Instruções sobre o Exercício 5

1001323 — Algoritmos em Grafos

Cândida Nunes da Silva

1º Semestre de 2022

1 Problema – Vingança

Você acabou de descobrir que seu namorado(a) te traiu, e está sendenta(o) por vingança. Como você sabe que seu namorado(a) adora assistir séries e jogar em rede, você sabe que a melhor vingança é deixá-lo(a) sem acesso à Internet. Para isso, você deseja cortar alguns fios que transportam dados do provedor de Internet até a casa de seu namorado(a). Desta forma, levará alguns dias ou semanas até que a conexão seja reestabelecida.

No entanto, você não conhece exatamente quais são as conexões que precisam ser cortadas para se vingar de seu namorado(a). Então você decidiu subornar um funcionário do provedor para te fornecer um mapa da rede de conexões do provedor até a casa de seu namorado(a). Este funcionário também lhe indicou uma pessoa que poderia cortar os fios por você, mediante pagamento. Ao montar o mapa das conexões o funcionário registrou, para cada conexão, quantos reais a pessoa indicada cobra para cortar a fiação que corresponde àquela conexão.

Agora você precisa avaliar se contratar a pessoa indicada para realizar seu plano de vingança é possível, ou seja, se o custo de contratá-lo cabe em seu orçamento. O custo de contratá-lo depende de quais conexões serão cortadas. Então sua tarefa agora é fazer um programa que determina, dada a rede de conexões passada na entrada com os respectivos custos de cortar cada conexão, qual é o custo mínimo para cortar a conexão do provedor à casa de seu namorado(a).

Seu algoritmo deve ter complexidade $O(n^3m)$, onde n é o número de pontos de recepção e/ou transmissão de dados e m é o número de conexões na rede entre pontos de recepção e/ou transmissão de dados.

2 Entrada

A entrada deve ser lida da entrada padrão (teclado). A primeira linha de cada caso de teste contém dois inteiros N e M, separados por um espaço em branco, que representam, respectivamente, quantos são os pontos de recepção e/ou transmissão de dados ($1 \leq N \leq 500$) e quantas são as conexões na rede entre pontos de recepção e/ou transmissão de dados ($1 \leq M \leq 10000$). Cada ponto de recepção e/ou transmissão de dados é representado por um inteiro entre 0 e $N-1$, sendo que 0 representa o provedor de Internet e $N-1$ representa a casa se seu namorado(a). Cada uma das M linhas subsequentes de cada caso de teste contém três inteiros A, B e C ($0 \leq A, B \leq N-1$ e $1 \leq C \leq 3000$), separados por um espaço em branco, indicando que existe uma conexão que transmite dados do ponto A até o ponto B e que custa C reais para cortar esta conexão.

3 Saída

A saída deve ser escrita na saída padrão (terminal). Para cada caso de teste seu programa deve imprimir uma linha com o valor mínimo em reais que custaria para contratar os serviços da pessoa indicada para efetuar seu plano de vingança.

4 Exemplo

Entrada	Saída
5 9 0 1 1000 0 2 1 0 3 3000 1 3 2000 1 4 3000 2 1 1000 2 3 1000 3 4 1000 2 4 1	2001

Entrada	Saída
9 17 0 1 7 0 2 2 0 4 9 1 2 5 1 3 2 2 4 1 2 5 3 3 6 6 4 2 7 4 7 4 5 3 1 5 6 5 5 7 10 6 5 4 6 8 2 7 6 6 7 8 11	9

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 "ex05-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 "ex05.in" seja o arquivo com os casos de teste, a linha de comando:

```
shell$ ./ex05-nomesn < ex05.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, "ex05.my.out". A respectiva linha de comando seria:

```
shell$ ./ex05-nomesn < ex05.in > ex05.my.out
```

Após a execução, a comparação pode ser feita usando o comando `diff` do Unix. Por exemplo, se o arquivo "ex05.out" contém as saídas esperadas, a linha de comando:

```
shell$ diff ex05.out ex05.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.
- apresentar fortes indícios de cola.