IBM 2023-2024 - General Software Developer Coding Assessment

- IBM General Software Developer 2023 - 2024 Intern OA 原题

60 mins 两道 coding

1. Question 1

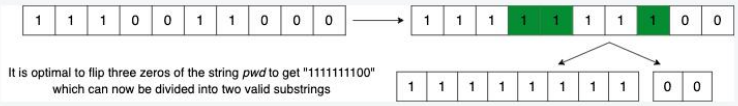
A password string, *pwd,* consists of binary characters (0s and 1s). A cyber security expert is trying to determine the minimum number of changes required to make the password secure. To do so, it must be divided into substrings of non-overlapping, even length substrings. Each substring can only contain 1s or 0s, not a mix. This helps to ensure that the password is strong and less vulnerable to hacking attacks.

Find the minimum number of characters that must be flipped in the password string, i.e. changed from 0 to 1 or 1 to 0 to allow the string to be divided as described.

Note: A *substring* is a contiguous sequence of characters in a string.

Example

Given *pwd* = "1110011000"



The two substrings have lengths 8 and 2 respectively. Since both lengths are even, the division is valid. So the answer is 3.

Function Description

Complete the function *getMinFlips* in the editor below.

2. Question 2

Given an array of *n* positive integers, assuming 0-based indexing, its cost is

$$\sum_{i=1}^{len(arr)-1} (arr_i - arr_{i-1})^2$$

len(arr) is the size of the array.

Insert any integer at any location of the array such that the cost of the array is minimized. Find the minimum possible cost of the array after inserting exactly one element.

Example

a = [1, 3, 5, 2, 10]

The cost of the array before insertion = $(1 - 3)^2 + (3 - 5)^2 + (5 - 2)^2 + (2 - 10)^2 = 81$.

Two of many scenarios are shown below.

1. Insert 4 between 3 and 5, cost of array = $(1 - 3)^2 + (3 - 4)^2 + (4 - 5)^2 + (5 - 2)^2 + (2 - 10)^2 = 79$.
2. Insert 6 between 2 and 10, cost of array = $(1 - 3)^2 + (3 - 5)^2 + (5 - 2)^2 + (2 - 6)^2 + (6 - 10)^2 = 49$.

It can be proven that 49 is the minimum cost possible. Return 49.

Function Description

Complete the function *getMinimumCost* in the editor below.

- 60min 两道题 题目在附件里

1. Question 1

There are *n* types of items in a shop's inventory, where the quantity of the i^{th} item is denoted by *quantity[i]*. These items are to be shipped in two consignments, where the first consignment contains items of type *[1, 2, .., j]*, and the second consignment contains the remaining item types, where *j* can be chosen such that $1 \leq j < n$. Note that both consignments must be non-empty, and all items of a type must be in the same consignment.

The shopkeeper wants the item counts in each consignment to be equal. To achieve this, the shopkeeper can perform the following move any number of times: increase or decrease the quantity of any item type by 1. The quantity of each item type must remain positive throughout.

Find the minimum number of moves in which the total quantities of both consignments can be made equal if the item types are split optimally.

Example:
Consider $n = 3$, *quantity* = [1, 4, 4].
Considering 1-based indexing, increase *quantity[3]* by 1, so *quantity' = [1, 4, 5]*. Partition using *j = 2* and consignments shipped are [1, 4] and [5]. This is optimal, so return the number of operations, 1.

Function Description
Complete the function *getMinimumMoves* in the editor below.

getMinimumMoves has the following parameter:
int quantity[n]: the quantities of each item type

Returns:
long_int: the minimum moves required to make the sums equal in an optimal division

Constraints

- $2 \leq n \leq 3 * 10^5$
- $1 \leq quantity[i] \leq 2 * 10^9$

▼Sample Case 0

Sample Input For Custom Testing

```
STDIN       FUNCTION
-----       --------
5       →   quantity[] size, n = 5
3       →   quantity = [3, 3, 6, 3, 9]
3
6
3
9
```

Sample Output

```
0
```

Explanation

The optimal division is [3, 3, 6] and [3, 9]. Both partitions have an equal sum of quantities so no moves are required.

▼Sample Case 1

Sample Input For Custom Testing

```
STDIN       FUNCTION
-----       --------
3       →   quantity[] size, n = 3
4       →   quantity = [4, 5, 7]
5
7
```

Sample Output

```
2
```

Explanation

It is optimal to make the following moves (1-based indexing):

- Reduce *quantity[1]*, *quantity[]* becomes [3, 5, 7].
- Increase *quantity[3]*, *quantity[]* becomes [3, 5, 8].

2. Question 2

Given an array of *n* positive integers, assuming 0-based indexing, its cost is

$$\sum_{i=1}^{len(arr)-1} (arr_i - arr_{i-1})^2$$

len(arr) is the size of the array.

Insert any integer at any location of the array such that the cost of the array is minimized. Find the minimum possible cost of the array after inserting exactly one element.

Example

a = [1, 3, 5, 2, 10]

The cost of the array before insertion = $(1 - 3)^2 + (3 - 5)^2 + (5 - 2)^2 + (2 - 10)^2 = 81$.

Two of many scenarios are shown below.

1. Insert 4 between 3 and 5, cost of array = $(1 - 3)^2 + (3 - 4)^2 + (4 - 5)^2 + (5 - 2)^2 + (2 - 10)^2 = 79$.
2. Insert 6 between 2 and 10, cost of array = $(1 - 3)^2 + (3 - 5)^2 + (5 - 2)^2 + (2 - 6)^2 + (6 - 10)^2 = 49$.

It can be proven that 49 is the minimum cost possible. Return 49.

Function Description

Complete the function *getMinimumCost* in the editor below.

getMinimumCost has the following parameter:

int arr[n]: an array of integers

Returns

long_int: the minimum possible cost of the array after inserting one element

▼Sample Case 0

Sample Input For Custom Testing

```
STDIN       FUNCTION
-----       --------
4       →   n = 4
4       →   arr = [4, 7, 1, 4]
7
1
4
```

Sample Output

```
36
```

Explanation

The cost of the array before insertion = $(4 - 7)^2 + (7 - 1)^2 + (1 - 4)^2 = 54$.

Insert 5 between 4 and 7, cost = $(4 - 5)^2 + (5 - 7)^2 + (7 - 1)^2 + (1 - 4)^2 = 50$.

Insert 4 between 7 and 1, cost = $(4 - 7)^2 + (7 - 4)^2 + (4 - 1)^2 + (1 - 4)^2 = 36$.

▼Sample Case 1

Sample Input For Custom Testing

```
STDIN       FUNCTION
-----       --------
3       →   n = 3
4       →   arr = [4, 7, 7]
7
7
```

Sample Output

```
5
```

Explanation

The cost of the array before insertion is $(4 - 7)^2 + (7 - 7)^2 = 9$.

Insert 5 between 4 and 7, cost = $(4 - 5)^2 + (5 - 7)^2 + (7 - 7)^2 = 5$.

Insert 6 between 4 and 7, cost = $(4 - 6)^2 + (6 - 7)^2 + (7 - 7)^2 = 5$.

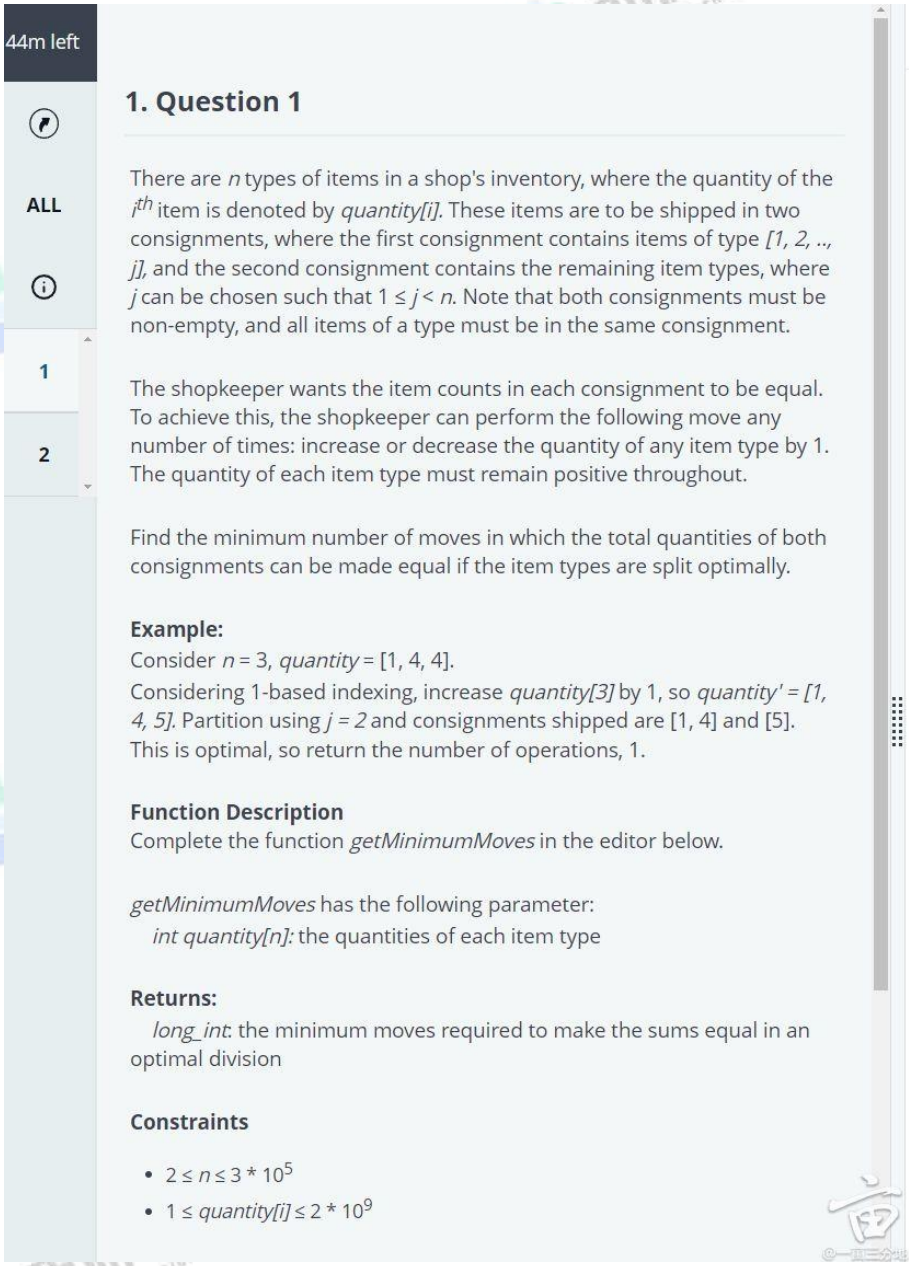
讨论区：

-楼主能求教下第一题思路吗

-使用一个前缀和的数组去做记录

-可以尝试用 prefix sum 的方法来做

- 一共俩道题，一道 leetcode，一道 sql



▶ Input Format For Custom Testing

▼ Sample Case 0

Sample Input For Custom Testing

```
STDIN       FUNCTION
-----       --------
5       →   quantity[] size, n = 5
3       →   quantity = [3, 3, 6, 3, 9]
3
6
3
9
```

Sample Output

```
0
```

Explanation

The optimal division is [3, 3, 6] and [3, 9]. Both partitions have an equal sum of quantities so no moves are required.

▼ Sample Case 1

Sample Input For Custom Testing

```
STDIN       FUNCTION
-----       --------
3       →   quantity[] size, n = 3
4       →   quantity = [4, 5, 7]
5
7
```

Sample Output

```
2
```

Explanation

It is optimal to make the following moves (1-based indexing):

- Reduce *quantity[1], quantity[]* becomes [3, 5, 7].
- Increase *quantity[3], quantity[]* becomes [3, 5, 8].

The consignment can be divided as [3, 5] and [8].

2. Question 2

A domain name registration service needs new reporting functionality. Create a query that returns a list of all active accounts, the number of unexpired domain names they have, and the nearest expiration date later than July 15, 2022.

The result should have the following columns: *username* | *domains* | *nearest_expiration*.

- *username* - account username
- *domains* - total number of domains for a specific account
- *nearest_expiration* - nearest expiration date for domains for a specific account

The result should be sorted in ascending order by *username*.

Note:

- Only active accounts should be included in the report.
- Only domain names that have not expired (with an expiration date greater than today) should be included in the report.
- Today is July 15, 2022.

▼ Schema

accounts		
name	**type**	**description**
id	SMALLINT	unique id, primary key
username	VARCHAR(255)	
is_active	SMALLINT	Account status: 1 = Active, 0 = Not active

domains		
name	**type**	**description**
account_id	SMALLINT	foreign key, accounts.id
name	VARCHAR(255)	
expiration_date	VARCHAR(19)	

▼ Sample Data Tables

accounts		
id	username	is_active
1	obeedie0	0
2	stopham1	1
3	ndolder2	1
4	jyanshinov3	1
5	ewilflinger4	0

domains		
account_id	name	expiration_date
1	imgur.com	2022-05-14
1	domainmarket.com	2022-07-02
1	comsenz.com	2022-07-28
1	gizmodo.com	2022-08-09
1	toplist.cz	2022-08-15
1	scientificamerican.com	2022-09-03
1	examiner.com	2022-12-18
1	photobucket.com	2023-01-22
2	merriam-webster.com	2022-02-20
2	tripod.com	2022-08-08
3	ca.gov	2022-04-24
3	ehow.com	2022-06-28
3	purevolume.com	2022-07-01
3	squidoo.com	2022-10-27
3	eepurl.com	2022-12-21
4	digg.com	2022-05-14
4	jugem.jp	2022-08-05
4	artisteer.com	2022-10-21
5	behance.net	2022-03-24
5	cnn.com	2022-05-11

▼ Expected Output

username	domains	nearest_expiration
jyanshinov3	2	2022-08-05
ndolder2	2	2022-10-27
stopham1	1	2022-08-08

- 一共就兩題, 60 分鐘作答時間

1. Question 1

A coding competition is being organized on the HackerRank platform. All participants need to be grouped into teams where each team has exactly two candidates and the sums of their skills must be equal for all teams. The *efficiency* of a team is the product of the skills of its members, e.g. for a team with skills [2, 3], the efficiency of the team is 2 * 3 = 6.

Find the sum of the efficiencies of the teams. If there is no way to create teams that satisfy the conditions, return -1.

Notes

- All participants must be assigned to a team.
- The answer is always unique.

Example

The skills of the candidates are *skill = [1, 2, 3, 2].* They can be paired as *[[1, 3], [2, 2]].* The sum of skills for each team is the same, i.e., 4.
The efficiency is computed as:

- Efficiency of [1, 3] = 1 * 3 = 3
- Efficiency of [2, 2] = 2 * 2 = 4

Return the sum of efficiencies, 3 + 4 = 7.

Function Description

Complete the function *getTotalEfficiency* in the editor below.

getTotalEfficiency has the following parameter:
int skill[n]: the skills of each candidate

Returns

long_int: the sum of the efficiencies if it is possible to form the teams, or -1 otherwise

2. Question 2

Two countries "I" and "P" wish to compete in a sports game. The country "I" wants to train the players in groups in order to maximize the strength of the group. Each player has a unique id starting from *1, 2, 3...* and so on. A group starts with one player and continues to grow. Initially the group contains a player with id *x*. This player can call the player with id *y* to join the group if the sum of the factorial of the digits of *x* is equal to *y*. Now the player with id *y* calls the player with id *z* to join the group if the sum of the factorial of the digits of *y* is equal to *z* and this continues to increase the group size.

For example, the player with id *23* can call the player with id *8* to join the group since *8 = 2! + 3!*. Where *x!* denotes factorial of *x*, i.e. *x! = 1 * 2 * 3 * ... x.*

The leader of the group is the player with a maximum id number. The strength of the group will be the id of leader multiplied by the number of players in the group. For example, if the group is *{4, 24, 26, 722, 5044, 169, 363601, 1454}* then its strength will be *363601 * 8 = 2908808.*

Given an integer *n* find the maximum strength of the group which starts with the player with id *n*.

Function Description:

Complete the function *maxStrength* in the editor below. *maxStrength* returns an integer, the maximum strength of the group.

maxStrength has the following parameter:
n: an integer

讨论区：

-求问第二题怎么做？没有最大人数限制的话，可以一直无限循环下去？

-會做到一個點是下一個數字已經出現過，用字典紀錄

- 投的是 2024 SWE intern 岗，两题 60 min 时间很充裕，题目难度不大

第一题地里面已经有了

1. Question 1

There are *n* types of items in a shop's inventory, where the quantity of the i^{th} item is denoted by *quantity[i]*. These items are to be shipped in two consignments, where the first consignment contains items of type *[1, 2, .., j]*, and the second consignment contains the remaining item types, where *j* can be chosen such that $1 \leq j < n$. Note that both consignments must be non-empty, and all items of a type must be in the same consignment.

The shopkeeper wants the item counts in each consignment to be equal. To achieve this, the shopkeeper can perform the following move any number of times: increase or decrease the quantity of any item type by 1. The quantity of each item type must remain positive throughout.

Find the minimum number of moves in which the total quantities of both consignments can be made equal if the item types are split optimally.

Example:
Consider *n* = 3, *quantity* = [1, 4, 4].
Considering 1-based indexing, increase *quantity[3]* by 1, so *quantity' = [1, 4, 5]*. Partition using *j = 2* and consignments shipped are [1, 4] and [5]. This is optimal, so return the number of operations, 1.

Function Description
Complete the function *getMinimumMoves* in the editor below.

getMinimumMoves has the following parameter:
int quantity[n]: the quantities of each item type

Returns:
long_int: the minimum moves required to make the sums equal in

第二题没见过，模拟的是一个 hashmap，顺位的方式就是+1

2. Question 2

Packets are sent to different ports on a computer system based on the hash of their packet ID. The value of the hash is as given below: `Hash = mod (packet_id, numberOfPorts)` where *mod* is the modulus operator and takes the mod of first operand by second operand.

The ports are numbered from *0* to *(number of ports) - 1*, and a packet is initially sent to the port that has the port number equal to the hash of its packet ID. Each port requires *t* seconds to process an arriving packet. If a port is currently processing a packet, any arriving packet is rerouted to the next port number, and so on. The list of ports is circular. If a packet arrives at the last port and it is busy, it is rerouted to the first port. Given a list IDs of *n* packets that arrive 1 per second, find the port to which each packet is finally sent. The first packet is sent at second *t = 1*.

Each port requires a time *t* to send a packet. If a port is currently sending a packet, this packet is then sent to the next port number, and so on. Given a list IDs of *n* packets that arrive *1* per second, find the port to which each packet is finally sent. The first packet is sent at time *t = 1*.

Example

numberOfPorts = 3
transmissionTime = 2
packetIds = [4, 7, 10, 6]

The destination ports, assuming no time conflicts are all calculated as *packetIds[i] modulo numberOfPorts*, so *[1, 1, 1, 0]* in this case. These arrive at times *1, 2, 3, 4*. The first packet is sent to port *1* with no conflicts. Port *1* will be occupied at times *1* and *2* due to the transmission time, so the second packet has a conflict and is sent to port *1 + 1 = 2*. The third packet wants to go to port *1* and arrives at

- 两道都是简单题

1. Question 1

There are *n* types of items in a shop's inventory, where the quantity of the i^{th} item is denoted by *quantity[i].* These items are to be shipped in two consignments, where the first consignment contains items of type *[1, 2, .., j],* and the second consignment contains the remaining item types, where *j* can be chosen such that $1 \leq j < n$. Note that both consignments must be non-empty, and all items of a type must be in the same consignment.

The shopkeeper wants the item counts in each consignment to be equal. To achieve this, the shopkeeper can perform the following move any number of times: increase or decrease the quantity of any item type by 1. The quantity of each item type must remain positive throughout.

Find the minimum number of moves in which the total quantities of both consignments can be made equal if the item types are split optimally.

Example:
Consider *n* = 3, *quantity* = [1, 4, 4].
Considering 1-based indexing, increase *quantity[3]* by 1, so *quantity' = [1, 4, 5].* Partition using *j = 2* and consignments shipped are [1, 4] and [5]. This is optimal, so return the number of operations, 1.

Function Description
Complete the function *getMinimumMoves* in the editor below.

getMinimumMoves has the following parameter:
int quantity[n]: the quantities of each item type

2. Question 2

Consider a string, *S,* that is a series of characters, each followed by its frequency as an integer. The string is not compressed correctly, so there may be multiple occurrences of the same character. A properly compressed string will consist of one instance of each character in alphabetical order followed by the total count of that character within the string.

Example
The string '*a3c9b2c1*' has two instances where '*c*' is followed by a count: once with *9* occurrences, and again with *1.* It should be compressed to '*a3b2c10*'.

Function Description
Complete the function *betterCompression* in the editor below.

betterCompression has the following parameter:
S: a compressed string

Returns:
string: the properly compressed string

Constraints

- *1 ≤ size of S ≤ 100000*
- *'a' ≤ characters in S ≤ 'z'*
- *1 ≤ frequency of each character in S ≤ 1000*

讨论区：

-您好，请问可以说一下第一道题的思路吗？

-将 given array 分成两半，左半边元素之和与右半边元素之和的 diff 最小值。

- 60 分钟 两道题， 一个 code， 一个 SQL

1. Minimize array cost： SUM （arr_i -arr_(i-1))^ 2 insert any interger at any location s.t. the cost (sum) of this array is minimize. Find the minimize cost after insert one element. Leetcode 有答案

2. SQL， 要求 one colume > another, and one colume contain"never"or "once", order by MAC;

- Hacckerank 60 分钟两道题，

第一题就是给你 prefix sum，求原数列，不同的地方是他们的 prefix sum 用的是 xor 不是加，所以就是相邻两个 prefix 求 xor 就是原数列了

第二题是题干比较复杂，大体意思是给定一个数字 a，每一位的阶乘求和是数字 b，然后一直操作直到形成重复，求最大的数字和数量的乘积，标准的 BFS，一个 queue和一个 set 就能解决了，我感觉也有数学一点的做法

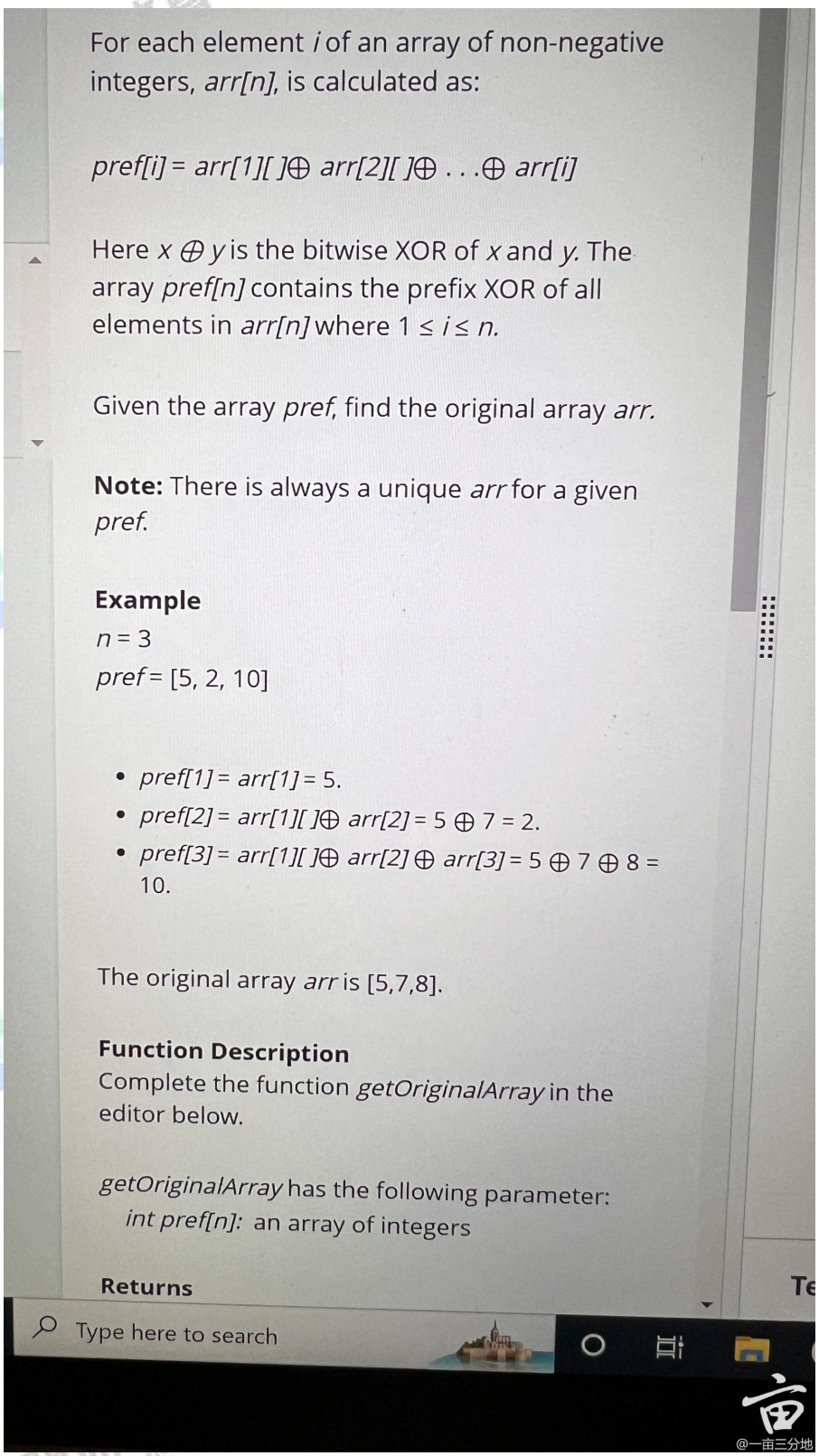
讨论区：

-第二题 a 的阶乘和形成 b，b 的生成 c，重复到什么时候结束呢?

-出现重复数字

-最后会是一个乘法群，数学稍微证一下就好

- General 的 OA，不知道之前有没有出现过 60mins 2 道题 感觉是 medium 的题



For each element *i* of an array of non-negative integers, *arr[n]*, is calculated as:

pref[i] = *arr[1][ ]*⊕ *arr[2][ ]*⊕ . . .⊕ *arr[i]*

Here *x* ⊕ *y* is the bitwise XOR of *x* and *y*. The array *pref[n]* contains the prefix XOR of all elements in *arr[n]* where 1 ≤ *i* ≤ *n*.

Given the array *pref*, find the original array *arr*.

Note: There is always a unique *arr* for a given *pref*.

Example

n = 3

pref = [5, 2, 10]

- *pref[1]* = *arr[1]* = 5.
- *pref[2]* = *arr[1][ ]*⊕ *arr[2]* = 5 ⊕ 7 = 2.
- *pref[3]* = *arr[1][ ]*⊕ *arr[2]* ⊕ *arr[3]* = 5 ⊕ 7 ⊕ 8 = 10.

The original array *arr* is [5,7,8].

Function Description

Complete the function *getOriginalArray* in the editor below.

getOriginalArray has the following parameter:
int pref[n]: an array of integers

Returns

Type here to search

Two countries "I" and "P" wish to compete in a sports game. The country "I" wants to train the players in groups in order to maximize the strength of the group. Each player has a unique id starting from *1, 2, 3...* and so on. A group starts with one player and continues to grow. Initially the group contains a player with id *x*. This player can call the player with id *y* to join the group if the sum of the factorial of the digits of *x* is equal to *y*. Now the player with id *y* calls the player with id *z* to join the group if the sum of the factorial of the digits of *y* is equal to *z* and this continues to increase the group size.

For example, the player with id *23* can call the player with id *8* to join the group since *8 = 2! + 3!*. Where *x!* denotes factorial of *x*, i.e. *x! = 1 * 2 * 3 * ... x*.

The leader of the group is the player with a maximum id number. The strength of the group will be the id of leader multiplied by the number of players in the group. For example, if the group is *{4, 24, 26, 722, 5044, 169, 363601, 1454}* then its strength will be *363601 * 8 = 2908808*.

Given an integer *n* find the maximum strength of the group which starts with the player with id *n*.

Function Description:

- IBM 新鲜 OA，求加米看面经！！！

两题都挺简单的，感觉属于 easy

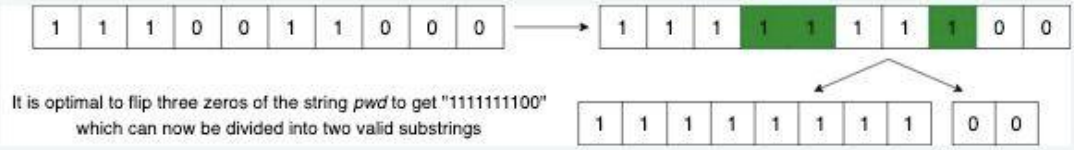
A password string, *pwd,* consists of binary characters (0s and 1s). A cyber security expert is trying to determine the minimum number of changes required to make the password secure. To do so, it must be divided into substrings of non-overlapping, even length substrings. Each substring can only contain 1s or 0s, not a mix. This helps to ensure that the password is strong and less vulnerable to hacking attacks.

Find the minimum number of characters that must be flipped in the password string, i.e. changed from 0 to 1 or 1 to 0 to allow the string to be divided as described.

Note: A *substring* is a contiguous sequence of characters in a string.

Example

Given *pwd* = "1110011000"



The two substrings have lengths 8 and 2 respectively. Since both lengths are even, the division is valid. So the answer is 3.

Function Description

Complete the function *getMinFlips* in the editor below.

getMinFlips takes the following arguments:

str pwd: the binary string

Returns

int: the minimum number of flips to make the division possible

Constraints

- $2 \le pwd \le 10^5$
- The length of *pwd* is even.
- *pwd* contains only 1s and 0s.

2. Question 2

Consider a string, *S*, that is a series of characters, each followed by its frequency as an integer. The string is not compressed correctly, so there may be multiple occurrences of the same character. A properly compressed string will consist of one instance of each character in alphabetical order followed by the total count of that character within the string.

Example

The string '*a3c9b2c1*' has two instances where '*c*' is followed by a count: once with *9* occurrences, and again with *1*. It should be compressed to '*a3b2c10*'.

Function Description

Complete the function *betterCompression* in the editor below.

betterCompression has the following parameter:

S: a compressed string

Returns:

string: the properly compressed string

Constraints

- *1 ≤ size of S ≤ 100000*
- *'a' ≤ characters in S ≤ 'z'*
- *1 ≤ frequency of each character in S ≤ 1000*