MC558 - Projeto e Análise de Algoritmos II
Instituto de Computação

Prof. Flávio Keidi Miyazawa

PED
Hismael Costa (hismael.costa@gmail.com)

TRABALHO PRÁTICO 1

Objetivo

Este trabalho tem por objetivo a auxiliar no entendimento de conceitos relacionados a *grafos dirigidos* através da implementação de algoritmos. Os principais conceitos abordados nesse trabalho são:

- Grau de entrada e saída de um vértice;
- Grafos dirigidos fortemente conexos;
- Trilha fechada euleriana.

Descrição

Escreva um programa que lê um arquivo de texto contendo um grafo dirigido $G = (V, A)$ e recebe dois parâmetros, o nome de G e o nome de um vértice $v \in V(G)$. O seu programa deve realizar as seguintes verificações:

- Se existir $u \in V(G)$, tal que, o grau de entrada de u é diferente do grau de saída, deve ser mostrada a seguinte mensagem: "*Erro: Existe vértice inviável.*" e a execução deve ser interrompida;
- Se G não for fortemente conexo, deve ser mostrada a seguinte mensagem: "*Erro: Grafo não eh fortemente conexo.*" e a execução deve ser interrompida.

Após o seu programa fazer todas as verificações acima e caso não seja finalizado por nenhuma, ele deve imprimir uma sequência de vértices que formam uma **Trilha Fechada Euleriana começando e terminando em** v. A construção da trilha deve ser realizada em *tempo linear no tamanho do grafo*.

Para construir uma trilha euleriana fechada, há possibilidades distintas para a escolha dos vértices na sequência, cabe ao aluno implementar o critério de escolha que achar mais adequado.

Implementação e Execução

Entrada e Saída

Como descrito acima, você receberá um arquivo de entrada contendo um grafo, com o seguinte formato:

# vértices	*# arestas*
Origem aresta 0	*Destino aresta 0*
⋮	⋮
Origem aresta m-1	*Destino aresta m-1*

A saída impressa da solução deve ser:

Índice da primeira aresta da trilha	*Origem da primeira aresta*	*Destino da primeira aresta*
Índice da segunda aresta da trilha	*Origem da segunda aresta*	*Destino da segunda aresta*
⋮	⋮	⋮
Índice da última aresta da trilha	*Origem da última aresta*	*Destino da última aresta*

Implementação

Serão disponibilizadas duas rotinas, uma para a leitura do grafo e outra para verificação da trilha.

A rotina de leitura retornará um objeto do tipo Grafo, caso nenhum erro de leitura seja disparado. Caso exista esse erro, o mesmo deve ser mostrado.

Após a leitura, deve-se implementar uma rotina com o seguinte cabeçalho:

bool trilha_euleriana (int n, int m, Grafo g, int origem[], int destino[], int trilha[], string mensagem, int RA)

- **n** e **m** são os números de vértices e arestas, respectivamente;
- **g** objeto do tipo Grafo, retornado a partir da leitura do arquivo;
- **origem** e **destino** são vetores de *m* posições, tal que, *(origem[0], destino[0])* é o par dos vértices de origem e destino da primeira aresta da trilha, e assim, sucessivamente;
- A rotina deve retornar **Verdadeiro** ou **Falso**, conforme o grafo tem ou não uma trilha euleriana fechada;
- A solução também deve devolver o vetor **trilha** com as *m* posições preenchidas após a execução da rotina;

- Caso não seja possível criar uma trilha euleriana para o grafo informado, escrever o motivo do erro na variável **mensagem** e escrever no programa principal;
- A variável **RA** serve para que você armazene seu RA, facilitando assim a correção nos testes que serão feitos no seu algoritmo.

Verificação

Após a execução da rotina *trilha_euleriana*, deve-se verificar se essa trilha está correta, antes de imprimir a saída. A rotina para a verificação será disponibilizada, necessitando apenas que o aluno faça seu uso.

Submissão

O trabalho deverá ser enviado por por e-mail com o assunto "Entrega do Teste Prático 1 - RA 999999", onde 999999 deve ser trocado pelo número do seu RA, para o endereço *h265684@dac.unicamp.br* e até as **23:59 do dia 13/10**. Antes de submeter seu trabalho, verifique se ele atende todas as especificações de submissão descritas abaixo:

- Dois arquivos devem ser submetidos:
  - *trilha_euleriana_principal.cpp*: Neste arquivo deve conter a leitura, chamada para a função de geração da trilha, verificação e impressão da mesma. O aluno não deve criar ou alterar as funções deste arquivo, com exceção da *main*;
  - *trilha_euleriana_ra999999.cpp*: Neste arquivo você deve implementar sua rotina de criação da trilha. Você também pode incluir outras rotinas, caso ache necessário. Deve-se trocar o *999999* no nome do arquivo, pela sua matrícula.
- O seu algoritmo deve ter complexidade de tempo linear no tamanho do grafo;
- O código deve estar comentado, de forma a facilitar o entendimento e correção;
- A implementação deve ser feita utilizando C++.

Avaliação

A avaliação será da seguinte forma:

- **70%** da nota será baseada na correta implementação do algoritmo;
- E os outros **30%** restantes serão baseados nos comentários e facilidade de leitura do programa.