



Ngôn ngữ lập trình C++

BÀI TẬP ÔN THI HSG TỈNH 2021



1☀. Đường chéo - Cross.Cpp

Cho bảng số gồm m dòng và n cột. Các dòng được đánh chỉ số từ 1 đến m (trên xuống dưới) và các cột được đánh chỉ số từ 1 đến n (trái sang phải). Ô nằm trên dòng i cột j được gọi là ô (i, j) .

Yêu cầu: Cho ô $(i_0; j_0)$, tính tổng các số thuộc 2 đường chéo đi qua ô $(i_0; j_0)$.

Dữ liệu cho trong file Cross.Inp gồm:

- Dòng đầu ghi 4 số nguyên dương m, n, i_0 và j_0 ($m, n \leq 100, 1 \leq i_0 \leq m, 1 \leq j_0 \leq n$).
- m dòng sau, mỗi dòng ghi n số nguyên, các số thuộc $[-1000; 10000]$.

Kết quả ghi ra file Cross.Out là tổng các số trên đường chéo chứa ô $(i_0; j_0)$.

Ví dụ:

Cross.Inp	Cross.Out
3 5 2 4 1 2 3 4 5 2 3 3 3 3 1 1 1 1 1	13



2☀. Đường chéo (2) - Cross2.Cpp

Cho bảng số gồm m dòng và n cột. Các dòng được đánh chỉ số từ 1 đến m (trên xuống dưới) và các cột được đánh chỉ số từ 1 đến n (trái sang phải). Ô nằm trên dòng i cột j được gọi là ô (i, j) .

Yêu cầu: Cho 2 ô $(i_0; j_0)$ và $(i_1; j_1)$ tính tổng các số thuộc các đường chéo đi qua ô $(i_0; j_0)$ hoặc qua ô $(i_1; j_1)$

Dữ liệu cho trong file Cross2.Inp gồm:

- Dòng đầu ghi 6 số nguyên dương m, n, i_0, j_0, i_1, j_1 ($m, n \leq 100, 1 \leq i_0, i_1 \leq m, 1 \leq j_0, j_1 \leq n$).
- m dòng sau, mỗi dòng ghi n số nguyên, các số thuộc $[-1000; 10000]$.

Kết quả ghi ra file Cross2.Out là tổng các số trên đường chéo chứa ô $(i_0; j_0)$ hoặc chứa ô $(i_1; j_1)$.

Nếu một số trên ô chứa thuộc nhiều đường chéo thì chỉ được tính 1 lần.

Ví dụ:



Cross2.Inp	Cross2.Out
3 5 2 4 2 1	18
1 2 3 4 5	
2 3 3 3 3	
1 1 1 1 1	

**3. Chú ếch và những lá sen**

Chú ếch Brica sống trong hồ sen rất đẹp. Có m lá sen được xếp theo một hàng ngang, các lá sen được đánh số thứ tự $1, 2, \dots, m$ (từ trái sang phải). Lá sen thứ i có vị trí x_i ($0 = x_1 < x_2 < \dots < x_m \leq 10^9$). Chú ếch Brica bắt đầu tại vị trí lá sen 1, và lần lượt nhảy tới các lá sen khác để đến lá sen m . Mỗi khi nhảy từ lá sen i đến lá sen j ($i < j$) thì độ dài của bước nhảy đó bằng $x_j - x_i$. Độ *uyển chuyển và nhẹ nhàng* của ếch được định nghĩa là độ dài của bước nhảy dài nhất của chú ếch đó thực hiện.

Yêu cầu: Tính độ *uyển chuyển và nhẹ nhàng* nhỏ nhất của chú ếch Brica để có thể nhảy từ lá sen 1 đến lá sen m mà không quá k lần nhảy. Tức là tìm số d nhỏ nhất sao cho độ dài của mỗi bước nhảy của ếch không quá d và số lần nhảy để đến lá sen m không quá k .

Dữ liệu cho trong file **Jumfrog.inp** gồm:

- Dòng đầu ghi hai số nguyên m và k tương ứng là số lá sen và số bước nhảy tối đa ($k < m$).
- Dòng sau ghi m số nguyên x_1, x_2, \dots, x_m ($0 = x_1 < x_2 < \dots < x_m \leq 10^9$).

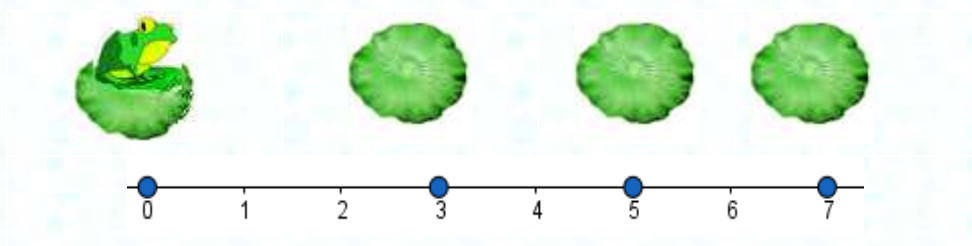
Kết quả ghi ra file **Jumfrog.out** là độ *uyển chuyển và nhẹ nhàng* nhỏ nhất tìm được.

Ví dụ:

Jumfrog.inp	Jumfrog.out
4 3	3
0 3 5 7	

Giới hạn:

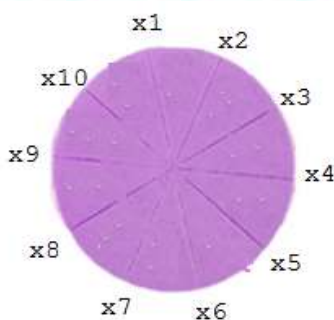
- Sub1: $m \leq 1000$;
- Sub2: $m \leq 10^5$.





4☀. Chạy vòng tròn nhặt số

Bé Tôm có N đồng xu hình tròn, trên mỗi đồng xu có ghi 1 số trong N số tự nhiên $1, 2, 3, \dots, N$. Không có hai đồng xu nào có số ghi trên đó giống nhau. Hiện tại, N đồng xu này được đặt tại N vị trí trên một đường tròn có N vạch chia (hình vẽ với $N = 10$). Các vạch được đánh số từ 1 đến N theo chiều kim đồng hồ. Đồng xu đặt tại vạch thứ i có số ghi trên đó là x_i (với $i = 1, 2, \dots, N$).



Bé Tôm sẽ xuất phát từ vị trí 1 (có ghi số x_1). Tôm sẽ chạy theo chiều kim đồng hồ và nhặt các đồng xu có ghi các số lần lượt là $1, 2, 3, 4, \dots, N$ để bỏ vào túi. Tôm sẽ dừng lại khi nhặt được N đồng xu và dừng tại vị trí 1.


Yêu cầu: Tính xem Tôm phải chạy bao nhiêu vòng tròn để nhặt được N đồng xu.

Dữ liệu cho trong file **CollecNumber.Inp** gồm:

- Dòng đầu ghi số nguyên dương N là số đồng xu cũng như là số vạch trên đường tròn.
- Dòng sau ghi N số nguyên dương x_1, x_2, \dots, x_N là các số ghi trên N đồng xu. Chú ý là x_1, x_2, \dots, x_N là một hoán vị của N số nguyên $1, 2, 3, \dots, N$.

Kết quả ghi ra file **CollecNumber.Out** là số vòng mà Tôm cần chạy để nhặt hết N đồng xu.

Ví dụ:

CollecNumber.Inp	CollecNumber.Out	Giải thích
5 1 3 2 5 4	3	 Vòng 1, nhặt được 2 đồng xu có số 1 và 2, Vòng 2, nhặt được 2 đồng xu có số 3 và 4. Vòng 3, nhặt được đồng xu có số 5.

Giới hạn:

- Sub 1: $N \leq 1000$;
- Sub 2: $N \leq 500.000$.

**5☀ Chia dãy**

Cho dãy số nguyên a_1, a_2, \dots, a_n . Ta chia dãy số thành k đoạn liên tiếp: a_1, \dots, a_{i_1} ; $a_{i_1+1}, \dots, a_{i_2}$; ..., $a_{i_{k-1}+1}, \dots, a_{i_k}$. Với $1 \leq i_1 < i_2 < \dots < i_k = n$. Ta gọi đại diện lớn nhất của một đoạn là số hạng lớn nhất trong các số hạng thuộc đoạn. Tức là số đại diện lớn nhất của dãy a_1, \dots, a_{i_1} là $\max(a_1, \dots, a_{i_1})$; đại diện lớn nhất của dãy $a_{i_{k-1}+1}, \dots, a_{i_k}$ là $\max(a_{i_{k-1}+1}, \dots, a_{i_k})$.

Yêu cầu: Tìm cách chia dãy số thành k đoạn để tổng đại diện lớn nhất của k đoạn đó là nhỏ nhất.

Dữ liệu cho trong file SPLITSEQ.INP gồm:

- Dòng đầu ghi hai số nguyên n và k ($1 \leq k \leq \min(100, n)$; $1 \leq n \leq 1000$).
- Dòng sau ghi n số nguyên a_1, a_2, \dots, a_n ($|a_i| \leq 10^9$).

Kết quả ghi ra file SPLITSEQ.OUT là giá trị nhỏ nhất của tổng k đại diện lớn nhất của k dãy con.

Ví dụ:

SPLITSEQ.INP	SPLITSEQ.OUT
4 2 1 3 2 4	5

Giải thích: Chia làm hai dãy: dãy 1; có đại diện lớn nhất bằng 1; dãy 3; 2; 4 có đại diện lớn nhất bằng 4.



6☀. Nén dãy số

Cho số nguyên dương n , ta có dãy số A gồm các số nguyên từ 1 đến n . Phép nén dãy số là tạo ra dãy số mới mà các phần tử được tạo ra bằng cách lần lượt cộng 2 số cạnh nhau của dãy số ban đầu. Mỗi lần nén dãy số, dãy số mới sẽ ít hơn dãy trước 1 phần tử. Ta nén dãy số đến khi chỉ còn 1 phần tử, phần tử đó là giá trị nén dãy số.

Yêu cầu: Cho n , tìm giá trị nén dãy số. Vì giá trị có thể rất lớn, nên chỉ cần in ra số dư của phép chia giá trị nén dãy số cho 10^9 .

Dữ liệu: cho trong file NENDAY.INP gồm một số nguyên dương n .

Kết quả: ghi ra file NENDAY.OUT là kết quả tìm được.

Ví dụ:

NENDAY.INP	NENDAY.OUT	Hình vẽ minh họa
4	20	

Giới hạn:

- Sub task 1: $n \leq 5000$;
- Sub task 2: $n \leq 2018201820182018$;