MC202 — ESTRUTURAS DE DADOS

Lab 11 — Grafos

Problema

Os reinos de Totorando e Wano sempre viveram em conflitos e, em sua última discussão, a rainha de Totorando, Big Mom, declarou guerra contra o reino de Wano. Desde então, os reinos estão se abastecendo, se organizando e elaborando estratégias para o dia da grande batalha. O rei de Wano, Kaido, dotado de astúcia, decidiu espionar a base de operações do reino de Totorando e contatou sua equipe de inteligência. A equipe sugeriu que fosse utilizado um drone para espionar as atividades do reino de Totorando. Contudo, o drone mais avançado que o reino possui tem certas limitações: o drone só pode sobrevoar até determinada altura e, além disso, ele só consegue se movimentar para a direita, esquerda, frente e trás, um movimento de cada vez, sendo incapaz de realizar movimentos simultâneos. Por tal motivo, a equipe de estratégia tem um grande problema, pois o terreno entre Wano e Totorando é cercado por monólitos dos quais o drone não consegue sobrevoar por serem muito altos. A equipe de estratégia teve que anunciar ao rei Kaido que ninguém da equipe sabia como contornar esse problema, contudo eles apresentaram ao rei uma solução: eles falaram ao rei sobre um grupo de aprendizes de MC202 que são especialistas em resolver problemas e que provavelmente seriam capazes de resolver este. Rapidamente, Kaido contatou a equipe e solicitou que os aprendizes elaborassem um programa que, dada a descrição da região entre Wano e Torando, retornasse uma possível rota que o drone possa realizar até a base de operações de Totorando.

Entrada

A primeira linha contém três inteiros, $H \geq 1$, $X \geq 0$ e $Y \geq 0$, que indicam, respectivamente, a altura máxima que o drone pode sobrevoar e a sua posição inicial (X, Y) no mapa. Na segunda linha, são dados quatro inteiros, $A \geq 0$, $B \geq 0$, $M \geq 0$ e $N \geq 0$, onde (A, B) é a coordenada da base de operações e M e N determinam o número de linhas e colunas do mapa, respetivamente. Em seguida são dadas M linhas contendo N inteiro positivos cada uma. Um mapa nada mais é do que a altura do relevo de cada m^2 da região. Exemplificando, se a posição (5, 4) do mapa tem o valor 5, significa que no ponto (5, 4) existe uma elevação de altura 5, assim, se o drone puder sobrevoar a altura no máximo 4, então ele não poderá passar por esse ponto.

Todos os mapa possuem um caminho saindo da posição de origem do drone até a base de operações.

Saída

A saída do seu programa deve imprimir um mapa com as mesmas dimensões do mapa dado na entrada, tal que em cada posição por que o drone não passa é impresso o

caractere ‘#’ e nas posições que representam o caminho do drone deve ser impressa a ordem em que seus movimentos devem ser realizados, iniciando de 0, que é sua posição inicial (veja o exemplo). Lembre-se de que o caminho que o drone deve realizar pode ser qualquer caminho possível, considerando suas limitações.

Exemplo

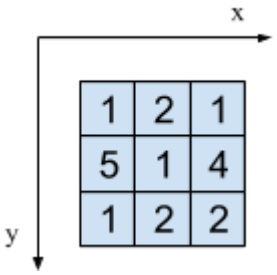
Entrada

```
3 0 0
1 2 3 3
1 2 1
1 5 2
6 1 3
```

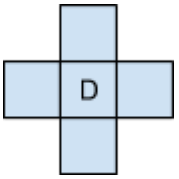
Saída

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

Informações



Exemplo de mapa.



Movimentos válidos do drone.

Nos arquivos auxiliares do susy há um script utilizado para verificar se o caminho está correto. Para executá-lo, use o seguinte comando:

`python3 compara.py <solução> <entrada>`

Por exemplo:

`python3 compara.py arq01.out arq01.in`

onde `arq01.out` é a saída do seu programa e `arq01.in` é a entrada (cada um dos testes no SuSy).

Observações

- Neste laboratório, é obrigatório modelar o problema como um problema em grafo. Adicione um pequeno comentário no cabeçalho de seu programa explicando quem são os vértices e quem são as arestas do seu grafo.
- Para as turmas A, B e C, esse laboratório tem peso 4.
- Deverão ser submetidos os seguintes arquivos: **`lab11.c`** e outros arquivos auxiliares para solução do problema.
- Observações sobre SuSy:
    - Versão do GCC: 4.4.7 20120313 (Red Hat 4.4.7-17).
    - *Flags* de compilação:
        - `-ansi -Wall -pedantic-errors -Werror -g -lm`
    - Utilize comentários do tipo `/* comentário */`;
      comentários do tipo `//` serão tratados como erros pelo SuSy
    - Tempo máximo de execução: 1 segundo.
    - Evite usar o sistema como compilador ou apenas para testar as instâncias.
    - Sempre use o software valgrind para detectar erros decorrentes do uso incorreto da memória dinâmica **em todos os testes abertos**. Relembrando:
        - `valgrind --leak-check=full ./lab11 < arq01.in`
- Além das observações acima, esse laboratório será avaliado pelos critérios:
    - Indentação de código e outras boas práticas, tais como:
        - uso de comentários (apenas quando são relevantes);
        - código simples e fácil de entender;
        - sem duplicidade (partes que fazem a mesma coisa).
    - Organização do código:
        - arquivo de cabeçalho (.h) contendo definições de constantes, tipos de dados criados pelo usuário, protótipo de funções etc;
        - arquivos de implementação (.c) que implementam as funções definidas no arquivo de cabeçalho;
        - tipos de dados criados pelo usuário e funções bem definidas e tão independentes quanto possível.
    - Corretude do programa:
        - programa correto e implementado conforme solicitado no enunciado;
        - desalocar toda memória alocada dinamicamente durante a execução do programa;
        - não realizar leitura ou escrita em blocos de memória não alocados;
        - inicialização de variáveis sempre que for necessário;
        - dentre outros critérios.