彰化高中 114學年度資訊學科能力競賽 校內複賽 題解

A. 方差緊密度 *Variance Tightness*

subtask1

枚舉切點的位置，預處理一下前綴 / 後綴和即可
時間複雜度為 $O(N)$

subtask2

因為 $N \leq 20$，我們可以爆搜 $N-1$ 個切點是否要切，然後花線性的時間判斷是否合法和取最大值
時間複雜度為 $O(2^N N)$

subtask3

定義 $dp[i][k]$ 為 A 的前綴 $A[1..i]$ 切分成 K 段的「方差緊密度和」最大值
轉移式：$dp[i][k] = \max_{1 \leq j < i}; dp[j][k-1] + R(j+1, i)$
時間複雜度為 $O(N^3)$

subtask4

定義 $C(i,j) = -R(i,j)$，發現到 C 滿足 Monge Condition：

$$C(i,j) + C(i+1,j+1) \leq C(i+1,j) + C(i,j+1)$$

證明如下：

有一恆等式：

$$N\sum_{i=1}^{N} x_i^2 - (\sum_{i=1}^{N} x_i)^2 = \sum_{1 \leq p < q \leq N} (x_p - x_q)^2$$

因此我們可以得到：

$$C(i,j) = \sum_{i \leq p < q \leq j} (A_p - A_q)^2$$

考慮差值：

$$\Delta = (C(i+1,j) + C(i,j+1)) - (C(i,j) + C(i+1,j+1)) \;=\; \qquad (A_i - A_{j+1})^2 \geq 0$$

移項即得：

$$C(i,j) + C(i+1,j+1) \leq C(i+1,j) + C(i,j+1)$$

因為轉移單調，我們可以透過分治優化在 $O(NK\log N)$ 或 SMAWK 在 $O(NK)$ 內解決

B. 伴手禮 *Souvenir*

subtask1

騙分用，`cout<<rand() % 2;` 都會過

因為題目說不是 **長邊都相等** ，就是 **寬邊都相等** ，所以答案不是 0 就是 1 ~~，所以 `rand()` 幾次就過了~~ ，所以 `cout<<(n == 0 ? 0 : 1);` 就過了

~~時間複雜度~~ $O(1)$

subtask2

唬爛用， $O(2^n)$ 都會過

subtask3

因為 $a_i, b_i \leq 100,\ \forall\ 1 \leq i \leq n$ ，所以行李們 `unique` 後頂多才 10000 多個。

於是 `unique` 後 **滾動 DP** 就行了ouob

時間複雜度 $O(n^2)$

subtask4

因為 $n \leq 2000$ ，~~看起來就很~~ n^2 ，所以我們快樂地 `普通 DP` 就會過了 (不用滾動，甚至比 ***subtask3*** 簡單)

時間複雜度 $O(n^2)$

subatsk5

因為題目說大行李要放在小行李下面，換句話說就是 **行李們的高度需要嚴格遞減** ，看起來跟 ***LIS*** 相當像。

於是我們套用 **偏敘** 的思維，首先對長邊做一次 ***sort*** ，這樣就可以確保我們等等依次遍歷的時候， **先走到的長邊一定可以放在後走到的長邊下面** ，但是寬邊呢?

我們只需要找出在這種情況下，寬邊的 ***LIS*** ，即是答案了，而理由很簡單，就是之前的 ***sort*** 。

理論上到這裡就已經結束了，但是還沒完，~~因為本人出題時無聊多加了個條件~~ ，就是 **只要長邊或寬邊相等** ，就無法疊起來。

這直接讓這題又上升了一個難度。 首先我們需要理解模板解，透過不斷地二分搜來找到最終答案，於是我們對這個解法做一點點修改 - - 對於相同長邊，只更新數列一次，就可以獲得可愛的 ***AC*** 了 ouob

> 當然你也可以用 `custom sort` 但我太笨了沒有想到QAQ

時間複雜度 $O(n \cdot \log_2^n)$

C. 希格瑪 *Sigma*

subtask1

模擬，~~有手就會寫~~。

subtask2

簡單舉幾個例子就會發現很多 $\lfloor\sqrt{K}\rfloor$ 會重複。
像是 $\lfloor\sqrt{1}\rfloor = \lfloor\sqrt{2}\rfloor = \lfloor\sqrt{3}\rfloor = 1$，$\lfloor\sqrt{4}\rfloor = \lfloor\sqrt{5}\rfloor = \ldots = \lfloor\sqrt{8}\rfloor = 2$
所以與其窮舉 K，不如窮舉 $\lfloor\sqrt{K}\rfloor = L$，並計算有幾個數開根號為 L (公式：$(L+1)^2 - L^2$)。
舉例來說，$\lfloor\sqrt{K}\rfloor = 2$ 的數有 $3^2 - 2^2 = 5$ 個。

提供一個視覺化的想法，可以畫出 $y = \lfloor\sqrt{x}\rfloor$ 的函數。
其圖上長方形的面積 (跟 x 軸圍出來的)，即為所求 (注意 L = N 只有一個數)。

subtask3

把 subtask2 的解法用 sigma 表示，之後用 sigma 平方公式化減，即可得出 $O(1)$ 的公式解。
須注意由於用到除法，所以模除要使用模逆元，不能直接模除上去。

D. 切糕 *Qiegao*

經典的 Sliding Window Median。
偷來的題解

E. 質數背包 *Prime Knapsack*

subtask1

轉換成背包問題，~~因為題目都提示能用背包子~~。
窮舉 1000 以內的質數，之後就當成無限背包問題解，~~給 90 分真的是佛心來著~~。

subtask2

著名的哥德巴赫猜想，任意大於二的偶數皆可用兩個質數組成。
對於奇數，則**至多**需要三個質數。
由於兩數相加為奇數，其充要條件為一數為奇另一數為偶。
眾所皆知，只有 2 是偶數的質數，故當一個奇數能用兩個質數相加表示，其中一個數必為 2。

故我們只需簡單的 if-else，就能解出來。
所以如果 N 是質數，則輸出 1。
如果 N 是偶數，又或者 N - 2 為質數，則輸出 2。
否則輸出 3。

~~你說如果遇到無解的情況怎麼辦？~~
~~那恭喜你獲得了一百萬美金。~~

subtask3

質數判斷怎麼 $O(\log N)$？
肯定是米勒拉賓直接砸下去。
~~這 1 分真難拿~~

F. 最佳訊號路徑 *Optimal Signal Path*

subtask1

從基地台開始跑 dijkstra。

subtask2

建一個虛擬的原點，之後將其與每個基地台連一條權重為 0 的邊，之後從虛擬的點開始跑 dijkstra，即可得出每個點最近的基地台。

subtask3

知道每個最近的基地台了，那要如何找到一條兩點間的路徑其離基地台最遠的點是最小的。
不如試試對答案二分搜，每次詢問搜答案，再用 dfs 跑一遍圖，這樣即可用 $O(QN\log w)$ 的時間解決這問題。

subtask4

怕有人不會 dijkstra 但會樹上換根 dp，故開此子題。
~~雖然好像沒甚麼意義~~

subtask5

最遠的點最小，是不是很有瓶頸路的感覺？
不過我們找出的是每個點離最近基地台的距離，其為點權而非邊權，沒辦法直接跑最小生成樹。
但不難發現，當我們經過一條路，事實上就是會踩到兩端點，而我們只在乎離基地台最遠的點，故邊權就設成兩端點的點權取 max。
接下來就是開心的跑 Kruskal 將圖重構成樹，再來 lca 找瓶頸。

這題算是實作偏難的題目，寫出來全國賽應該有拿到三等獎的實力。
寫不出來的也別氣餒，多練習就寫得出來了。ouobbb

> 這題有另解，詳見 ***Testers' Code***

G. 神秘交叉 *Intersect*

題目要求我們找出序列中的 **最大連續 XOR 值** 。
~~題目長那樣純粹是我忘記打題敘~~

subtask1

$n \leq 1000$ ，這數字一看就很 $O(n^2)$ ，但是要怎麼做呢?

我們先嘗試構造暴力解，首先，枚舉 **左界** 和 **右界** ，然後每次從 **左界** 一個一個 ***XOR*** 到 **右界** ，不斷更新最大值，最後獲得答案。

這樣的算法複雜度為 $O(n^3)$ ，顯然跑不完。

於是我們想到 ***XOR*** 是有 **可復原性** 的，也就是:

$$(a \oplus b = c) \Rightarrow (c \oplus b = a) \wedge (c \oplus a = b)$$

於是，我們套用 **前綴和 *prefix sum*** 的思維，將它們一個一個 ***XOR*** 起來，形成一個新的序列 $< \text{prefix} >$

於是我們把 **從左界 l *XOR* 到右界 r 的值** 換成 $\text{prefix}*r* *oplus* *text{prefix}*{l - 1}，其中\text{prefix}*n* = *text{prefix}*{n - 1} \oplus a_n, \text{prefix}_0 = 0$

於是我們的總複雜度從 $O(n^3)$ 降到了 $O(n^2)$

時間複雜度 $O(n^2)$

subtask2

概念

但是 $O(n^2)$ 實在太慢了，因為 n 最大可以到 10^5

於是我們開始思考要如何讓它 **更快** 。

首先，我們思考對於任意數字 k ，最理想的狀態是 **前綴和序列中剛好有 k 的補數** ，也就是 $\overline{k}$ (**C++** 裡寫作 `~k`) 。

為甚麼呢?

因為 **補數** 的定義是把舊的數字中，每一個 ***bit*** 翻轉(`0` 換成 `1` ，反之亦然) ，而在 ***XOR*** 的世界中，一個 `0` 和 一個 `1` 會生出 `1` $(0 \oplus 1 = 1)$，這樣可以保證 $k \oplus \overline{k}$ 最大(因為每個 ***bit XOR*** 出來的結果都是 `1`)。

但事情總不會那麼順利， $\overline{k}$ 不會每次都剛好出現在序列中。

所以，我們需要 **取捨** 。

倘若有兩個數，前 l 個 ***bit*** 都不一樣，但是第 $l+1$ 相同，則我們將這兩個數的契合度稱為 l 。

而契合度越高的數字，所 ***XOR*** 運算出來的值也越大。
於是這個問題，就升級成了 **找契合度** 的單純問題。

舉個例子:

```
 6 -> 0110
11 -> 1011
 2 -> 0010
```

可以輕易看出與 11 契合度最高的數字為 6 ，因為它們的首位相同數字出現在第 3 個 ***bit*** ，較 11 和 2 右邊，因此可以推測出 11 和 6 的 ***XOR*** 結果較大。

因此，我們便開始實作。

實作

經過上面的思考，我們所要做的，就是不斷快速查詢 **前 n 個 *bit*** 的方法(其中 n 會連續)。

因此，我們想到了 ***TRIE*** ，這原本是一種用於 **字串搜尋** 的資料結構，旨在快速地找出和任意字串的最長共同前綴。

我們製造一棵 ***TRIE*** ，接著對數字們做遍歷，每次進入 ***TRIE*** 中尋找與這個數字契合度最高的數字，然後不斷更新最大值。

最後，我們確認一下空間、時間。

空間肯定夠，因為最多只有約 $\lceil log_2^{(10^{18})} \rceil \times 10^5 = 60 \times 10^5 = 6 \times 10^6$ 個 ***bit*** ，相當穩。

時間的話，$O(n \times \lceil log_2^{max(<a>)} \rceil)$

> 是說這題也能用一堆 `set` 肝出來。
> ~~但我太笨了沒想到~~
> 詳見 ***Testers' Code***

時間複雜度 $O(n \times log_2^{max(<a>)})$

H. ARC 人類表現估計 *H-ARC*

閱讀題，~~讀完就會寫了~~。

I. 家俱製造商 *Furniture*

subatsk1

唬爛用， $O(n^7)$ 都會過

subtask6

因為題目說:

> 不可能有無法開始的工作

但這裡又說

$m < n$

而聯通圖最少邊數為 $n-1$，即 $m < n \land n-1 \leq m$
也就是說 $m = n-1$

所以這張圖圖會是一棵樹，因此我們只要快樂地從原點走下去就行了ouob

subtask7

沒錯，仔細觀察可以發現這其實是一張可愛的 ***DAG*** ，也就是 **有向無環圖** ，而說到這個就得馬上想到 **拓樸排序** 。

於是我們使用 **拓樸排序** 將節點按照時間排好後，從頭到尾跑一次，每次將這個點的所有相鄰子孫的時間更新(往後挪)，最後就可以獲得答案ouob。

排序時間複雜度: $O(n \cdot log_2^n)$
更新時間複雜度: $O(n)$

J. 香蕉衝突 *Banana Wars*

subtask1

cout << "0\n";

subtask2

窮舉每一對線段，判定是否相交。
時間複雜度 $O(n^2)$

subtask3

把右邊的端點由上而下編號，左邊也是。
之後每條線段即可以一點對表示 (a, b)。
跟據 a 排序，之後看 b 的逆序數對個數及相交個數。

subtask4

將所有點逆時針編號，後每條線段即可以一點對表示 (a_1, b_1)。
考慮線段 $(a_1, b_1), (a_2, b_2)$，只要 $a_1 \le a_2$ 且 $a_2 \le b_1 \le b_2$ ，兩線段即相交。
只要跟據 a 排序，窮舉每條線 (a_2, b_2)，看排在他前面的線 (a_1, b_1)，其有多少個滿足 $a_2 \le b_1 \le b_2$ 。
這可以透過在值域上開 BIT 實現 (用類似找逆序數對的方法)。
時間複雜度 $O(n \log n)$

這題應該算是難觀察，但容易實作的題？