

# Amazon

---

## 社招oa :

---

1. 机器人组装 求最短时间 - min heap

因为js没有heap, 我就拿python做的。最开始一直有个case过不了, 于是拿掉了特殊判断, 结果就过了

2. [Amazon](#) Fresh找最短路径 - BFS

最开始忘加visited, 结果没有一个通过, 加了全过了

---

9.16收到oa邮件, 晚上刚刚做, 都是地里大家发过的, 九十分钟两道题 :

1. 会员非会员文件排序,  
利口酒散起

2. 机器人找障碍,

bfs, 注意输入不是数组是列表

之后十五分钟写思路

---

刚做完, 都是最近的面经题

Reorder log

入参是list 不是string数组

Remove obstacle

注意有个坑

Lot 复制给二维数组时

List <Integer> sub= lot.get(i) 这个本地eclipse 可以 傻逼oa死活通不过

后来删除了直接for二重循环复制过了

---

热带雨林的OA

- 第一题类似Two sum, 只有一组解, 返回数字所在的index即可。这里直接使用hashmap就行。  
O(N)复杂度。

- 第二题类似2Dlist找宝藏, 1是能走的路, 0是障碍不能走, 9是终点。从左上角出发最少步数到终点。这里dfs, bfs都行, 推荐bfs因为是要算最少步数。

---

Q1. 合并零件，找最小cost，地里有

Q2. 卡车在一个2D matrix里面送货，1可以走，0不能走，9是目的地。输出需要的步数就可以了。里口类似题是溜叁。

amcatglobal.aspiringminds.com

amcat

amazon

Automata  
Amazon

QUESTION  
1 out of 2

aspiringminds

HELP

Problem | Test Cases | Output

Amazon Fulfillment Builder is a new feature that enables Amazon warehouses to create new items to ship to customers out of smaller parts. As part of this project, Amazon wants to estimate the time it will take for a worker to create the item to be ready for a customer shipment.

The Amazon Fulfillment Builder will provide an estimate about the time it will take for the item to be created based on the size of each of the parts. The worker can only combine two parts at a time. The time required to put two parts together is equal to the sum of the parts sizes. The size of the newly constructed part is also equal to the sum of the part's sizes. This process is repeated until all of the parts have been merged together to form the final product.

Write an algorithm to output the minimum possible time to put the N parts together and build the final product.

**Input**  
The input to the function/method consists of two arguments:  
*numOfParts*, an integer representing the number of the parts;  
*parts*, a list of integers representing the size of the parts.

**Output**  
Return an integer representing the minimum time required to assemble all the parts.

**Constraints**  
 $2 \leq \text{numOfParts} \leq 10^5$   
 $1 \leq \text{parts}[i] \leq 10^5$   
 $0 \leq 1 < \text{numOfParts}$

**Example**  
Input:  
*numOfParts* = 4

Compile and Run

```
1 // IMPORT LIBRARY PACKAGES NEEDED BY YOUR PROGRAM
2 // SOME CLASSES WITHIN A PACKAGE MAY BE RESTRICTED
3 // DEFINE ANY CLASS AND METHOD NEEDED
4 import java.util.List;
5 import java.util.*;
6 // CLASS BEGINS, THIS CLASS IS REQUIRED
7 public class Solution
8 {
9     // METHOD SIGNATURE BEGINS, THIS METHOD IS REQUIRED
10    int minimumTime(int numOfParts, List<Integer> parts)
11    {
12        // WRITE YOUR CODE HERE
13
14        // create min heap
15        Queue<Integer> pq = new PriorityQueue<>();
16
17        for (Integer part : parts) {
18            pq.add(part);
19        }
20
21        int cost = 0;
22        while(pq.size() > 1) {
23            int tmp = pq.poll() + pq.poll();
24            cost += tmp;
25            pq.add(tmp);
26        }
27
28        return cost;
29    }
30    // METHOD SIGNATURE ENDS
31 }
```

SUBMIT ANSWER

and build the final product.

### Input

The input to the function/method consists of two arguments:  
*numOfParts*, an integer representing the number of the parts;  
*parts*, a list of integers representing the size of the parts.

### Output

Return an integer representing the minimum time required to assemble all the parts.

### Constraints

$$2 \leq \text{numOfParts} \leq 10^6$$

$$1 \leq \text{parts}[i] \leq 10^6$$

$$0 \leq i < \text{numOfParts}$$

### Example

Input:

*numOfParts* = 4

*parts* = [8, 4, 6, 12]

Output:

58

Explanation:

The optimal way to assemble the parts is as follows:

Step 1: Assemble the parts of size 4 and 6 (time required is 10). Size of remaining parts after merging: [8, 10, 12].

Step 2: Assemble the parts of size 8 and 10 (time required is 18). Size of remaining parts after merging: [18, 12].

Step 3: Assemble the parts of size 18 and 12 (time required is 30).

Total time required to assemble the parts is  $10 + 18 + 30 = 58$ .

Amazon Software Development

amcatglobal.aspiringminds.com

amcat

amazon

Automata  
Amazon

67m  
57%

QUESTION  
2 out of 2

aspiringminds

HELP

Problem | Test Cases | Output

Amazon Fresh is a grocery delivery services that offers consumers the option of purchasing their groceries online and having them delivered on schedule. The Amazon Fresh team is planning a route for a delivery truck to deliver customer orders in the city of Techlandia. The planner will create a delivery area for each order to effectively plan the route. The area is abstracted as a grid. Not all locations are accessible by road. The truck only needs to make a single delivery.

Write an algorithm to determine the minimum distance required for the truck to deliver the order.

Assumptions:

- Some places in the delivery area cannot be accessed by the driver, as there are no roads into those locations.
- The delivery area can be represented as a two-dimensional grid of integers, where each integer represents one cell.
- The truck must start from the top-left corner of the area, which is always accessible, and can move one cell up, down, left, or right at a time.
- The truck must navigate around the areas without roads and cannot leave the area.
- The accessible areas are represented as 1, areas with without roads are represented by 0 and the order destination is represented by 9.

Input  
The input to the function/method consists of three arguments:  
*numRows*, an integer representing the number of rows;  
*numColumns*, an integer representing the number of columns;  
*area*, representing the two-dimensional grid of integers.

Output  
Return an integer representing the total distance traversed to deliver the order else return -1.

Compile and Run

```
1 // IMPORT LIBRARY PACKAGES NEEDED BY YOUR PROGRAM
2 // SOME CLASSES WITHIN A PACKAGE MAY BE RESTRICTED.
3 // DEFINE ANY CLASS AND METHOD NEEDED
4 import java.util.List;
5 import java.util.*;
6 // CLASS BEGINS, THIS CLASS IS REQUIRED
7 public class Solution
8 {
9     // METHOD SIGNATURE BEGINS, THIS METHOD IS REQUIRED
10    int minimumDistance(int numRows, int numColumns, List<List<Integer>> c
11    {
12        // WRITE YOUR CODE HERE
13        boolean[][] visited = new boolean[numRows][numColumns];
14
15        int ans = 0;
16        int[][] DIRS = new int[][]{{1, 0}, {-1, 0}, {0, 1}, {0, -1}};
17
18        Queue<int[]> q = new LinkedList<>();
19        q.add(new int[] {0, 0});
20
21        while(!q.isEmpty()) {
22            int size = q.size();
23
24            for (int i = 0; i < size; i++) {
25                int[] curr = q.poll();
26                int x = curr[0];
27                int y = curr[1];
28
29                if (x < 0 || x >= numRows || y < 0 || y >= numColumns ||
30                    area.get(x).get(y) == 0 || visited[x][y]) {
31                    continue; // skip this point
32                }
33
34                visited[x][y] = true;
```

SUBMIT TEST SUBMIT ANSWER



7m  
1s

QUESTION  
2 out of 2

aspiringminds  
Empowering Geniuses

HELP ?

**Problem** | **Test Cases** | **Output** |

represented by 0 and the order destination is represented by 9.

**Input**

The input to the function/method consists of three arguments:  
*numRows*, an integer representing the number of rows;  
*numColumns*, an integer representing the number of columns;  
*area*, representing the two-dimensional grid of integers.

**Output**

Return an integer representing the total distance traversed to deliver the order else return -1.

**Constraints**

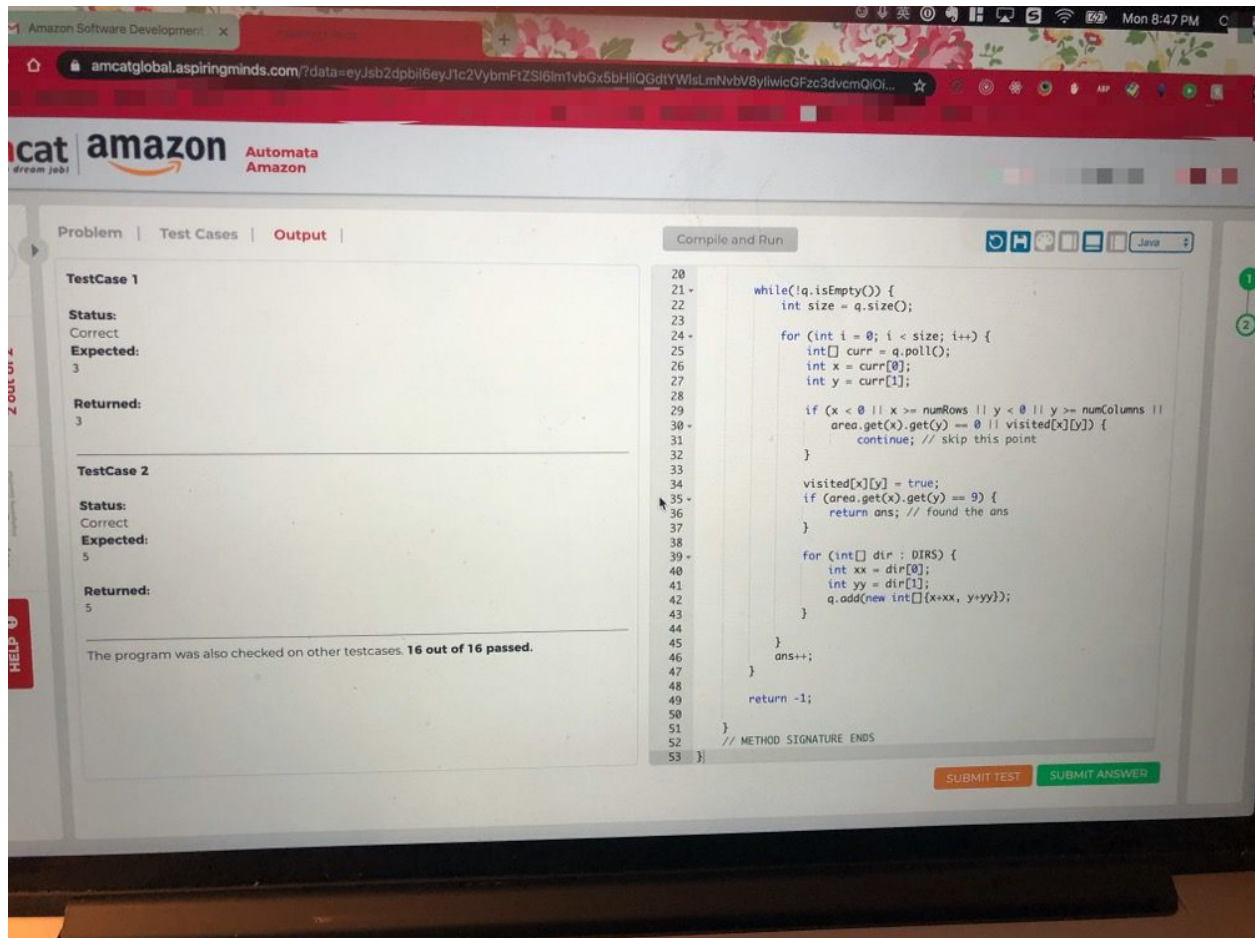
$1 \leq \text{numRows}, \text{numColumns} \leq 1000$

**Example**

Input:  
*numRows* = 3  
*numColumns* = 3  
*area* =  
[[1, 0, 0],  
[1, 0, 0],  
[1, 9, 1]]

Output:  
3

Explanation:  
Starting from the top-left corner, the truck traversed the cells (0,0) -> (1,0) -> (2,0) -> (2,1).  
The truck traversed the total distance 3 to deliver the order.  
So, the output is 3.



## New grad OA2

- [Two Sum - Unique Pairs](#)



[New Grad] +1

- [21Merge Two Sorted Lists](#)



[New Grad]



- 1192 [Critical Connections in a Network](https://leetcode.com/problems/critical-connections-in-a-network/) [New Grad] +1 16个test cases有2个没过 地里有geeks的链接, 原代码.输入输出都是List<PairInt>

<https://www.geeksforgeeks.org/bridge-in-a-graph/>

[Misty2019](#) 发表于 2019-9-5 01:31

谢谢楼主的信息, 想问一下critical connection那道题输出顺序有要求嘛, 比如 1 3, 2 4是输出, 那么输出1 3 ...

-我记得是没有的! 只要注意edge本身 -> 需要是1 3和2 4, 而不是3 1和4 2。

The screenshot shows a coding problem interface. On the left, there's a sidebar with 'QUESTION 2 out of 2' and 'aspiringminds' logo. The main content area has the following text:

Write a method that returns the critical connections in AMZN525.

**Input**  
The input to the function/method consists of three arguments -  
*numOfServers*, an integer representing the number of servers in the data center;  
*numOfConnections*, an integer representing the number of connections between the servers;  
*connections*, a list of pairs of integers representing the connections between two servers.

**Output**  
Return a list of integer pairs representing the critical connections. Output an empty list if there are no critical connections.

**Constraints**  
 $0 \leq \text{numOfServers} \leq 100,000$   
 $0 \leq \text{numOfConnections} \leq 100,000$   
 $1 \leq \text{connections}[i][j] \leq \text{numOfServers}$   
 $0 \leq i < \text{numOfConnections}$   
 $0 \leq j < 2$

**Example**  
Input:  
*numOfServers* = 5  
*numOfConnections* = 5

On the right side of the screenshot, there's a code editor with some Java code visible, including a class definition and some comments.

我用dfs low是跑过了16个test

case<https://www.1point3acres.com/bbs/forum.php?mod=viewthread&tid=549051&extra=page%3D3%26filter%3Dsortid%26sortid%3D311%26searchoption%5B3046%5D%5Bvalue%5D%3D5%26searchoption%5B3046%5D%5Btype%5D%3Dradio%26searchoption%5B3109%5D%5Bvalue%5D%3D1%26searchoption%5B3109%5D%5Btype%5D%3Dradio%26sortid%3D311%26orderby%3Ddateline>



给了一个warehouse list和一个road list, warehouse之间可能有road连接, 初始所有warehouse都是连接起来的, 返回critical road (critical road指的是移除它, 原图不再是一个connected component) 本质上就是求图中的bridge,

## 2. Critical Links

题名不记得了, 就是那道著名的找 server 中的 critical connections, 这道题是关于图的 Finding bridge 题目, 关于bridge的问题有现成的算法, 如果不熟悉的同学请参考以下链接:

Notes: 如果你对图的概念及算法不熟悉的话, 建议先去熟悉一下Graph数据结构的基础知识, 这道题是Undirected Graph, 关于Directed Graph的内容可以暂时不看。

关于 Articulation Point or Cut-Vertex in a Graph:

<https://www.hackerearth.com/practice/notes/nj/> (建议看Finding Bridge前先把这个看了, 这个看懂了Finding bridge就改两行代码)

关于 Finding Bridge in a Graph: <https://www.geeksforgeeks.org/bridge-in-a-graph/>

做题的时候记得手动 import java.util.\* (如果没遇到这问题就忽略)

题目中有以下要注意的地方:

第一, 当图为空的时候, 记得要返回空List

第二, 当找到一个 Critical connection 的时候, 记得把两个 vertex 的 id 排一下序再存入list里, 比如 6 和 7 之间的 bridge 应该是 (6, 7), 你如果存入的是 (7, 6) 就会报错

第三, 函数返回值是List<PairInt>, 这个是亚麻自定义的 data structure, 记得处理一下就行, 我记得定义如下:

```
public class PairInt {
    int first;
    int second;
    public PairInt(int first, int second) {
        this.first = first;
        this.second = second;
    }
}
```

<https://www.1point3acres.com/bbs/forum.php?mod=viewthread&tid=547008&extra=page%3D5%26filter%3Dsortid%26sortid%3D311%26searchoption%5B3046%5D%5Bvalue%5D%3D5%26searchoption%5B3046%5D%5Btype%5D%3Dradio%26searchoption%5B3109%5D%5Bvalue%5D%3D1%26searchoption%5B3109%5D%5Btype%5D%3Dradio%26sortid%3D311%26orderby%3Ddateline>


- [Favorite Genres](#)



[New Grad]

- 240[Search a 2D Matrix II](#)  [New Grad] 但是要注意返回的是 ArrayList 类型，而不是二维数组。

- 138[Copy List with Random Pointer](#)  [New Grad]

- 572[Subtree of Another Tree](#)  [New Grad] 但是要注意返回值不是 true or false，变成 1 或者 -1 了。
- 105[Construct Binary Tree from Preorder and Inorder Traversal](#)

Amazon wants to improve the Audible experience for its customers by offering better book recommendations based on what the user has already listened to. To make this possible, Amazon needs to identify users' favorite book genre(s). This is done by identifying the most listened to book genres from the user history and giving them recommendations from the same. A user can have more than one favorite genre if he/she has listened to the same number of books per each of the genres.

Amazon has following information to identify a user's interest(s):

- The name of the user mapped to the collection of all the books that he/she has listened to, and
- A list of all books and their associated genres. For the scenario, imagine that a book can belong to only one genre.

Write an algorithm that will produce a map/dictionary of users and their favorite book genres.

#### **Input**

The input to the function/method consists of four arguments:

*numUsers*, an integer number of users;

*userBooksListenedTo*, a map with user names as keys and a list of all the books that the user has listened to as values;

*numGenres*, an integer number of all genres available;

*bookGenres*, a map with book genre as keys with a list of all the books within that genre as values.

#### **Output**

Return a map/dictionary with the preferences of all users. Key represents the user name and value represents a list of his/her favorite genres.

#### **Note**

User names, genres, and book names all are case sensitive English alpha-numeric strings.

#### **Example**

Input

*numUsers*= 3

```
userBooksListenedTo= {  
    "Fred": ["mass", "world", "stress"],  
    "Jenie": ["happy", "pride"],  
    "Rone": ["alexender"]  
}
```

*numGenres*= 3

```
bookGenres= {  
    "Horror": ["mass", "stress"],  
    "Comedy": ["happy"],  
    "Romance": ["world", "alexender", "pride"]  
}
```

Output

```
{  
    "Fred": ["Horror"],  
    "Jenie": ["Comedy", "Romance"],  
    "Rone": ["Romance"]  
}
```

知道地里面经很多，我也就不废话了（主要是想求米）

9/6 收到OA1， Debug都是地里原题；

9/9 收到OA2， coding section也是地里原题， two sum和deep copy；work styles assessment就随意选的符合自己性格的；

9/12 收到OA3， work simulation部分我觉得还挺迷的， 不确定按照什么标准选；logic的题差不多是小土刀的原题。时间非常充裕。

然后就是漫长的等待...

想说的就这些， 有什么问题可以再补充（估计没有。。）

---

Q2. isSameTree。 tree的结构是binary tree 做法一样。

输出 是int -1 = false 1 = true 结束

---

oa 2

1: 一个map user对books 一个kind:books 返回一个user最喜欢的类型的map

2: isSubtree

---

第二题：和lc 138. Copy List with Random Pointer题目很像， 但是有节点不在head的next里， 比如1->2->3->null, 1->random=4,4不在head1的next里面儿。

做法：首先判头是不是空， 因为LZ被这个边界卡了半天哈哈哈哈。新建一个hashmap。建一个queue把头放进去。每次出队， 都判断在不在map里， 如果不在就加入map， 把他的next和random都放进队里。

然后遍历hashmap,把每个val的next和random根据key的next和random赋值。

---

OA2 是两道coding, 第一道是find subtree, 第二道是书本的那个问题， 具体是给你一个hashmap， 然后求出user最喜欢的种类， 也很简单。

---

1. 给两个map， 第一个是string 人名 : vector<string> 书， 第二个是string 类型 : vector<string> 书

需要返回一个map， string 人名 : vector<string> 最喜欢看的类型

```
// map1 {"Tom" : ["booka", "bookb"]; "Jack" : ["booka","bookc","bookd"]};
```

```
// map2 {"happy" : ["booka", "bookd"]; "sad" : ["bookb","bookc"]};
```

```
// res { "Tom" : ["happy", "sad"]; "Jack" : ["happy"]};
```

这里jack看了A C D,但是A D都是happy， 两票， C是sad， 一票， 所以只有happy最喜欢的类型是happy。

需要注意的edge case有三种。 第一种是map1是空的， 直接返回一个空map。第二种情况是map2是空的， 这个时候要把人名加进去， 但是每个人名的value是个空vector。第三种情况是map1只有人名， value里的vector是空， 这个时候要把人名存进去， value是空的。

之前看面经自己作的时候没注意第三个情况， 导致17个test有两个没过， 知道最后只剩下5分钟终于弄出来了， 玩的真是心跳。。。



<https://repl.it/repls/BusyLowClimate>

这个是我自己写的，写的挺烂的，不过可以参考参考，有什么更好的方法也可以讨论。

---

## OA3

---

刚刚做完的亚麻OA3，分为work simulation 和逻辑题两大部分，work simulation分为5个不同的场景，有4个小时的时间可以做，可以说时间非常充裕了。感觉这部分像是在做阅读题，大概就是看员工之间的对话和邮件，然后回答问题，把提供的解决方案拖到5个框框中，分别为highly effective, very effective, moderately effective, slightly effective, ineffective，好纠结，见仁见智吧，感觉没有个标准答案。我做的这几个题小土刀的资料里没有，感觉像是新题，但是不难，大家不必紧张。第二部分就是传统的逻辑题，考之前刷一遍小土刀的资料就行，不过注意应用题可能题干部分会改，别照搬小土刀的答案，仔细审题就行。

---

刚刚做完OA3，回馈地里积福积福，希望有机会能拿到VO

work simulation真的蛮难的，记得有一道是看两个algorithm哪个好，4h  
logical时间很紧张，遇到了六桌问题还有快递费那个，35min

---

OA3分为两部分 Work Simulation 和 逻辑题。Work Simulation有四个小时，时间充足，可以慢慢做。逻辑题只有35分钟，时间比较紧迫。Work Simulation 有五个部分，每个部分两三题。模拟工作场景，遇到问题怎么解决。逻辑题类似于小学奥数题，小土刀有很多例子可以参考。（很多跟alphabet相关的！）

---

想分享一下自己准备亚麻bq过程的一些心得 最后有附上我整理的问题列表

我看了地理大部份亚麻的面经 很多人都有提到bq要好好准备/问很多bq 不过分享细节的文章比较少 所以抛砖引玉一下

首先如果你是非码农的同学 基本上bq的数量会是码农的两三倍左右 因为不像码农通常一轮45分钟的面试会有30分钟的coding 加上15分钟的 bq, 我自己跟朋友遇到的经验都是100% bq 不要怀疑 就是五轮满满的bq 所以只准备4-5个故事肯定不够 以下分几点说明

- 要准备几个故事: 我认为大概需要准备25-30个 projects/stories 其中大概8个大型的完整故事 所谓大型故事就是 1. 问题够困难 2. 结果够明确 3. 由你主导或贡献明显 通常来说如果顺着STAR 说完需要花一分半到两分钟以上的就是大型故事 在五轮的面试裡面通常一轮会说3-5个故事 你的目标要订在每个大型故事最多重複两次 然后所有的小故事都只用一次 因为之后所有面试官会开会讨论 你讲最多故事就能给committee提供越多你的过去经验 这点对社招的同学尤其重要

- Leadership Principles: LP的重要性不用多说 不过我自己最后的感想是在分类故事的时候不要照着LP去分 因为在面试的时候面试官通常不会直白地问你 “Tell me about a time you were frugal” 或 “Tell me about a time you demonstrated customer obsession” 我建议是照着实际的问题去准备 然后对于每个故事(尤其是大型故事) 一定要准备对付follow up questions

- Follow up Questions:几乎每一个故事都会有follow up 通常不出以下几种:
  - o 这个project 是你自己发掘的还是被指派的 怎么发掘的
  - o 过程中有哪些人参与 他们的反应如何 如果反应不好你怎么说服他们 他们对你的说服反应如何
  - o 除了你的解法以外 有没有想过其他解法 为什么选你这个解法
  - o 这个project 除了你提到的immediate results 以外 对公司整体的帮助是什么
  - o 你如何判断这个project 成功了/完成了/失败了
  - o 比较难的follow up, 叫你讲一个跟刚刚完全相反的故事 例如刚刚问你 “Tell me about a time you took risk and succeed” 就变成 “Tell me about a time you took risk and failed” 或者我遇到的 “Tell me about a time that you convinced a group to follow your idea”变成 “Tell me about a time you were convinced by the team and gave up your idea”

基本上面试官只能从你的叙述之中来问follow up 所以一开始在说故事的时候不要把全部细节都说出来 一来故事变得很冗长 二来要留空间给面试官问follow up

- 小抄/笔记: 基本上面试的时候没有什么空挡可以看小抄或做笔记 我觉得唯一需要准备的就是把全部故事的名字写在一张纸上 字体记得要大且清楚 然后面试官在问问题的时候开始用余光在纸上选故事 故事说完以后记得在旁边做个记号提醒自己这个用过了 要这么做的原因是每个人总有几个故事是记得特别熟 在慌张的情况之下就会不自觉讲自己最熟的故事 导致一个故事重複使用多次

- 练习: 如果可能的话 有人能一起mock interview 肯定效果最好 没有的话可以录下自己的回答 一来知道大概讲了多久 二来顺顺口条 练习的时候一定要说出来 切忌看着问题然后只想着恩我知道要用哪个故事来回答 脑中想跟实际说差很多 一定要说出来

最后附上我自己整理的问题列表 这个大概涵盖90%可能会出现的题目 多多少少一定会有没遇到的题目 重要的是故事多准备 遇到问题的時候不要猜这是对应哪个LP 照着字面问题回答就好

觉得这篇有帮助的给我加个米吧....打了好久啊

以下内容需要积分高于 200 您已经可以浏览

- Tell me about yourself/ Tell me about your background
- Why [Amazon](#)

- Why this position
- Why leaving your current company
- Your strength
- Your weakness
- Tell me about a project that your most proud of
- Tell me about the most challenging project you worked on
- Tell me about a project that you had failed
- Tell me about a time you took risk and succeed/failed
- Tell me about a time you have to change the status quo
- Tell me about a time you had to deal with tight deadline and you were able/not able to meet the target
- Tell me about a time you had conflict with your manager/peer/colleagues
- Tell me about a time you come up with a simple solution to a complex problem
- Tell me about a time you come up with a new approach to an old problem
- Tell me about a time you received negative feedback from your manager
- Tell me about a time you made a decision without your manager's approval
- Tell me about a time you had to make a decision when there is not enough data or information
- Tell me about a time you were assigned a project with unclear responsibility
- Tell me about a time you had to sacrifice short term gains for the long term goals
- Tell me about a time you were 50%/75% on a project and found you have made a mistake and have to change direction
- Tell me about a time you came up with a solution that customer didn't ask and they end up like it
- Tell me about a time that customer tell you they want something but you know that's not really what they want
- Tell me about a time you had to convince the team/convinced by the team
- Tell me about a project that you have to overcome big obstacle
- Tell me about an experience that you have to earn trust/gain buy-in from another group
- Tell me about a time you step outside of your job scope and solved a problem
- What is your proudest/biggest innovation
- How do you select metric to measure your project success
- Have you ever learn something new by yourself and end up using what you learn to solve problems at work
- If you have conflicting goals, how do you make trad-off
- Tell me about a time you have to seek outside help to dive deep on a problem to find the solution
- Tell me about a time you decided to take on a project instead of being assigned to you

### **Q1: Schedule the design review meeting (1)**

- 1 - We can take our best guess at an estimate on our own
- 2 - We should work for a couple of days to gauge our progress, and then complete our estimate from there
- 4 - We should consult a coworker who has more relevant experience on this type of task
- 3 - We should conduct our own investigation utilizing online research materials and internal documentation
- 5 - Let's ask our manager how we should go about developing an estimate

### **Q2: Schedule the design review meeting (2)**

- 3 - Ask all parties to identify a back-up person who could meet during a designated time
- 3 - See if there is a backup person on the Localization Team that can meet
- 5 - Set-up videoconferencing to include all POC's regardless of their physical location
- 1 - Agree to postpone the design review for two weeks when all parties have more availability
- 2 - Discuss the design review over email
- 4 - Agree to schedule the meeting at Xavier's location an hour away

### **Q3: Response to Ravi (1)**

- 3 - We should miss the conference and increase the timeline to four weeks because we have four weeks of work
- 4 - Take a day to investigate whether adding additional resources would allow us to meet the original timeline, and re-evaluate afterwards
- 1 - Tell the Localization team if can't be done in the timeline, so we should go ahead with the US launch and delay the global launch even though it means adding an additional week of effort to the four week estimate
- 5 - Take two weeks to create a prototype of the feature to demo at the conference, then take the additional two weeks needed to fully complete the feature
- 2 - We can still hit the two week deadline without any changes by working harder and putting in overtime

### **Q4: Response to Ravi (2)**

Begin your investigation using the old error logs, but tell Ravi he will need to run the new logs if the old logs aren't useful

### **Q5: Response to Aaron and Jacob (1)**

Can you tell me more about what you're talking about?

### **Q6: Response to Aaron and Jacob (2)**

You said we have an internal database of both digital and physical books. How did we get the physical book data if the Book Database API doesn't give it to us?

### **Q7: Response to Aaron and Jacob (3)**

I recommend you go with Jacob's solution. We should miss the deadline to build our own service and meet all the requirements.

### **Q8: Roadmap**

Since you know more about the programming language than anyone else, you revise the estimate for porting to Java.



**Q9: Response to Nadia**

What were the internal test case results?

**Q10: Most likely cause of German language issue**

Site is using proxy server location to determine displayed language

**Q11: Most likely cause of invalid recommendation issue**

Database field storing username is too short

**Q12: Log trace investigation success**

- 5 - Increase time allotted for testing in overall lifecycle
- 5 - Update automated end-to-end tests to include broader data coverage
- 3 - Write more unit tests to include edge cases
- 3 - Have team members perform more manual testing before checking code in
- 1 - Increase the size of QA team
- 1 - Have more user testing in beta phase

**Q13: Response for meeting the deadline**

- 2 - Work on the project on your own, putting in extra effort to finish on time
- 3 - Work on the project on your own until Priya is available, then continue to work on it together
- 4 - Work on the project with Ben, being sure to watch his work closely because of his lack of experience
- 5 - Tell your manager you will not be able to complete the project in the time available
- 1 - Cut features from the product so you will be able to meet the two week deadline
- 3 - Start working on the project right away with Ben. Then ask Priya to contribute what she can when she is available

**Q14: Response for completing this work on time**

- 4 - Work with the Customer Incentives Team to identify the critical features that they need by the deadline, and focus on those
- 2 - Push the timeline back another week to ensure there is enough time for all work to be completed accurately
- 3 - Ask your whole team for help, explaining the urgency that another team is blocked
- 5 - Ask your manager for help in determining the best approach to meet the new deadline
- 1 - Put in extra hours yourself to make sure everything gets done on time

**Q15: Upgrade**

- 4 - We should not perform this upgrade at this point in time. We promised the Retail Website Team we would have their new features complete by the proposed deadline. Let's postpone the upgrade to another time
- 2 - We should not perform the upgrade because it will not have a significant impact on the Retail Website Team's experience. We should focus on the Retail Website Team's requests
- 3 - We should not perform this upgrade at this point in time. Our top focus is meeting our agreed upon commitment with the Retail Website Team, so we should finish that first. We can focus on the upgrade afterwards by pushing our deadlines for some of our other projects
- 1 - I think we should perform the upgrade. The right thing to do is push back on the Retail Website Team because it will keep our team from having to do the same work twice

5 - I think we should perform the upgrade. As a compromise, we can include the gift recommendation feature the Retail Website Team wants by the deadline and then complete the upgrade. We can finish the seasonal-based gift recommendations feature after the deadline

2 - I think we should perform the upgrade. The right thing to do is push back on the Retail Website Team because it will allow us to more efficiently serve the customer and the customer will be helped in the long run.

#### **Q16: New product design**

2 - A, C, D, G

1 - A, C, D, G, H

4 - A, B, D

3 - A, C, F

5 - A, D, F

3 - F, G

#### **Q17: ?**

?

#### **Q18: Problem with Product.wasPurchasedByUser()**

It has performance issue

#### **Q19: Most effective way of improving ShoppingCart()**

Change the design of ShoppingCart by removing ShoppingCart user and making shopping cart a property of User instead

#### **Q20: Five tests within ShoppingCartTest()**

Fail - Test1

Pass - Test2

Fail - Test3

Pass - Test4

Fail - Test5

#### **Q21: Ask Jacob a question**

3 - Do any other projects depend on fixing this problem?

5 - How many customers is this affecting?

5 - How does this affect customers

4 - Are we receiving complaints from customers?

2 - How long will it take to solve this problem?

1 - If I help you with this problem, will you help me finish my work today?