# Instruções sobre o Exercício 3

1000608/1001323 — Algoritmos em Grafos

Cândida Nunes da Silva

2º Semestre de 2020 – ENPE

## 1 Problema – Engenharia de Tráfego

Um engenheiro de tráfego recebeu a tarefa de melhorar o escoamento do trânsito de uma cidade. Para tal ele recebeu um mapa da cidade definindo todos os cruzamentos da cidade e, para cada trecho de rua entre dois cruzamentos, a mão do trecho; isto é, se é de mão dupla ou de mão única, e qual o sentido dos trechos de mão única. O mapa que lhe foi dado da organização atual garante que de qualquer cruzamento da cidade se pode chegar em qualquer outro cruzamento. Este é um requisito fundamental para a organização do trânsito da cidade, caso contrário um cidadão poderia ser forçado a desrespeitar as mãos de direção para poder voltar para casa. O engenheiro tem uma proposta de como uma reorganização das mãos de direção poderia melhorar o escoamento do trânsito; porém ele não consegue garantir que a sua proposta satisfaz o requisito fundamental de permitir acesso a qualquer cruzamento a partir de qualquer outro cruzamento. Então ele contratou você, um programador experiente, para avaliar se a proposta dele satisfaz esse requisito ou não.

## 2 Entrada

A primeira linha de cada caso de teste contém dois inteiros $N$ e $M$, separados por um espaço em branco, que representam, respectivamente, quantos cruzamentos ($2 \leq N \leq 300$) e quantos trechos de ruas entre cruzamentos há na cidade ($1 \leq M \leq 600$). Cada cruzamento é representado por um inteiro entre 0 e $N - 1$. Cada uma das $M$ linhas subsequentes de cada caso de teste contém três inteiros $A$, $B$ e $D$ ($0 \leq A, B < N$, $1 \leq D \leq 2$), separados por um espaço em branco, indicando que existe um trecho de rua que liga os cruzamentos $A$ e $B$; se $D = 1$ então o trecho é de mão única e vai do cruzamento $A$ para o cruzamento $B$ e se $D = 2$ o trecho é de mão dupla. Você pode supor que para cada par de cruzamentos $A$ e $B$ existe no máximo um trecho de rua que os liga.

## 3 Saída

A saída deve ser escrita na saída padrão (terminal). A saída deve ser uma única linha, contendo o texto “Adequado.” sempre que a proposta do engenheiro satisfizer o requisito fundamental ou o texto “Inadequado.” caso contrário.

## 4 Exemplos

| Entrada | Saída |
|---|---|
| 10 18<br>0 1 2<br>1 2 1<br>1 3 1<br>1 4 2<br>2 3 1<br>4 0 1<br>5 1 1<br>5 7 2<br>6 1 1<br>6 2 1<br>6 3 1<br>6 9 1<br>7 4 1<br>7 8 2<br>8 0 1<br>8 9 2<br>9 3 1<br>9 5 2 | Inadequado. |

| Entrada | Saída |
|---|---|
| 10 20<br>0 1 2<br>1 2 1<br>1 3 1<br>1 4 1<br>2 0 1<br>2 3 2<br>4 0 1<br>5 1 1<br>5 2 1<br>5 7 2<br>6 1 1<br>6 2 1<br>6 3 2<br>6 9 1<br>7 4 2<br>7 8 2<br>8 0 1<br>8 9 2<br>9 3 1<br>9 5 2 | Adequado. |

## 5 Desenvolvimento e Apresentação

Cada aluno deve implementar a sua solução individual. A implementação da solução do problema deve ser em C em arquivo único. O nome do arquivo deve estar na forma "ex03-nomesn.c", onde "nomesn" representa o primeiro nome do aluno seguido das iniciais de seu sobrenome. Note que todas as letras são minúsculas e o separador é "-" (hífen) e não "_" (underscore).

O juiz online verificará seu programa comparando para cada um dos casos de teste se a saída gerada pelo seu programa é igual à saída esperada. É **imprescindível** que o **algoritmo** implementado esteja correto, isto é, retorne a solução esperada para **qualquer** entrada. É **desejável** que a implementação seja eficiente.

## 6 Ambiente de Execução e Testes

O programa deve ser compilável em ambiente Unix com `gcc`. Sugere-se que os testes também sejam feitos em ambiente Unix. Deve-se esperar que a entrada seja dada na entrada padrão (teclado) e

não por leitura do arquivo de testes. Da mesma forma, a saída deve ser impressa na saída padrão (terminal), e não em arquivo.

A motivação dessa exigência é apenas simplificar a implementação de entrada e saída, permitindo o uso das funções `scanf` e `printf` da biblioteca padrão para leitura e escrita dos dados, sem precisar manipular arquivos.

Por outro lado, é evidente que efetivamente entrar dados no teclado é muito trabalhoso. Em ambiente Unix, é possível usar redirecionamento de entrada na linha de comando de execução para contornar esse problema. Supondo que o nome do arquivo executável seja análogo ao arquivo fonte, e "ex03.in" seja o arquivo com os casos de teste, a linha de comando:

```
shell$ ./ex03-nomesn < ex03.in
```

executa o programa para todos os casos de teste de uma só vez, retornando todas as saídas em sequência para o teminal. Novamente, pode-se usar o redirecionamento de saída na linha de comando para escrever a saída em um arquivo de saída de nome, por exemplo, "ex03.my.out". A respectiva linha de comando seria:

```
shell$ ./ex03-nomesn < ex03.in > ex03.my.out
```

Após a execução, a comparação pode ser feita usando o comando `diff` do Unix. Por exemplo, se o arquivo "ex03.out" contém as saídas esperadas, a linha de comando:

```
shell$ diff ex03.out ex03.my.out
```

serve para comparar automaticamente os dois arquivos, retornando nada caso sejam idênticos e as linhas onde há discrepâncias caso contrário.

# 7 Notas

As notas serão baseadas na correção da solução implementada, clareza do código fonte e eficiência da solução.

Trabalhos que não atendam aos requisitos mínimos expressos neste documento de forma a invibializar o teste do programa receberão nota ZERO. Em particular, receberá nota ZERO todo programa que:

- não compila em ambiente Unix;
- dá erro de execução;
- não usa entrada e saída padrão para leitura e escrita.