**Bài 1:**

Cho dãy số nguyên . Tìm một đoạn con có đúng k phần tử liên tiếp sao cho ước chung lớn nhất của các phần tử này là lớn nhất.

***Dữ liệu vào:***

* Dòng đầu:
* Dòng tiếp theo: .

***Kết quả:***

* Giá trị lớn nhất của ước chung lớn nhất của phần tử liên tiếp.

***Ví dụ:***

|  |  |
| --- | --- |
| **INPUT** | **OUTPUT** |
| **10 3**  **2 6 4 3 18 12 24 8 7 5** | **6** |

* Ban đầu tính của phần tử đầu tiên, gọi là 1 window
* Dịch window sang phải để tính phần tử tiếp theo, thực chất là bỏ 1 phần tử ở đầu rồi thêm 1 phần tử ở cuối 🡪 Window là 1 cái queue
* Nghĩa là cần các thao tác trên queue:
  + Thêm phần tử ở cuối
  + Bỏ phần tử ở đầu
  + Tính gcd của toàn bộ queue (khó)
* Cài đặt queue bằng 2 stack nằm ngược nhau (gọi là và )
  + Thêm phần tử vào cuối queue = Thêm vào stack
  + Bỏ phần tử ở đầu queue = Pop từ stack
    - Nếu đang rỗng thì pop từng phần tử ở rồi push vào
  + Gcd của queue =
    - Gcd của mỗi stack thì dễ tính
* Mỗi phần tử tối đa là cho vào stack 1 lần, chuyển từ stack sang 1 lần, tính gcd 2 lần 🡪

**Bài 2:**

Gọi là số lượng các cặp khác nhau sao cho và là nguyên tố. Hai cặp được gọi là khác nhau nếu 1 trong 2 số tham gia vào cặp này và không tham gia vào cặp còn lại. Cho số nguyên . Hãy tính giá trị biểu thức: .

**Dữ liệu vào**:

* Chứa số nguyên .

**Kết quả**:

* Gồm một số nguyên duy nhất là giá trị .

**Ví dụ:**

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| 9 | 12 |

Giải:

* Tạo sàn nguyên tố snt
* Duyệt từ i = 2 -> n để tìm g(i) => O(n)
* Duyệt p = 3 -> i để tìm p là snt, rồi suy ra q, nếu q là snt thì res++ => O(n)

=> O(n^2)

**Bài 3:**

Cho hai dãy số nguyên 𝐴 = và 𝐵 = () hãy tìm một phần tử trong dãy 𝐴 và một phần tử trong dãy 𝐵 có || là nhỏ nhất có thể (1 ≤ 𝑖 ≤ 𝑚; 1 ≤ 𝑗 ≤ 𝑛).

**Dữ liệu vào:**

* Dòng 1 chứa hai số nguyên dương 𝑚, 𝑛 ≤
* Dòng 2 chứa 𝑚 số nguyên (∀𝑖: || < )
* Dòng 3 chứa 𝑛 số nguyên (∀𝑗: | < )

**Kết quả:** Ghi ra hai chỉ số 𝑖 và 𝑗 của hai phần tử tương ứng tìm được.

**Ví dụ:**

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| **4 5**  **1 8 2 9**  **-5 -6 3 -7 -4** | **2 4** |

Giải thích: || = |8 + (−7)| = 1

* Tìm min . Chia 2 trường hợp
* 🡪 và . Cố định từng thì cần tìm lớn nhất mà 🡪 Sort mảng lại rồi tìm kiếm nhị phân
* : Tương tự
* Độ phức tạp:

**Bài 4:**

Cho số và một dãy gồm số . Định nghĩa cặp số nghịch nhau là cặp hai số và trong dãy thỏa điều kiện và .

**Yêu cầu:** Đếm số lượng các cặp số nghịch nhau trong dãy.

**Dữ liệu vào:**

* Dòng đầu tiên chứa một số nguyên
* Dòng thứ 2 chứa số nguyên .

**Kết quả:**

* Gồm một số nguyên duy nhất là số lượng các cặp số nghịch nhau trong dãy.

**Ví dụ:**

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| **6**  **1 2 4 3 5 1** | **5** |

* Sửa đổi thuật toán Merge Sort để đếm số cặp nghịch thế
* Hàm trả về số cặp nghịch thế nằm trong , đồng thời sắp xếp tăng dần
  + Chia đoạn thành và
  + Số cặp nghịch thế nằm trọn vẹn trong là
  + Số cặp nghịch thế nằm trọn vẹn trong là
  + Cần đếm số cặp nghịch thế mà nằm trong còn nằm trong
    - Lúc này và đã sắp xếp tăng
    - Với mỗi trong , cần đếm số lượng trong mà
    - Do tăng dần và tăng dần nên dùng 2 con trỏ để đếm trong
* Độ phức tạp:

**Bài 5:**

Cho xâu và xâu chỉ gồm các chữ cái thường. Xâu được gọi là xuất hiện tại vị trí i của xâu nếu:

**Yêu cầu:** Hãy tìm tất cả các vị trí mà xâu xuất hiện trong .

**Dữ liệu vào**

* Dòng 1: xâu .
* Dòng 2: xâu .

Độ dài không quá .

**Kết quả**

Ghi ra các vị trí tìm được trên 1 dòng (thứ tự tăng dần). Nếu không xuất hiện trong thì bỏ trắng.

**Ví dụ:**

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| **Aaaaa**  **aa** | **1 2 3 4** |

* Hàm Z

**Bài 6:**

Cho dãy gồm các số nguyên đã được sắp xếp sẵn theo thứ tự tăng dần trong mỗi danh sách. Các phần tử trong dãy có thể có cùng giá trị.

**Yêu cầu:** Hãy gộp tất cả dãy này lại thành một danh sách đã được sắp xếp theo thứ tự tăng dần.

**Dữ liệu vào**

* Dòng đầu tiên chứa số nguyên là số lượng các dãy số.
* dòng tiếp theo, mỗi dòng chứa một dãy các số nguyên được sắp xếp theo thứ tự tăng dần. Mỗi dòng chứa không quá 500 số nguyên.
* Tổng số lượng các phần tử trong dãy không vượt quá .

**Kết quả**

Gồm một dòng chứa dãy số đã được gộp lại theo thứ tự tăng dần.

**Ví dụ:**

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| **3**  **1 4 5**  **1 3 4**  **2 6** | **1 1 2 3 4 4 5 6** |

* Xem như mỗi lượt chọn 1 phần tử trong dãy đưa vào kết quả
* Vì mỗi dãy đã được sắp tăng 🡪 Phần tử luôn có thể chọn vào kết quả trước phần tử
* Ban đầu chỉ quan tâm phần tử đầu tiên của mỗi dãy, cái nào bé nhất cho vào kết quả và bỏ nó đi khỏi dãy
* Trở về bài toán ban đầu nhưng kích thước bé hơn
* Cần 1 tập hợp hỗ trợ:
  + Lưu phần tử (đầu của mỗi dãy)
  + Lấy min trong tập hợp
  + Xóa min trong ra khỏi tập hợp
  + Chèn số tiếp theo ngay sau số bị xóa vào tập hợp
* Priority queue
* Độ phức tạp: là số lượng phần tử trong dãy, mỗi lần thao tác với priority queue mất nên tổng cộng là

**Bài 7:**

Cho một dãy gồm số nguyên không âm là giá trị độ cao của một vùng đất. Một cơn mưa đủ nhiều đổ lên vùng đất này và nước còn đọng lại như hình bên dưới. Biết rằng nước có thể chảy về các vị trí thấp hơn kề cạnh nếu mực nước ở vị trí hiện tại lớn hơn vị trí kề cạnh. Hai biên của vùng đất là biển và có mực nước bằng 0.



**Yêu cầu:** Hãy tính tổng lượng nước còn đọng lại sau cơn mưa.

**Dữ liệu vào**

* Dòng đầu chứa số nguyên dương là chiều rộng của vùng đất. .
* Dòng thứ 2 chứa số nguyên không âm là độ cao tại vị trí trên vùng đất đó.

**Kết quả**

Gồm một số nguyên duy nhất là tổng lượng nước tại tất cả các vị trí sau cơn mưa.

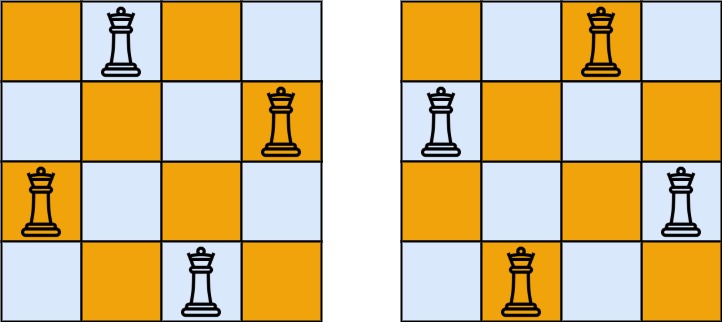
**Ví dụ:**

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| **4,2,0,3,2,5** | **9** |

* Với mỗi cột, tính lượng nước đọng lại trên cột đó, giả sử là cột
* Nếu bên phải toàn là các cột thấp hơn hoặc bằng 🡪 Khi nước rơi vào sẽ bị chảy hết sang phải
* Ngược lại nếu có 1 cột lớn hơn thì nước sẽ đọng với chiều cao của mực nước là chiều cao cột đó
* Nếu có nhiều cột thì nước đọng với chiều cao của cột lớn nhất
* Tương tự với bên trái
* Dễ thấy mực nước là min của 2 bên
* Với mỗi cột cần tìm và
* Mảng max tiền tố và hậu tố

**Bài 8:**

Cho bàn cờ kích thước . Hãy tìm tất cả các cách đặt quân hậu lên bàn cờ này sao cho không có 2 quân hậu nào có thể tấn công lẫn nhau. Biết rằng các quân hậu có thể di chuyển theo hàng dọc, hàng ngang và đường chéo. Bên dưới là 2 ví dụ cách đặt 4 quân hậu lên bàn cờ



**Dữ liệu vào**

* Gồm số là kích thước cạnh của bàn cờ và số lượng quân hậu.

**Kết quả**

* Dòng đầu tiên chứa số là số cách đặt các quân hậu khác nhau có thể có.
* Các dòng bên dưới chứa các ma trận đại diện cho bàn cờ kết quả. Các ma trận cách nhau bởi một dòng trống.
* Mỗi ma trận gồm dòng. Mỗi dòng chứa ký tự:
  + Ký tự ‘.’ biểu diễn tại vị trí ô trống
  + Ký tự ‘Q’ biểu diễn vị trí đó chứa quân hậu

**Ví dụ:**

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| **4** | **2**  **.Q..**  **...Q**  **Q...**  **..Q.**  **..Q.**  **Q...**  **...Q**  **.Q..** |

* Đánh số bàn cờ từ hàng , cột
* Dễ thấy mỗi con hậu phải đặt ở 1 cột khác nhau
* Vậy chỉ cần chọn hàng nào cho mỗi con hậu
* Khi đặt 1 con hậu, cần đánh dấu những ô nó kiểm soát (hàng, chéo chính, chéo phụ) để khi đặt con khác thì né ra 🡪 Mảng đánh dấu hàng, đường chéo chính, đường chéo phụ
* Các vị trí nằm cùng đường chéo chính thì có giống nhau
  + Lấy làm id cho mỗi đường chéo chính
  + Mảng không có chỉ số âm 🡪 Chuyển thành để có miền giá trị là
* Các vị trí nằm cùng đường chéo phụ thì có giống nhau
  + Lấy làm id cho mỗi đường chéo phụ
* Độ phức tạp: nhưng mỗi bước chỉ đặt vào những ô không bị kiểm soát nên chạy rất nhanh

**Bài 9:**

Cho 2 chuỗi và gồm các ký tự latin in thường ‘a’-‘z’. Trong chuỗi có thể có thêm các ký tự ‘\*’ và ‘.’ với ý nghĩa:

* Ký tự ‘\*’ biểu diễn rằng có thể lặp lại ký tự ngay trước đó một hoặc nhiều lần hoặc có thể không lặp lại lần nào. Ví dụ: chuỗi “abbbb” có thể được sinh ra từ chuỗi “ab\*” hoặc chuỗi “a” cũng có thể được sinh ra từ chuỗi “ab\*”.
* Ký tự ‘.’ biểu diễn rằng chỉ có thể thay thế cho một ký tự duy nhất tại vị trí đó trong chuỗi. Ví dụ: chuỗi “abc” có thể được sinh ra từ chuỗi “a.c” hoặc “ab.”.

**Yêu cầu:** Kiểm tra xem chuỗi có phải được sinh ra từ chuỗi hay không.

**Dữ liệu vào**

* Dòng đầu tiên chứa chuỗi . .
* Dòng thứ hai chứ chuỗi . .

**Kết quả**

* Trả về “true” nếu chuỗi có thể được sinh ra từ chuỗi , và “false” nếu ngược lại.

**Ví dụ:**

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| **mississippi**  **mis\*is\*p\*.** | **False** |
| **aab**  **c\*a\*b** | **True** |

* Nếu là \* thì hiển nhiên không thể nào sinh ra
* nếu từ có thể sinh ra
* Chuyển trạng thái bằng cách xét kí tự cuối cùng của và
  + là chữ cái 🡪 phải giống và phải là true
  + là chấm 🡪 là gì cũng được và phải là true
  + là \*
    - là chữ
      * lặp lại ít nhất 1 lần 🡪 và phải là true
      * không lặp lại nữa 🡪 phải là true
    - là dấu chấm
      * lặp lại ít nhất 1 lần 🡪 phải là true
      * không lặp lại nữa 🡪 phải là true
    - là \*
      * phải là true (2 dấu \* liên tiếp thì dấu \* bên phải không có tác dụng)
* Khởi tạo
  + ,
  + nếu là \* và
* Độ phức tạp:

**Bài 10:**

Cho một ma trận kích thước chứa các số 0 hoặc 1 tại mỗi ô.

**Yêu cầu:** Tìm hình vuông lớn nhất chứa toàn số 0 hoặc toàn số 1.

Ví dụ hình ảnh:

A picture containing text, water

Description automatically generated

**Dữ liệu vào**

* Dòng đầu tiên chứa 2 số nguyên dương và tương ứng là số lượng dòng và cột của ma trận. (
* dòng tiếp theo, mỗi dòng chứa số 0 hoặc 1 cách nhau bởi khoảng trắng.

**Kết quả**

* Diện tích của hình chữ nhật lớn nhất chứa toàn số 1 hoặc toàn số 1

**Ví dụ:**

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| **4 5**  **1 0 1 0 0**  **1 0 1 1 1**  **1 1 1 1 1**  **1 0 0 1 0** | **2** |

* Tìm hình vuông lớn nhất toàn số 0 (số 1 tương tự)
* Giả sử tìm được hình vuông có cạnh là 🡪 Luôn tìm được hình vuông có cạnh 🡪 Tìm kiếm nhị phân để có lớn nhất
* Với mỗi giá trị , cần kiểm tra tồn tại hình vuông có cạnh là hay không?
* Duyệt qua góc trái trên của hình vuông là 🡪 góc phải dưới là
* Nếu hình vuông đó có tổng là 0 🡪 Thỏa mãn
* Tính tổng hình vuông nhanh trong : Dùng tổng tiền tố là tổng từ ô
* Độ phức tạp:

**Bài 11:**

Xét cụm dữ liệu, mỗi cụm dữ liệu được chia thành 2 nhóm: nhóm loại 1 và nhóm loại 2. Cụm dữ liệu thứ có trọng số các nhóm tương ứng . Cần chọn ra nhóm loại 1 và nhóm loại 2 trong các cụm dữ liệu để làm mẫu thử với nguyên tắc mỗi cụm dữ liệu chỉ chọn 1 nhóm, do đó .

**Yêu cầu:** Tìm cách chọn các nhóm dữ liệu trong mỗi cụm sao cho tổng trọng số của tất cả các nhóm được chọn là lớn nhất.

**Dữ liệu vào:**

* Dòng đầu tiên:
* Dòng thứ trong dòng tiếp theo:

**Kết quả:**

* Gồm một số nguyên duy nhất là tổng trọng số lớn nhất.

|  |  |
| --- | --- |
| **INPUT** | **OUTPUT** |
| **4 2 1**  **4 9**  **3 5**  **7 2**  **5 5** | **21** |

**Ý tưởng:**

* Sắp xếp theo giảm dần, khi đó luôn chọn nhóm loại 1 trước xong mới chọn nhóm loại 2. Vì giả sử chọn nhóm loại 2 trước là và chọn nhóm loại 1 sau là với thì ta có nên , tức là đảo sang chọn nhóm 1 trước vẫn có kết quả tốt hơn.
* Sau đó quy hoạch động là giá trị lớn nhất khi xét vật đầu tiên, và đã chọn được vật. Chuyển trạng thái bằng cách
  + Bỏ qua 1 vật:
  + Chọn nó vào nhóm 1 (nếu nhóm 1 chưa chọn đủ vật):
  + Chọn nó vào nhóm 2 (nếu nhóm 1 đã chọn đủ vật):
* Độ phức tạp:

**Bài 12:**

Hãy chỉ ra cách xây dựng một hình vuông có diện tích là số nguyên dương cho trước với các tọa độ nguyên trong phạm vi từ đến .

Input: Số nguyên dương .

Output: Tọa độ 4 đỉnh của hình vuông được liệt kê cùng hoặc ngược chiều kim đồng hồ. Trường hợp không có lời giải thì thông báo Impossible.

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| **5** | **1 2**  **2 4**  **3 1**  **4 3** |
| **3** | **Impossible** |

# **Ý tưởng:**

* Giả sử và là 2 đỉnh có tọa độ nguyên tạo thành cạnh của hình vuông có diện tích
* Do là điểm nguyên nên và cũng phải nguyên
* Vậy nếu không biểu diễn được thành tổng của 2 bình phương thì không có kết quả
* Do hệ tọa độ lớn ) mà nhỏ (), nên giả sử khi tìm được 1 hình vuông, có thể dời nó về đỉnh
* Vậy giả sử là 1 đỉnh hình vuông
* Khi đó tìm bằng cách biểu diễn thành tổng 2 bình phương
* Tìm điểm sao cho và vuông góc
  + Vector vuông góc vector
  + Suy ra tọa độ của
* Tìm đỉnh còn lại

**Bài 13:**

Trong mặt phẳng cho điểm có tọa độ nguyên sao cho không có 2 điểm nào trùng nhau và không có 3 điểm nào thẳng hàng ( là một số chẵn).

**Yêu cầu:** Hãy chọn ra 2 điểm trong số điểm để đường thẳng qua 2 điểm này chia tập điểm thành 2 phần có số lượng điểm bằng nhau.

**Dữ liệu vào:**

* Dòng đầu tiên:
* Dòng thứ trong dòng tiếp theo: – tọa độ của điểm thứ .

**Kết quả:**

* Gồm 2 số là số thứ tự của 2 điểm được chọn.

**Ví dụ:**

|  |  |
| --- | --- |
| **INPUT** | **OUTPUT** |
| **6**  **3 5**  **1 3**  **3 1**  **6 1**  **8 3**  **6 5** | **2 5** |

**Ý tưởng:**

* Giả sử cố định 1 điểm là điểm trái nhất
* Khi đó quét đường chia mặt phẳng thành 2 phần, ngược chiều kim đồng hồ thì sẽ có hệ số góc tăng dần
* Vậy sắp xếp điểm còn lại theo hệ số góc với điểm trái nhất tăng dần. Lấy điểm ở giữa nối với điểm trái nhất sẽ chia thành 2 phần có số điểm bằng nhau
* Chú ý trường hợp có 2 điểm trái nhất 🡪 hệ số góc đường thẳng nối 2 điểm này không xác định

Chart, line chart

Description automatically generated

**Bài 14:**

Cho đa giác lồi đỉnh có tọa độ nguyên . Các đỉnh được liệt kê ngược chiều kim đồng hồ và không có 3 đỉnh nào thẳng hàng. Chỉ ra một cách kẻ 1 đường chéo đi qua 2 đỉnh và chia đa giác thành 2 phần có diện tích chênh lệch nhau ít nhất.

**Dữ liệu vào**

* Dòng đầu tiên:
* Dòng thứ trong dòng sau: .

**Kết quả:**

* Gồm 2 số nguyên là số thứ tự 2 đỉnh được chọn.

**Ví dụ:**

|  |  |
| --- | --- |
| **INPUT** | **OUTPUT** |
| **4**  **0 2**  **0 0**  **2 0**  **3 3** | **2 4** |

* Duyệt qua tất cả cách nối đường chéo rồi tính chênh lệch bé nhất, gọi là diện tích đa giác và là 1 phần đa giác khi bị cắt, thì cần tìm min
* Cố định 1 đỉnh, duyệt đỉnh còn lại của đường chéo theo ngược chiều kim đồng hồ, khi đó phần diện tích bị cắt sẽ tăng dần bằng cách lấy diện tích đường chéo trước đó cộng thêm diện tích hình tam giác
* Độ phức tạp:

**Bài 15:**

Trên mặt phẳng cho 2 hình tròn có cùng bán kính và tọa độ tâm tương ứng và . Tính diện tích phần hợp của 2 hình tròn trên mặt phẳng. Tọa độ tâm và bán kính hình tròn đều là các số nguyên.

**Dữ liệu vào:**

* Gồm các số nguyên

**Kết quả:**

* Chứa một số thực duy nhất là diện tích phần hợp lấy chính xác 3 chữ số thập phân.

**Ví dụ:**

|  |  |
| --- | --- |
| **INPUT** | **OUTPUT** |
| **1 2 3 4 2** | **22.850** |

* Trường hợp 2 đường tròn không giao nhau (so sánh khoảng cách giữa 2 tâm với tổng của 2 bán kính)
* Trường hợp đường tròn này nằm trong đường tròn kia (do cùng bán kính nên tâm phải bằng nhau)
* Trường hợp 2 đường tròn giao nhau thì tính phần giao bằng diện tích hình quạt tròn trừ đi diện tích hình tam giác

**Bài 16:**

Cho một cây đỉnh. Khoảng cách giữa 2 đỉnh là số cạnh trên đường đi từ đến . Độ rộng của cây là tổng khoảng cách giữa tất cả cặp đỉnh . Hãy xác định độ rộng của cây.

**Dữ liệu vào**

* Dòng đầu tiên: .
* dòng tiếp theo: – một cạnh của cây.

**Kết quả:**

* Gồm một số nguyên duy nhất là độ rộng của cây.

**Ví dụ:**

|  |  |  |
| --- | --- | --- |
| **TREE.INP** | **TREE.OUT** |  |
| **5**  **1 2**  **1 3**  **3 4**  **3 5** | **18** |

* Gọi là khoảng cách từ đỉnh đến đỉnh
* Đề bảo tính tổng các mà
* Tính tổng các , không cần quan tâm tới , xong rồi chia 2
* Giả sử gốc cây tại đỉnh 1
* Với mỗi đỉnh , cần tính khoảng cách đến đỉnh thuộc 2 loại:
  + Nằm trong cây gốc
  + Nằm ngoài cây gốc
* là tổng khoảng cách từ đến các đỉnh nằm ngoài cây gốc
* là tổng khoảng cách từ đến các đỉnh nằm trong cây gốc
* là số lượng đỉnh trong cây gốc , tính cả
* Tính
  + Gọi là 1 con trực tiếp của
  + Khi đó với mỗi đường đi xuất phát từ xuống con của , có thể tạo thành đường đi xuất phát từ xuống con của bằng cách nối cạnh
  + Do đó tổng với là con
* Tính
  + Đường đi từ ra 1 nút anh em của , tức là với là nút con của
    - Gọi là cha
  + Đường đi từ ra 1 nút bên ngoài
* Độ phức tạp:

**Bài 17:**

Cho số nguyên dương . Hãy tìm số nguyên dương nhỏ nhất có không quá 9 chữ số thỏa chỉ chứa các chữ số {0,1} và là bội của .

**Dữ liệu vào:**

* Gồm một số nguyên dương

**Kết quả:**

* Gồm một số nguyên duy nhất là

**Ví dụ:**

|  |  |
| --- | --- |
| **INPUT** | **OUTPUT** |
| **6** | **1110** |

* Số cần tìm có tối đa 9 chữ số 🡪 Coi như nó có 9 chữ số và cho phép có các chữ số 0 ở đầu
* Vậy có trường hợp, mỗi trường hợp là 1 dãy nhị phân
* Quay lui để liệt kê các trường hợp
  + Điền các số từ trái sang phải, điền số 0 rồi số 1 để đảm bảo số tìm dc là nhỏ nhất
* Độ phức tạp:

**Bài 18:**

Cho dãy số là một hoán vị của các số từ đến . Có thể thực hiện thao tác sau trên dãy số: chọn vị trí và đảo ngược thứ tự các phần tử , các phần tử còn lại giữ nguyên.

**Yêu cầu:** Tìm số thao tác thực hiện ít nhất để dãy có thứ tự tăng dần.

**Dữ liệu vào:**

* Dòng đầu tiên:
* Dòng thứ hai:

**Kết quả:**

* Gồm một số duy nhất là số thao tác ít nhất.

**Ví dụ:**

|  |  |  |
| --- | --- | --- |
| INPUT | OUTPUT | Giải thích |
| 5  5 2 3 4 1 | 4 | 5 2 3 4 1  4 3 2 5 1  2 3 4 5 1  5 4 3 2 1  1 2 3 4 5 |

* Coi mỗi hoán vị là 1 đỉnh, mỗi phép biến đổi từ hoán vị này sang hoán vị kia là 1 cạnh nối 2 đỉnh với trọng số là 1
* Tìm đường đi ngắn nhất từ đỉnh tới đỉnh trong đồ thị trọng số là 1
* BFS trên đồ thị có Đỉnh, Cạnh
* Làm sao đánh dấu các đỉnh đã thăm khi mỗi đỉnh là 1 mảng chứ không phải 1 số
  + Chuyển hoán vị về số 🡪 Chuyển về thứ tự từ điển của nó (bài 20)

**Bài 19:**

Cho đoạn liên tiếp, đoạn thứ có chiều cao. Chiều cao chênh lệch giữa 2 đoạn liên tiếp không vượt quá 1 đơn vị. Ta cần đào sâu xuống càng sâu càng tốt. Ta dự định chi đồng cho công việc này. Chi phí giảm chiều cao của một đoạn đi 1 đơn vị là 1 đồng.

***Yêu cầu:*** Cho biết các chiều cao và chi phí . Hãy xác định chiều cao thấp nhất (đào sâu nhất) mà ta có thể đào được với đồng mà vẫn đảm bảo chênh lệch giữa 2 đoạn liên tiếp không quá 1 đơn vị.

* Dòng đầu chứa 2 số nguyên dương
* Dòng thứ 2 chứa số nguyên không âm

***Kết quả***

Gồm một số nguyên duy nhất là độ cao thấp nhất mà công ty có thể đào được.

***Ví dụ:***

|  |  |  |
| --- | --- | --- |
| INPUT | OUTPUT | Giải thích |
| **4 3**  **1 1 1 1** | **-1** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **1** |  |  |  |  | | **0** |  |  |  |  | | **-1** |  |  |  |  | |
| **4 3**  **1 2 2 1** | **0** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **2** |  |  |  |  | | **1** |  |  |  |  | | **0** |  |  |  |  | |

* Giả sử biến đổi được về độ sâu thì cũng biến đổi được về độ sâu (vì tốn ít tiền hơn)
* Chặt nhị phân độ sâu nhỏ nhất có thể biến đổi
* Miền chặt: min\_height – T 🡪 min\_height
* Kiểm tra giá trị
  + Giả sử biến đổi về , khi đó phải biến đổi các cột bên trái và phải cho chênh lệch 2 cột liên tiếp k quá 1
  + Xét bên phải, bên trái tương tự
  + Nếu thì không cần phải biến đổi bên phải
    - Ngược lại cần biến đổi về
    - Lập luận tương tự, gọi là cột bên phải gần nhất mà thì chỉ cần biến đổi từ cột đến
    - Chi phí biến đổi là
  + Với mỗi vị trí , gọi là cột bên phải gần nhất mà thì luôn có 🡪 Chạy 2 con trỏ để tìm cho mỗi trong
  + Chi phí biến đổi có đoạn thì dùng tổng dồn, đoạn thì dùng toán
* Độ phức tạp:

**Bài 20:**

Xét tất cả các hoán vị của dãy số tự nhiên (1,2, … , 𝑛). Giả sử rằng các hoán vị được sắp xếp theo thứ tự từ điển và đánh số từ 1 tới 𝑛!

Ví dụ với 𝑛 = 3, có 6 hoán vị: (1,2,3); (1,3,2); (2,1,3); (2,3,1); (3,1,2); (3,2,1)

**Yêu cầu:** Cho trước một hoán vị () hãy cho biết số thứ tự 𝑥 của hoán vị đó và ngược lại: Cho trước một số thứ tự 𝑦 (1 ≤ 𝑦 ≤ 𝑛!) hãy tìm dãy hoán vị () mang số thứ tự 𝑦.

**Dữ liệu vào:**

* Dòng 1: Chứa 𝑛 số (𝑛 ≤ 20)
* Dòng 2: Chứa số 𝑦

**Kết quả:**

* Dòng 1: Ghi số 𝑥
* Dòng 2: Ghi 𝑛 số

**Ví dụ:**

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| **2 1 3**  **4** | **3**  **2 3 1** |

* Có thể biểu diễn hoán vị thành cây với độ sâu là , ví dụ
* Text

  Description automatically generated
* 1 đường đi từ gốc đến lá tương ứng 1 hoán vị
* Từ số thứ tự ra hoán vị hay hoán vị ra số thứ tự đều là di chuyển trên cây này