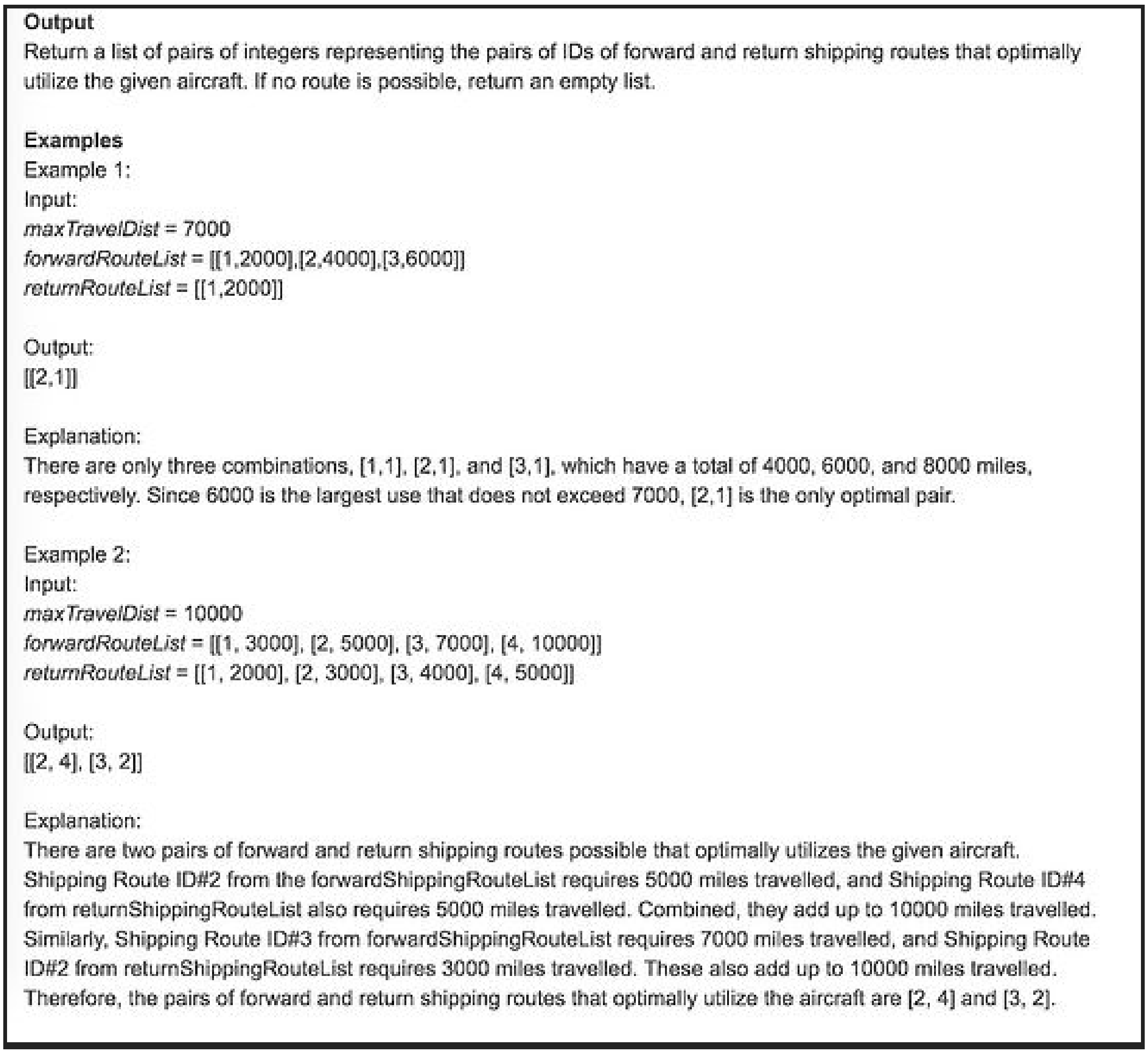
Each aircraft should be assigned two shipping routes at once: one forward route and one return route. Due to the complex scheduling of flight plans, all aircraft have a fixed maximum operating travel distance, and cannot be scheduled to fly a shipping route that requires more travel distance than the prescribed maximum operating travel distance. The goal of the system is to optimize the total operating travel distance of a given aircraft. A forward/return shipping route pair is considered to be "optimal" if there does not exist another pair that has a higher operating travel distance than this pair, and also has a total less than or equal to the maximum operating travel distance of the aircraft.

For example, if the aircraft has a maximum operating travel distance of 3000 miles, a forward/return shipping route pair using a total of 2900 miles would be optimal if there does not exist a pair that uses a total operating travel distance of 3000 miles, but would not be considered optimal if such a pair did exist.

Your task is to write an algorithm to optimize the sets of forward/return shipping route pairs that allow the aircraft to be optimally utilized, given a list of forward shipping routes and a list of return shipping routes.

Input
The input to the function/method consists of three arguments:
maxTravelDist, an integer representing the maximum operating travel distance of the given aircraft;
forwardRouteList, a list of pairs of integers where the first integer represents the unique identifier of a forward shipping route and the second integer represents the amount of travel distance required by this shipping route;
returnRouteList, a list of pairs of integers where the first integer represents the unique identifier of a return shipping route and the second integer represents the amount of travel distance required by this shipping route.

Output



Output
Return a list of pairs of integers representing the pairs of IDs of forward and return shipping routes that optimally utilize the given aircraft. If no route is possible, return an empty list.

Examples
Example 1:
Input:
maxTravelDist = 7000
forwardRouteList = [[1,2000],[2,4000],[3,6000]]
returnRouteList = [[1,2000]]

Output:
[[2,1]]

Explanation:
There are only three combinations, [1,1], [2,1], and [3,1], which have a total of 4000, 6000, and 8000 miles, respectively. Since 6000 is the largest use that does not exceed 7000, [2,1] is the only optimal pair.

Example 2:
Input:
maxTravelDist = 10000
forwardRouteList = [[1, 3000], [2, 5000], [3, 7000], [4, 10000]]
returnRouteList = [[1, 2000], [2, 3000], [3, 4000], [4, 5000]]

Output:
[[2, 4], [3, 2]]

Explanation:
There are two pairs of forward and return shipping routes possible that optimally utilizes the given aircraft. Shipping Route ID#2 from the forwardShippingRouteList requires 5000 miles travelled, and Shipping Route ID#4 from returnShippingRouteList also requires 5000 miles travelled. Combined, they add up to 10000 miles travelled. Similarly, Shipping Route ID#3 from forwardShippingRouteList requires 7000 miles travelled, and Shipping Route ID#2 from returnShippingRouteList requires 3000 miles travelled. These also add up to 10000 miles travelled. Therefore, the pairs of forward and return shipping routes that optimally utilize the aircraft are [2, 4] and [3, 2].

【Heap问题】
1. Top K closest numbers / 找最近的K个餐馆 / 饭店最近的位置 / 给一堆的餐馆的位置， 和customer的位置， 找出离customer 最近的餐馆。 / 找最近的k个餐馆， 给了餐馆的坐标， 和距离的定义 / 一个是求K个最近的餐馆， 用heap做就好了。
Given a list of points, and a point, find the K closest points
函数：nearestKRestruants(List<List<Integer>> Locations, int K)

array -- vector vector -- list list 加大难度

参考解法：用max heap做， for 每个点， 算dist, heapify 所有点， 最后pop k top points。时间: **O(n+klogn)**。all 15 test cases passed。这里似乎不用考虑**integer overflow**。

k个地点距离， 用x^2 + y^2就能通过所有test case， 非常简单。 （但之后**面试会不会问怎么optimize就不一定了**， 但我也没想出怎么optimize）

2. Five hightest：选每个产品五个评分最高的求平均值。给一个list，每个element是<productId, productRating>，求每个product最高的5个评分的Average

参考解法：TreeMap<productId, Heap(5)> 然后遍历

3. 卡车送货，求k个最近距离。一个二维grid，有些点标记成了送货地址，输出从（0，0）出发最近的k个目标。
参考解法：Heap

【Follow-up】第二题还是送货，有障碍，求到目标地址的最短距离
参考解法：bfs

4. 找最近的几个点

【求Pair Sum】

1. 飞机来回路程的问题

油箱最多走Max = 10000
去程：[1, 2000] [2, 5000]
回程：[1, 5000] [2, 2000] [3, 8000]
问飞机最多可以走哪几条路？不一定能走满Max，可以是最靠近Max的数。

参考解法：见题2（思路一模一样）

2. 类似Two Sum Closest
给两个数组。从两个数组中分别取一个数，**和要小于等于k。找到和最大的组合。**
（其他描述：返回两个list中和最大但不超过一个值的下标－函数名optimalUtilization）
给2个sorted array，和一个整数capacity，**每个array各找出一个数，组成一个pair**。找出pair满足以下条件：
1）sum of pair <= capacity
2) sum is maximum
后来真的遇到这道题的时候还是**先用O(N*N)的算法检查**了所有可能的组合，oj是可以过的。我觉得这题的考点不在降低时间复杂度，而在不能**错过任何一个重复的组合**。

https://yeqiuquan.blogspot.com/2017/04/lintcode-533-two-sum-closest-to-target.html

参考思路：
两层for循环，穷举所有组合。这种方式简单能过。

其它思路：sort, 2pointer 或者binary search。

输入输出比较复杂。不建议用nlogn的解法。直接双循环暴力解即可。

sortA, sortB: mlogm + nlogn, binary search num from A in B. mlogn, total: mlogm + nlogn + mlogn

容易出错点：**数组里有重复元素，漏解**

3. 有两个List of apps， 里面存着每一个app的index 和 它所需的memory。 从 foregroundList<Integer>（[1, 200], [2, 300]） 和一个boregroundList<Integer> ([1, 400], [2, 500]) 中各取一个组成一对，使得它们memory加起来的和最接近但不超过capacity。 需要注意的是多个app的memory有可能是相同。

例如： foregroundList<Integer> =[[1, 2000]], boregroundList<Integer> = [[1, 8000], [2, 8000]], capacity = 10000. 需要返回 [[1,1], [1,2]]。

描述2：给了两组application的ID和内存需要，以及一个目标内存，求一对application的ID使得它们的内存使用之和最接近目标内存。

参考思路：
可以用暴力解，也可以用two pointers。我用了treeset， all case passed。
第二题：O(N*N) -> O(NlogN)
先sort BACK list,
再遍历Front list：
for each element i in FRONT
search for lower_bound(capacity-i)
输出的candidates用max stack存储，有新来的就一直pop直到top等于新来的或者stack为空。

顺序你是指相同的capacity的时候，不同id number组合的排序吗？好像是没有特别要求吧。。。

4. leetcode-001 Two Sum

【BFS的题】

1. **list of list 中BFS最短路径题**。
0，1，9组成grid。从（0，0）出发，4个方向走。0能走，1不能走。到达9的最小步数。
给一个二维数组，里面有1，0，9，找从左上到有9的最小距离

参考解法：BFS

2. Maze返回最短距离。在maze里求到某个点的最短距离，OA里面input是**List<List<Integer>>**

参考答案：BFS

3. 停车场里找obstable最短路径的题目。记得没找到obstacle要返回-1，不然只会过12个test case。加了这行就16个test case全过了，不需要再optimize。

参考解法：BFS

4. Black and White

【MST】

1. MST：给你一堆connection，求能链接所有城市的最小的cost。具体谷歌搜索 **mst algorithm** 就有类似的题。（感觉这是oa2的标配。做得时候最后有一个testcase 想了很久，就是如果不能成功的链接所有程序，例如用unionfind 有一个 城市跟别的城市不一样组，就返回空list）

其他描述：
一道MST的题的变形，搜一下MST的算法，感觉挺难的，我反正写了好久 好像8个test case吧。
一个MST的问题，写之前可以google一下求MST的算法，练一练。

参考解法：**最小生成树两种算法及Union Find**

https://www.geeksforgeeks.org/prims-minimum-spanning-tree-mst-greedy-algo-5/

2. **Maximum Minimum Path**。

【Tree】

1. 求最大**subtree**。

【LinkedList】
1. 合并两个LinkedList / merge two sorted linkedlist 关键词：package， belt。

【字符串】
1. count substring with k distinct characters
参考解法：**用一个int[26]存a-z字母出现的index，一旦长度到了k，存下来**