Leet Code Weekly 262

Grey Wang

1. Two Out of Three

Description

Given three integer arrays, return a **distinct** array containing all the values that are present in **at least two** out of three arrays. You may return the values in **any order**.

Input

nums1 = [1, 1, 3, 2]

nums2 = [2, 3]

nums3 = [3]

Output

[3, 2]

Contraints

- **1 <= nums1.length, nums2.length, nums3.length <= 100**
- **1 <= nums[i] <= 100**

1. Two Out of Three

Description

Given three integer arrays, return a **distinct** array containing all the values that are present in **at least two** out of three arrays. You may return the values in **any order**.

Input

nums1 = [1, 1, 3, 2]

nums2 = [2, 3]

nums3 = [3]

Output

[3, 2]

Contraints

- **1 <= nums1.length, nums2.length, nums3.length <= 100**
- **1 <= nums[i] <= 100**

如何解一道题?

1. 读题
2. 理解数据
3. 数据规模?
4. 思路?
5. 复杂度?
6. Coding

1. Two Out of Three

Description

Given three integer arrays, return a **distinct** array containing all the values that are present in **at least two** out of three arrays. You may return the values in **any order**.

Input

nums1 = [1, 1, 3, 2]

nums2 = [2, 3]

nums3 = [3]

Output

[3, 2]

Contraints

- **1 <= nums1.length, nums2.length, nums3.length <= 100**
- **1 <= nums[i] <= 100**

思路1 朴素枚举 O(n^2)

```
has(nums, i):
    for num in nums:
        if num == i:
            return 1;
    return 0;


for i := 1..100:
    if has(nums1, i) + has(nums2, i) + has(nums3, i) >= 2:
        i in result
```

1. Two Out of Three

Description

Given three integer arrays, return a **distinct** array containing all the values that are present in **at least two** out of three arrays. You may return the values in **any order**.

Input

nums1 = [1, 1, 3, 2]

nums2 = [2, 3]

nums3 = [3]

Output

[3, 2]

Contraints

- **1 <= nums1.length, nums2.length, nums3.length <= 100**
- **1 <= nums[i] <= 100**

思路2 预处理(哈希) + 枚举 O(n)

```
make_hash(nums):
    h = [0 for i in 1..100]
    for i in nums:
        h[i] = 1;
    return h;


h1 := make_hash(nums1)
h2 := make_hash(nums2)
h3 := make_hash(nums3)


for i := 1..100:
    if h1[i] + h2[i] + h3[i] >= 2:
        i in result
```

2. Minimum Operations to Make a Uni-Value Grid

Description

Given a 2D integer grid of m * n and an integer x. In an operation, you can **add** x to or **subtract** x from any element in the grid.

A **uni-value grid** is a grid where all the elements of it are equal.

Return the **minimum** number of operations to make the grid **uni-value**. If it is not possible, return -1.

Input

grid = [[2, 4], [6, 8]]

x = 2

Output

4

Contraints

- **m == grid.length**
- **n == grid[i].length**
- **1 <= m, n <= 10^5**
- **1 <= m * n <= 10^5**
- **1 <= x, grid[i][j] <= 10^4**

2. Minimum Operations to Make a Uni-Value Grid

Description

Given a 2D integer grid of m * n and an integer x. In an operation, you can **add** x to or **subtract** x from any element in the grid.

A **uni-value grid** is a grid where all the elements of it are equal.

Return the **minimum** number of operations to make the grid **uni-value**. If it is not possible, return -1.

Input

grid = [[2, 4], [6, 8]]

x = 2

Output

4

Contraints

- **m == grid.length**
- **n == grid[i].length**
- **1 <= m, n <= 10^5**
- **1 <= m * n <= 10^5**
- **1 <= x, grid[i][j] <= 10^4**

解题的常见角度?

1. **题目类型? 筛选常见策略**

   最优解问题：搜索，动态规划，枚举答案，贪心。。。

2. **数据规模? 排除一些策略。**

   10^5 个元素，搜索不太可能。。

   10^4的值域 + 10^5个元素，状态空间太大，DP不太可能。。

   枚举10^4的值域 * 10^5个元素的验证，枚举答案不太可行。。

   要不试试贪心?

3. **问题简化?**

   二维跟一维有区别吗? 没有。

4. **边界条件? 由简单再到一般化。**

   [a] => 0

   [a, b] => |b - a| / x

5. **构造贪心策略?**

2. Minimum Operations to Make a Uni-Value Grid

Description

Given a 2D integer grid of m * n and an integer x. In an operation, you can **add** x to or **subtract** x from any element in the grid.

A **uni-value grid** is a grid where all the elements of it are equal.

Return the **minimum** number of operations to make the grid **uni-value**. If it is not possible, return -1.

Input

grid = [[2, 4], [6, 8]]

x = 2

Output

4

Contraints

- **m == grid.length**
- **n == grid[i].length**
- **1 <= m, n <= 10^5**
- **1 <= m * n <= 10^5**
- **1 <= x, grid[i][j] <= 10^4**

思路 中位数 O(N^log N)

```
nums := sort(flatten(grid))

uni_val := nums[nums.size() / 2]

ans := 0

for t in nums:

    if abs(uni_val - t) % x != 0:

        impossible!

    else

        ans += abs(uni_val - t) / x
```

3. Stock Price Fluctuation

Description

Given a stream of **records** about a particular stock. Each record contains a **timestamp** and the corresponding **price** at that timestamp. Some records may be incorrect, another record may appear later in the stream **correcting** the price of the previous wrong record.

Design the ***StockPrice*** class that:

- ***update(int timestamp, int price);***
- ***int current();*** *// Returns the* ***latest price*** *of the stock*
- ***int maximum();*** *// Returns the maximum price*
- ***int minimum();*** *// Returns the minimum price*

Contraints

- **1 <= timestamp, price <= 10^9**
- **At most 10^5 calls made in total.**
- current, maximum, and minimum will be called **only after** update has been called at least once.

1. 题目类型?

纯数据结构题。

2. timestamp => price 映射关系，Map

3. maximum / minimum 优先队列?

3. Stock Price Fluctuation

Description

Given a stream of **records** about a particular stock. Each record contains a **timestamp** and the corresponding **price** at that timestamp. Some records may be incorrect, another record may appear later in the stream **correcting** the price of the previous wrong record.

Design the ***StockPrice*** class that:

- ***update(int timestamp, int price);***
- ***int current();*** *// Returns the* ***latest price*** *of the stock*
- ***int maximum();*** *// Returns the maximum price*
- ***int minimum();*** *// Returns the minimum price*

Contraints

- **1 <= timestamp, price <= 10^9**
- **At most 10^5 calls made in total.**
- current, maximum, and minimum will be called **only after** update has been called at least once.

```
思路1 平衡树 + 优先队列 O(n * log(n))

last_timestamp := -INF

prices := map<int, int>()

q_max = max_priority_queue<record>()

q_min = min_priority_queue<record>()


update(timestamp, price):
    last_timestamp := max(last_timestamp, timestamp)
    prices[timestamp] = price;
    q_max.insert(record(timestamp, price))
    q_min.insert(record(timestamp, price))
current():
    prices[last_timestamp]
maximum():
    while record = q_max.pop()
        if prices[record.timestamp] == record.price:
            return record.price
minimum():
    while record = q_min.pop()
        if prices[record.timestamp] == record.price:
```

4. Partition Array Into Two Arrays to Minimize Sum Difference

Description

Given an integer array nums of 2 * n integers. You need to partition nums into two arrays of length n to **minimize the absolute difference** of the **sums** of the arrays. To partition nums, put each element of nums into one of the two arrays.

Return the **minimum** possible absolute difference.

Input

[3, 9, 7, 3]

Output

|(3 + 9) - (7 + 3)| = 2

Contraints

- **1 <= n <= 15**
- **nums.length == 2 * n**
- **-10^7 <= nums[i] <= 10^7**

4. Partition Array Into Two Arrays to Minimize Sum Difference

Description

Given an integer array nums of 2 * n integers. You need to partition nums into two arrays of length n to **minimize the absolute difference** of the **sums** of the arrays. To partition nums, put each element of nums into one of the two arrays.

Return the **minimum** possible absolute difference.

Input

[3, 9, 7, 3]

Output

|(3 + 9) - (7 + 3)| = 2

Contraints

- **1 <= n <= 15**
- **nums.length == 2 * n**
- **-10^7 <= nums[i] <= 10^7**

解题的常见角度?

1. **题目类型? 筛选常见策略**

   最优解问题：搜索，动态规划，枚举答案，贪心。。。

2. **数据规模? 排除一些策略。**

   15 * 2 个元素，大概率可以搜索。。

   2 * 10^7的值域 * 2^30 的集合状态，DP不太可能。。

   枚举答案? 不确定。。

   贪心? 不确定。。

3. **问题简化?**

   选定n个元素的和为 now, 那么另一半元素为 sum - now，对应当前解为: |sum - 2 * now|

4. Partition Array Into Two Arrays to Minimize Sum Difference

Description

Given an integer array nums of 2 * n integers. You need to partition nums into two arrays of length n to **minimize the absolute difference** of the **sums** of the arrays. To partition nums, put each element of nums into one of the two arrays.

Return the **minimum** possible absolute difference.

Contraints

- **1 <= n <= 15**
- **nums.length == 2 * n**
- **-10^7 <= nums[i] <= 10^7**

思路1. 朴素搜索 O(2^2n) -> 2^30 = 2 * 10^9 超时

```
n := nums.size() / 2

sum := nums.sum()

ans := +INF

dfs(now, cnt, i):

    if cnt == n:

        ans := min(ans, abs(sum - now * 2))

    else if i < 2 * n:

        if cnt < n:

            dfs(now + nums[i], cnt + 1, i + 1)

        dfs(now, cnt, i + 1)

dfs(0, 0, 0)
```

Tip: 算法比赛里，10^8一般可视为1秒

4. Partition Array Into Two Arrays to Minimize Sum Difference

Description

Given an integer array nums of 2 * n integers. You need to partition nums into two arrays of length n to **minimize the absolute difference** of the **sums** of the arrays. To partition nums, put each element of nums into one of the two arrays.

Return the **minimum** possible absolute difference.

Contraints

- **1 <= n <= 15**
- **nums.length == 2 * n**
- **-10^7 <= nums[i] <= 10^7**

思路2. 双向搜索 + 二分查找 O(2^n * log(2^n)) -> O(n * 2^n)

Tip: 降低搜索深度

```
n := nums.size() / 2
sum := nums.sum()
dfs(now, cnt, i, nums, &values):
    values[cnt].add(now)
    if i < n:
        if cnt < n:
            dfs(now + nums[i], cnt + 1, i + 1, nums, &values)
        dfs(now, cnt, i + 1, nums, &values)

l_values = [{} for i in 0..n]
r_values = [{} for i in 0..n]
dfs(0, 0, 0, nums, l_values)
dfs(0, 0, 0, nums + n, r_values)

ans := min(abs(sum - 2 * l_values[0][0]), abs(sum - 2 * r_values[0][0]))
for i in 1..(n - 1):
    for l_value in l_values[i]:
        find nearest r_value to (sum / 2 - l_value) in r_values[n - i]
        ans := min(abs(sum - l_value - r_value))
```

4. Partition Array Into Two Arrays to Minimize Sum Difference

Description

Given an integer array nums of 2 * n integers. You need to partition nums into two arrays of length n to **minimize the absolute difference** of the **sums** of the arrays. To partition nums, put each element of nums into one of the two arrays.

Return the **minimum** possible absolute difference.

Contraints

- **1 <= n <= 15**
- **nums.length == 2 * n**
- **-10^7 <= nums[i] <= 10^7**

```
思路3. 状态压缩 + 双向搜索 + 二分查找 O(n * 2^n)

n := nums.size() / 2

sum := nums.sum()


l_values = [{} for i in 0..n]

r_values = [{} for i in 0..n]

for mask in 0..(1 << n):

    cnt := l_value := r_value := 0

    for i in 0..n:

        if mask & (1 << i):

            cnt += 1

            l_value += nums[i]

            r_value += nums[i + n]

    l_values[cnt].add(l_value)

    r_values[cnt].add(r_value)


ans := min(abs(sum - 2 * l_values[0][0]), abs(sum - 2 * r_values[0][0]))

for i in 1..(n - 1):

    for l_value in l_values[i]:

        find nearest r_value to (sum / 2 - l_value)
```

Tip: 状态压缩，将集合状态表示成比特位的形式

例：{3, 9, 7, 3} 选取 {3, 9}
可表示为：1100 = 3