Algoritmos e Estruturas de Dados 2
Trabalho 2

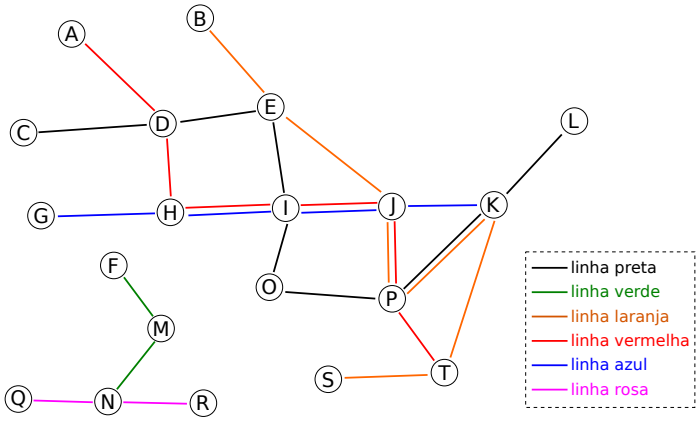
Prof. Ricardo Oliveira - 2025/1

Enunciado

Você acabou chegar na grande metrópole de Towleedou, e está aprendendo como funciona o sistema de transporte público por metrô da cidade.

Há N estações de metrô na cidade. As estações são identificadas pelas N primeiras letras do alfabeto (estação A, estação B, etc.). Há L linhas de metrô no sistema. Cada linha possui um nome próprio, e passa por uma sequência de estações, em ambos os sentidos.

Considere o seguinte exemplo do sistema de metrô:



Neste exemplo, há 20 estações (de A até T) e 6 linhas de metrô (preta, verde, laranja, vermelha, azul e rosa). A linha preta passa pelas estações $C \to D \to E \to I \to O \to P \to K \to L$ e também, no sentido oposto, pelas estações $L \to K \to P \to O \to I \to E \to D \to C$; a linha azul passa pelas estações $G \to H \to I \to J \to K$ e também pelas estações $K \to J \to I \to H \to G$; e assim por diante.

Escreva um programa em C que lê do usuário a descrição do sistema de metrô da cidade e, em seguida, lê uma série de consultas do tipo a b e determina o menor caminho (em número de estações) para ir da estação a até a estação b (imprimindo instruções para tal – veja exemplo abaixo), ou determina que não é possível fazer tal caminho. Por fim, seu programa deve imprimir uma lista de *redes integradas* e suas respectivas estações (duas estações estão em uma mesma rede integrada se existe um caminho possível de uma para outra).

Entrada e Saída

Seu programa deve ler a entrada do usuário. A entrada começa com um inteiro N indicando o número de estações. A próxima linha contém um inteiro L, o número de linhas de metrô. Cada linha de metrô é descrita em uma linha da entrada. Cada linha da entrada inicia com o nome da linha de metrô, seguido de um inteiro K indicando o número de estações pelas quais a linha passa. Em seguida a linha conterá K letras (separadas por espaço), indicando as estações. Após, a entrada conterá uma sequência de consultas, cada uma na forma a b. Para cada consulta, imprima o número mínimo de estações necessárias a passar para ir de a e b, seguido de uma sequência de instruções de trocas de linha, conforme o exemplo de saída abaixo. A entrada termina com `* *`. Ao final da execução, imprima a lista de redes integradas do sistema, indicando quais estações estão em cada rede, conforme o exemplo abaixo.

Exemplo de entrada	Exemplo de saída
20 6 preta 8 C D E I O P K L verde 3 F M N laranja 7 B E J P K T S vermelha 7 A D H I J P T azul 5 G H I J K rosa 3 Q N R A H A T L G F J S O C I A Q F R * *	De A para H: 3 estacoes Na estacao A pegue a linha vermelha Desca na estacao H --- De A para T: 6 estacoes Na estacao A pegue a linha vermelha Na estacao D troque para a linha preta Na estacao E troque para a linha laranja Na estacao P troque para a linha vermelha Desca na estacao T --- De L para G: 6 estacoes Na estacao L pegue a linha preta Na estacao K troque para a linha azul Na estacao J troque para a linha vermelha Na estacao H troque para a linha azul Desca na estacao G --- Impossivel ir de F para J. --- De S para O: 4 estacoes Na estacao S pegue a linha laranja Na estacao T troque para a linha vermelha Na estacao P troque para a linha preta Desca na estacao O --- De C para I: 4 estacoes Na estacao C pegue a linha preta Desca na estacao I --- Impossivel ir de A para Q. --- De F para R: 4 estacoes Na estacao F pegue a linha verde Na estacao N troque para a linha rosa Desca na estacao R --- Rede Integrada 1: A D C E I O P K L T S J H G B Rede Integrada 2: F M N Q R

Na saída de uma consulta, qualquer caminho que passe pelo *menor* número possível de estações pode ser impresso, independentemente das trocas de linha. Por exemplo, no caminho da estação A para a estação T, também é possível trocar da linha laranja para a linha vermelha na estação J ao invés de na estação P como na saída de exemplo. Como outro exemplo, no caminho de C para I, ao invés de fazer todo o trajeto na linha preta, é possível trocar para a linha vermelha na estação D e passar pela estação H ao invés de passar pela estação E – as duas opções passam por quatro estações ao todo (que é o menor número possível de estações entre C e I), e portanto ambas poderiam ser impressas pelo seu programa. Por fim, as redes integradas e suas estações podem ser impressas em qualquer ordem.

Implementação

- Utilize a **lista de adjacência** para representar o grafo;
- Para cada consulta, use uma **Busca em Largura (BFS)** para definir o menor caminho entre duas

estações. Seu trabalho deve *obrigatoriamente* conter e incluir uma biblioteca de Fila para a implementação da BFS. A Fila pode ser implementada tendo como base um vetor ("estática") ou uma lista ligada ("dinâmica"), à sua escolha.

- Para determinar as redes integradas, use uma **Busca em Profundidade (DFS)**. Você pode escolher entre implementar a DFS de maneira recursiva ou de maneira iterativa. Caso opte pela implementação iterativa, seu trabalho deve conter e incluir também uma biblioteca de Pilha, implementada com um vetor ("estática") ou com uma lista ligada ("dinâmica"), como preferir;
- Você pode assumir que haverá no máximo 26 estações e no máximo 50 linhas de metrô. Você também pode assumir que o nome de cada linha de metrô terá no máximo 50 letras, todas minúsculas;
- Não esqueça de liberar toda a memória alocada pelo seu programa ao final da execução.

Orientações

- O trabalho pode ser feito por equipes de *até* 2 (dois) estudantes;
- Submeta, via *Moodle*, um pacote **zip** ou **tar.gz**[1] contendo todo o código-fonte necessário para compilar e executar seu trabalho, além de um arquivo de texto (txt) onde conste:
  - O nome de todos os integrantes da equipe;
  - Toda informação que a equipe julgar relevante para a correção (como *bugs* conhecidos, detalhes de implementação, escolhas de projeto, etc.)
- **Não** inclua no pacote arquivos de configuração da sua IDE (arquivos do CodeBlocks/VSCode/etc), *sob pena de redução da nota*. Esses arquivos não são importantes para a compilação e execução do código, e incluí-los apenas atrasa a correção;
- Comente adequadamente seus códigos para facilitar a correção;
- Atenção: a correção será parcialmente automatizada, e a saída do programa será testada com outras entradas além das fornecidas como exemplo. *Siga* ***fielmente*** *o formato de saída dado nos exemplos*, sob pena de grande redução da nota;
- Certifique-se que seu programa compila e funciona antes de submetê-lo;
- O trabalho deve ser entregue até **22 de Junho de 2025, 23:59**, apenas via *Moodle*. Trabalhos entregues por outros meios ou fora do prazo não serão aceitos. É suficiente que o trabalho seja submetido por apenas um estudante da equipe;
- Trabalhos detectados como cópia/plágio (de colegas, da internet ou de ferramentas de IA) ou comprados receberão **todos** a nota 0 (**ZERO**) e estarão sujeitos a abertura de Processo Administrativo Disciplinar Discente.

[1] Outros pacotes (rar/7zip/etc.) não serão aceitos