

Đề chính thức

Thời gian làm bài: 120 phút

Hướng dẫn:

- Thí sinh tạo một thư mục có tên là số báo danh của thí sinh trên đĩa **D:** (Nếu không có đĩa **D:** thì cán bộ coi thi hướng dẫn tạo thư mục trên đĩa khác!). Lưu tất cả bài làm vào thư mục này.
- Sau khi hết giờ làm bài, cán bộ coi thi hướng dẫn thí sinh đóng các tập tin bài làm, thoát chương trình soạn thảo, nén thư mục và chép bài (thư mục bài làm đã được nén) theo hướng dẫn của BTC.

NỘI DUNG ĐỀ THI

Bài 1: Số lần xuất hiện nhiều hơn k (đặt tên file bài làm là **Frequency)**

Viết chương trình tìm số X có số lần xuất hiện nhiều hơn k lần trong dãy N số đã cho như sau:

- Đọc dữ liệu từ bàn phím hoặc từ file **Frequency.inp**:
 - + Dòng đầu tiên là số nguyên N ($1 \leq N \leq 4 \times 10^9$);
 - + Dòng thứ hai gồm N số nguyên a_1, a_2, \dots, a_N , mỗi số cách nhau bởi 1 hoặc nhiều khoảng trắng;
 - + Dòng cuối cùng là số nguyên k ($1 < k \leq 1000$).
- Tìm số nguyên X xuất hiện nhiều hơn k lần và xuất kết quả ra màn hình hoặc file **Frequency.out**:
 - + Mỗi dòng gồm 2 số X: y (với X là số nguyên trong dãy số và y là số lần xuất hiện của X).

Ví dụ dữ liệu nhập vào	Ví dụ kết quả
10	2:3
1 2 3 4 4 5 6 1 2 4	4:3
2	

Hình 1. Ví dụ dữ liệu nhập vào và kết quả

Bài 2: Khu vực có mật độ dân cư cao nhất (đặt tên file bài làm là **Density)**

Người ta chia bản đồ dân cư của tỉnh A thành một lưới gồm có $N \times M$ ô, giá trị của mỗi ô là mật độ dân cư của ô đó. Viết chương trình tìm các ô có mật độ dân cư lớn nhất so với các ô lân cận của nó.

- Đọc dữ liệu từ bàn phím hoặc file **Density.inp**:
 - + Dòng đầu tiên: 2 số nguyên N, M là số dòng và số cột (với $1 \leq N, M \leq 10^9$);
 - + N dòng tiếp theo: Mỗi dòng có M giá trị là mật độ dân cư của M ô.
- Tìm tọa độ của ô có mật độ cao nhất so với các ô lân cận của nó và xuất kết quả ra màn hình hoặc file **Density.out** (Biết ô (x,y) có tối đa 8 ô lân cận là $(x-1, y-1)$, $(x-1, y)$, $(x-1, y+1)$, $(x, y-1)$, $(x, y+1)$, $(x+1, y-1)$, $(x+1, y)$, $(x+1, y+1)$; chỉ số dòng từ 1 đến N, chỉ số cột từ 1 đến M):
 - + Mỗi dòng là tọa độ của ô có mật độ cao nhất so với các ô lân cận của nó.

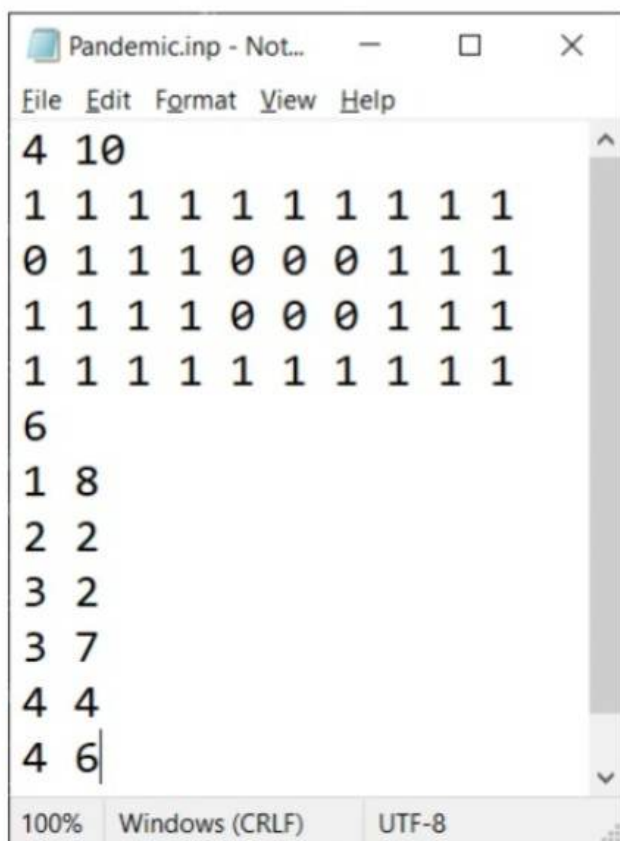
Ví dụ dữ liệu nhập vào	Ví dụ kết quả
4 6	1 6
1 2 3 4 2 8	3 3
2 5 4 3 3 2	
4 6 9 6 4 3	
1 3 7 5 3 3	

Hình 2. Ví dụ dữ liệu nhập vào và kết quả

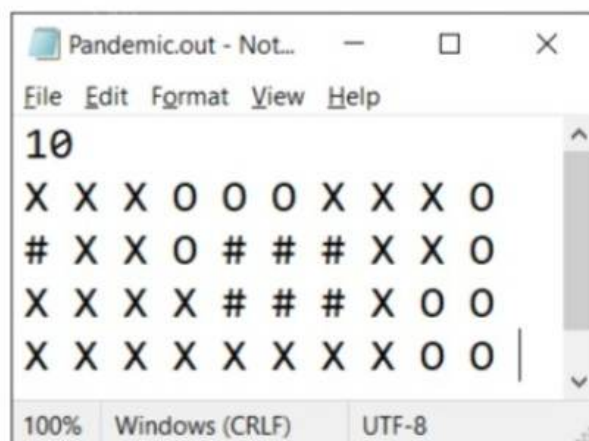
Bài 3: Xử lý dịch bệnh (đặt tên file bài làm là **Pandemic**)

Người ta chia bản đồ khu vực cần kiểm soát dịch bệnh thành 1 lưới gồm có $N \times M$ ô. Giá trị của ô là 0 (nếu chưa xảy ra dịch bệnh, không cần xử lý) hoặc 1 (đã xảy ra dịch bệnh, cần xử lý). Mỗi đội xử lý dịch bệnh được đặt tại 1 ô và có khả năng xử lý cho 9 ô bao gồm ô đặt đội xử lý và 8 ô bao xung quanh ô đặt đội xử lý. Viết chương trình tìm các ô cần được xử lý nhưng nằm ngoài phạm vi xử lý của các đội như sau:

- Đọc dữ liệu từ bàn phím hoặc file **Pandemic.inp**:
 - + Dòng đầu tiên: 2 số nguyên N, M là số dòng và số cột (với $1 \leq N, M \leq 10^9$);
 - + N dòng tiếp theo: Mỗi dòng có M giá trị 0 hoặc 1 là giá trị của các ô;
 - + Dòng tiếp theo là số S : Số đội xử lý đã được bố trí ($1 \leq S \leq 1000$);
 - + S dòng cuối: Mỗi dòng là tọa độ x, y (với $1 \leq x \leq N, 1 \leq y \leq M$) của đội xử lý.
- Tìm và xuất kết quả ra màn hình hoặc file **Pandemic.out**:
 - + Dòng đầu tiên: Tổng số ô cần xử lý nhưng nằm ngoài phạm vi xử lý của các đội;
 - + N dòng tiếp theo: Bản đồ gồm $N \times M$ ô được gán nhãn như sau: ký tự # đối với ô không cần xử lý, ký tự X đối với ô cần xử lý và thuộc phạm vi xử lý của các đội, ký tự O đối với ô cần xử lý nhưng nằm ngoài phạm vi xử lý của các đội.



```
Pandemic.inp - Not...
File Edit Format View Help
4 10
1 1 1 1 1 1 1 1 1 1
0 1 1 1 0 0 0 1 1 1
1 1 1 1 0 0 0 1 1 1
1 1 1 1 1 1 1 1 1 1
6
1 8
2 2
3 2
3 7
4 4
4 6
```



```
Pandemic.out - Not...
File Edit Format View Help
10
X X X O O O X X X O
# X X O # # # X X O
X X X X # # # X O O
X X X X X X X X O O
100% Windows (CRLF) UTF-8
```

Hình 3. Ví dụ dữ liệu nhập trong file **Pandemic.inp** và kết quả trong file **Pandemic.out**

Bài 4: Viết lại họ và tên (đặt tên file bài làm là **WriteName**)

Cách viết họ và tên theo kiểu của người Việt là: họ, chữ lót/tên đệm (nếu có), tên. Ví dụ: Hồ Xuân Hương, Lê Lợi. Cách viết họ và tên theo kiểu của nước A là: tên, họ (hoặc họ và chữ lót nối với nhau bởi dấu -). Ví dụ: Huong, Ho-Xuan; Loi, Le. *Ký tự đầu tiên của mỗi từ là ký tự in hoa, các ký tự còn lại của mỗi từ là ký tự in thường; tên và họ (hoặc họ và chữ lót nối với nhau bởi dấu -) được viết cách nhau bởi 1 dấu phẩy và 1 khoảng trắng.* Viết chương trình viết lại họ và tên như sau:

- Đọc dữ liệu từ file **WriteName.inp** như sau:
 - + Mỗi dòng là họ và tên của một người được viết theo kiểu của người Việt (có thể chưa được chuẩn hóa do dư khoảng trắng hoặc sử dụng ký tự in hoa, in thường chưa đúng).
- Viết lại họ và tên theo cách viết của nước A và lưu vào file **WriteName.out** như sau:
 - + Mỗi dòng là họ và tên của một người.



Hình 4. Ví dụ dữ liệu nhập trong file WriteName.inp và kết quả trong file WriteName.out

Bài 5: Mã hóa Caesar (đặt tên file bài làm là Caesar)

- Gọi Plaintext= $x_1x_2...x_m$ là nội dung văn bản gốc (với x_i là ký tự trong bảng chữ cái A).
- Gọi Ciphertext= $y_1y_2...y_m$ là kết quả mã hóa (với y_i là ký tự trong bảng chữ cái A).
- Bảng chữ cái A gồm có 27 ký tự được sắp xếp như hình bên dưới.

Vị trí	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
Ký tự	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	'

- Gọi:

k là khóa;

P là vị trí của ký tự x_i trong bảng chữ cái A;

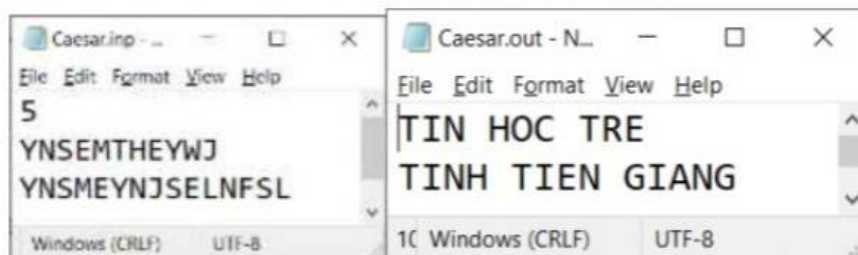
C là vị trí của ký tự y_j trong bảng chữ cái A;

N là số ký tự trong bảng chữ cái A, $N=27$.

- Muốn giải mã Ciphertext= $y_1y_2...y_m$ thành Plaintext= $x_1x_2...x_m$ thì người ta giải mã từng ký tự y_i thành ký tự x_i theo công thức $P=(N+C-k) \bmod N$. ($A \bmod B$ là chia A cho B lấy phần dư).

Viết chương trình giải mã như sau:

- Đọc dữ liệu từ file **Caesar.inp**:
 - + Dòng thứ nhất: Số nguyên k là khóa (với $1 < k < 26$);
 - + Các dòng tiếp theo, mỗi dòng là một nội dung Ciphertext cần được giải mã.
- Giải mã và ghi kết quả vào file **Caesar.out**:
 - + Mỗi dòng là một kết quả Plaintext tương ứng với Ciphertext đã cho.

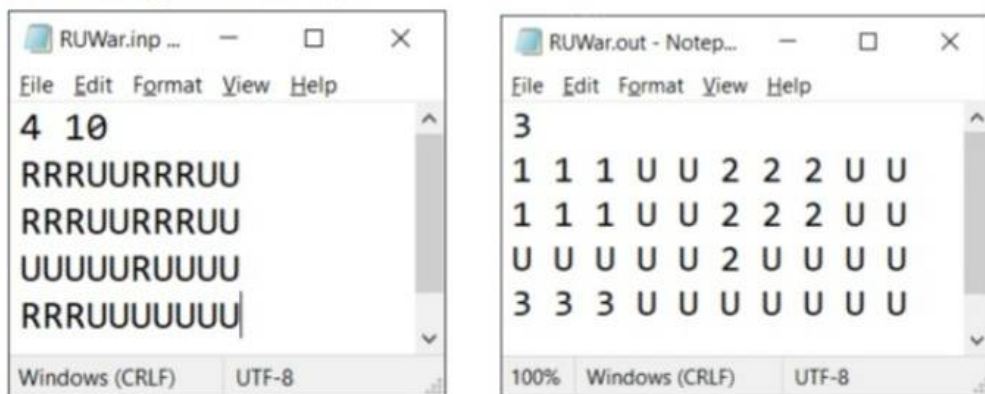


Hình 5. Ví dụ dữ liệu trong file Caesar.inp và kết quả trong file Caesar.out

Bài 6: Chiến tranh giữa R và U (đặt tên file bài làm là RUWar)

Người ta chia bản đồ khu vực giao tranh giữa đội quân R và đội quân U thành 1 ma trận có $N \times M$ ô ($100 \leq N, M \leq 700$). Mỗi ô được gán nhãn là R (nếu quân R đang chiếm đóng), U (nếu quân U đang chiếm đóng). Tìm và đánh số các khu vực được quân đội R chiếm đóng (mỗi khu vực được quân đội R chiếm đóng là một số ô được gán nhãn R liền kề nhau hay mỗi khu vực là một thành phần liên thông) như sau:

- Đọc dữ liệu từ file **RUWar.inp**:
 - + Dòng thứ nhất: 2 số nguyên N, M;
 - + N dòng tiếp theo là $N \times M$ ô được gán nhãn R hoặc U.
- Tìm và đánh số các khu vực được quân đội R chiếm đóng và ghi kết quả vào file **RUWar.out**:
 - + Dòng thứ nhất: số khu vực được quân đội R chiếm đóng;
 - + N dòng tiếp theo là bản đồ có $N \times M$ ô được gán nhãn cho mỗi ô theo quy tắc: ký tự U là ô do quân đội U chiếm đóng; các số 1, 2, 3, ... là số thứ tự của các khu vực do quân đội R chiếm đóng (tất cả ô thuộc khu vực thứ nhất được gán số 1; tất cả ô thuộc khu vực thứ hai được gán số 2, tất cả ô thuộc khu vực thứ 3 được gán số 3, v.v.).



Hình 6. Ví dụ dữ liệu nhập RUWar.inp và kết quả RUWar.out

Lưu ý:

- Cán bộ coi thi không giải thích đề thi.
- Thí sinh không được sử dụng tài liệu.

----- HẾT -----