1. <https://leetcode.com/discuss/interview-question/344677/Amazon-or-Online-Assessment-2019-or-Min-Cost-to-Connect-Ropes>
2. 2. BFS 走 2D arra, function的input是一個List<List<Integer>>，我是自己build 2D array, 给一个2D grid，上面有一些obstacles，找出从左上角到右下角的最短路径

查找最短路径到达obstacle.  基本和 Treasure Island 一样，BFS就行， 注意0 和 1的具体含义，这里搞错了找了半天  
Input (int numRows, int numColumns, List<List<Integer>> lot)  
0 表示 trench 不可通行  
1 表示 flat 可通行  
9 表示 obstacle  
  
Example  
numRows = 5  
numColumns = 4  
[[1,0,0,0],  
[1,0,0,0],  
[1,1,0,0],  
[0,1,0,0],  
[0,9,‍‌‌‌‍‍‍‍‍‍‌‌‌‌‌‌‌‍0,0]  
]  
  
Output: 5

<https://leetcode.com/discuss/interview-question/347457>

1. song duration给出一堆音乐和时长，找出一对音乐，时长刚好等于target，如果有tide，选择pair中有更长时间音乐的那个。题目没说有没有duplicate，但是这似乎不影响，当作没有duplicate做就行了, ba道题的两道第一题是给你一个songDuration list 和一个rideDuration，然后找两首歌的index，使得他们的duration加起来和跟rideDuration 正好差三十。 简单的two sum
2. robot move obstacle. 类似利口505
3. 第一题是从草堆里找针，其实就是四撒巴的变形。
4. 第二题是匹配括号,‍‌‌‌‍‍‍‍‍‍‌‌‌‌‌‌‌‍直接是二史。
5. merge file/合并文件
6. 机上放音乐还是影片，两个加起来刚好等于一个值而且要包含所有符合的结果里面最长一个电影的。返回一对int
7. 飞机来回程利用飞机max range最大的组合。输入是两个二维list，包含index和range  
   第一个用了map O(n), 第二个用暴力
8. IDE真的非常难用，要手动import库
9. 是接绳子那道题换了个皮，求minimum cost，example: [8, 4, 6, 12], output = (4 + 6) + (10 + 8) + (18 + 12) = 58. PriorityQueue秒了
10. 给两个int array，给个target，用两个array中的数字凑最接近target的组合。example：array1: [[1, 2000], [2, 4000], [3,6000]] array2: [[1, 2000]] target是7000 答案是[2 , 1], 每个数组的每一行的第一个数字是id，第二个数字是实际数字。上面的例子，[1,1]是4000，[2,1]是6000，[3,1]是8000 不能超过target 所以答案是[2,1] 有个坑是组合可能不‍‌‌‌‍‍‍‍‍‍‌‌‌‌‌‌‌‍止1对，比如array1中还有个数是[4, 4000], 那答案就是[[2,1], [4,1]]。用TreeMap秒了。
11. <https://leetcode.com/discuss/interview-question/344650/Amazon-Online-Assessme%E2%80%8D%E2%80%8C%E2%80%8C%E2%80%8C%E2%80%8D%E2%80%8D%E2%80%8D%E2%80%8D%E2%80%8D%E2%80%8D%E2%80%8C%E2%80%8C%E2%80%8C%E2%80%8C%E2%80%8C%E2%80%8C%E2%80%8C%E2%80%8Dnt-Questions>
12. 1. reorder logs，[刷题](http://www.1point3acres.com/bbs/forum-84-1.html)网原题，改写comparator就可以  
    2. foreground apps和background apps那道，用treemap
13. 第一题就相当于是closest two sum (从两个不同array 找)  
    第二题是无人机来回，去程和回程不能超过limit且要最长，解法基本就是foreground app 和background app 一样  
    其实这两题想法都差不不多，就是第二题implementation‍‌‌‌‍‍‍‍‍‍‌‌‌‌‌‌‌‍更繁琐点