Alcochete 2021

Quando pensávamos que o novo aeroporto de Lisboa ia para o Montijo, eis que, após complicações inesperadas, parece que talvez não vá. Assim, a hipótese de Alcochete volta para cima da mesa. Não sei se será desta, mas ainda que se faça um aeroporto no Montijo, não durará mais do que uns 30 anos, dizem os peritos, porque lá por 2050 estará inundado, por causa da subida no nível médio dos oceanos (culpa das alterações climáticas!) Nessa altura, será preciso um verdadeiro novo aeroporto, a sério, e então será em Alcochete, a fazer fé nos estudos já realizados.

Seja agora, seja daqui a 30 anos, quando for construído o aeroporto em Alcochete, ou, melhor, na zona do atual campo de tiro (e não sobre a simpática vila de Alcochete, bem entendido), será preciso deitar abaixo alguns dos muitos sobreiros que atualmente existem no local.

Ora o sobreiro é a árvore nacional e tem de ser defendido a todo o custo. Portanto, ainda que alguns sobreiros tenham de ser abatidos, quanto menos, melhor.

A nossa contribuição será calcular a localização exata da pista principal, para que a construção da pista requeira o abate do número mínimo de sobreiros que for possível.

Estudo prévio

Por hipótese, a área reservada para o aeroporto é um retângulo com lados horizontais e verticais (quando vista num mapa, bem entendido).

A pista principal também é um retângulo, de lados paralelos aos da área reservada, e totalmente incluído no retângulo da área reservada.

Para efeitos do nosso estudo, o retângulo da área reservada está dividido em quadrículas de lado unitário e cada sobreiro estará plantado numa dessas quadrículas. Uma vez que as quadrículas correspondem a uma área não muito pequena, pode acontecer que vários sobreiros ocupem a mesma quadrícula.

Escreva um programa que, dados as dimensões da área reservada, as dimensões da pista a construir e a localização de cada sobreiro dentro da área reservada, liste por ordem crescente de número de sobreiros a abater todas as possíveis localizações da pista a construir.

Requisito técnico: a ordenação deve ser feita usando o *insertionsort*.

Submeta no problema A.

Input

A primeira linha do ficheiro de dados contém dois números inteiros, **H** e **V**, que representam a medida do lado horizontal e a medida do lado vertical da área reservada. A segunda linha contém dois números inteiros, **C** e **A**, que representam a medida do lado horizontal e a medida do lado vertical da pista. Seguem-se linhas em número indeterminado, uma para cada sobreiro: cada uma dessas linhas contém dois números inteiros, **X** e **Y**, que representam as coordenadas da quadrícula que contém o sobreiro. Entenda-se por "coordenadas de cada quadrícula", de facto as coordenadas do canto inferior esquerdo da quadrícula.

Output

Cada linha do ficheiro de saída representa uma localização possível para a pista, no interior da área reservada, e o número de sobreiros que seria preciso abater para construir a pista nessa localização.

A localização da pista será representada no ficheiro de saída pelas coordenadas da quadrícula do canto inferior esquerdo do retângulo respetivo.

Assim, cada linha terá três números inteiros, separados por um espaço: a coordenada **X** da localização da pista, a coordenada **Y**, e o número de sobreiros que existem nessa localização (os quais seria preciso abater se a pista fosse construída ali).

O ficheiro de saída estará ordenado pelo número de sobreiros, desempatando pela coordenada **Y** e depois pela coordenada **X**.

Restrições

- Comprimento do lado horizontal da área reservada, **H**: 0 < **H** <= 100.
- Comprimento do lado vertical da área reservada, **V**: 0 < **V** <= 100.
- Comprimento do lado horizontal da pista, **C**: 0 < **C** <= **H**.
- Comprimento do lado vertical da área reservada, **A**: 0 < **A** <= **V**.
- Coordenada **X** de um sobreiro: 0 <= **X** < **H**.
- Coordenada **Y** de um sobreiro: 0 <= **Y** < **V**.

Exemplo

Input

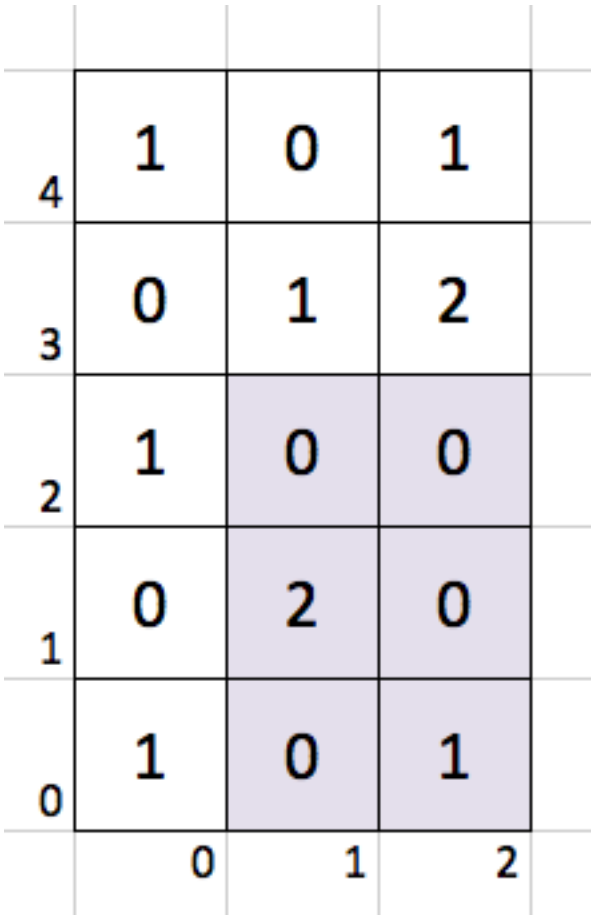
```
3 5
2 3
0 0
2 4
1 1
1 1
1 3
2 0
2 3
2 3
0 2
0 4
```

Output

```
1 0 3
0 2 3
0 0 4
0 1 4
1 2 4
1 1 5
```

Explicação

Os dados correspondem à situação descrita na seguinte figura:



Uma pista de 2 por 3 pode ser colocada em 6 localizações diferentes numa área de 3 por 5. Varrendo as localizações possíveis, da esquerda para a direita e depois de baixo para cima, contamos os seguintes números de sobreiros, em cada caso: 4, 3, 4, 5, 3, 4. Há, portanto, duas localizações onde é possível abater só 3 sobreiros. Dessas, prefere a primeira, assinalada a cinzento) por a coordenada **Y** ser inferior.

Estudo de pormenor

Suponhamos que a área reservada é um quadrado de lado **N** e que a pista é um quadrado de lado **w**. Então, haverá no máximo **(N - w + 1) * (N - w + 1)** pistas possíveis. Para calcular o número de sobreiros em cada uma dessas pistas são precisos **w * w** acessos à matriz. Logo, ao todo, o programa fará **(N - w + 1) * (N - w + 1)* w * w** acessos à matriz.

Colocando esta fórmula no Excel, fazendo **N** igual a 100, por exemplo, e calculando para todos os valores de **w** entre 0 e 100, verificamos que o máximo, 6502500, se atinge para **w** = 50 e **w** = 51.

Podemos especular que o valor máximo se atinge quando a largura da pista é metade da largura da área reservada. Com efeito, a largura da pista é uma fração da largura da área reservada. Ou seja, **w = x*N**, para **x** entre 0 e 1. Nestas condições, o número de acessos será **(N - x*N + 1) * (N - x*N + 1)* x*N * x*N**, ou seja, **N^4 * (1-x-(1/N))^2 * x^2**. Concluímos que para um dado **N** o valor máximo se obtém para o máximo de **(1-x-(1/N))^2* x^2**, para **x** entre 0 e 1. Admitindo que **N** é um valor muito grande, podemos eliminar, "à engenheiro", a parcela **1/N**, ficando com **(1-x)^2* x^2**.

Derivando a função **f(x) = (1-x)^2* x^2**, obtemos facilmente **f'(x) = 2x*(2x^2-3x+1)**. Facilmente também, calculamos que a derivada tem zeros em

0, 0.5 e 1. O primeiro e o terceiro são mínimos locais de **f** e o segundo é um máximo local.

O Wolfram Alpha confirma isto :-)

Concluímos que, para um valor de **N** grande, o número máximo de acessos à matriz nos cálculos realizados pelo nosso programa é **(N/2)^4**, ou **N^4/16**, o que está de acordo com as nossas contas no Excel.

Em geral, um algoritmo cujo tempo de cálculo cresça com a quarta potência do tamanho dos dados não conseguirá processar dados volumosos em tempo útil.

Acumulação

A segunda parte do trabalho consiste em substituir o cálculo do número de sobreiros até agora realizado "à bruta" (somando todos os números presentes "dentro" da localização da pista) pelo cálculo que recorre à matriz de acumulação.

Usando esta técnica, o número de sobreiros em cada localização é calculado usando quatro acessos à matriz de acumulação. Portanto, o tempo de cálculo fica a depender apenas do número de localizações possíveis. Ora o número máximo de localizações possíveis ocorre quando a pista é um quadrado de lado unitário: neste caso há **N^2** localizações. Portanto, na pior das hipóteses o número de acessos à matriz de acumulação, durante os cálculos será **N^2*4**. A estes há que somar os acessos à matriz original e à matriz de acumulação, durante a construção da matriz de acumulação. Ora, o cálculo do valor para cada posição da matriz de acumulação, exceto a primeira linha e a primeira coluna, envolve um acesso à matriz original e três acessos à matriz de acumulação. Logo, ao todo teremos por cada cálculo, cinco acessos às matrizes. Havendo **N*N** valores a calcular concluímos que haverá **5N^2+2N+1** acessos ao todo. A parcela **2N+1** corresponde à inicialização a zero da primeira linha e da primeira coluna, e é desprezável, face à parcela em **5N^2**.

Concluímos que o número total de acessos às matrizes, na acumulação e nos cálculos é da ordem de **N^2**. Portanto, o tempo de cálculo nesta fase das operações é proporcional a **N^2**.

Requisito técnico: a ordenação deve ser feita usando o *qsort*. Perante as novas restrições (ver abaixo), o *insertionsort* não correria em tempo útil.

Submeta no problema B.

Novas restrições

Nesta segunda parte do problema, o programa deve ser capaz de aceitar áreas reservadas maiores:

- Comprimento do lado horizontal da área reservada, **H**: 0 < **H** <= 1000.
- Comprimento do lado vertical da área reservada, **V**: 0 < **V** <= 1000.

Output

Para evitar sobrecarregar o Mooshak com ficheiros de output muito volumosos, o seu programa deve apenas escrever os 10000 primeiros valores do array ordenado, ou todos se forem menos de 10000.

Note bem: o Mooshak tem um limite para o tamanho de output permitido. Se o seu programa escrever mais do que o permitido obterá o erro de "*Output Limit Exceeded*" ou "*Invalid Function*".

Na verdade, o output dos testes para o problema A nunca excede 10000 linhas, pelo que esta regra apenas é necessária para o problema B.

Atenção à memória disponível na pilha de execução

Note que uma matriz de 1000×1000 ocupa 8 megabytes. Logo, estará muito próximo do limite de memória na pilha de execução. Se for preciso usar duas matrizes dessas, é melhor pensar em alocá-las na memória dinâmica…