Instruções sobre o Exercício 7

1000608/1001323 — Algoritmos em Grafos

Cândida Nunes da Silva

2º Semestre de 2020 – ENPE

1 Problema – Alocação de Professores

O coordenador de um curso universitário precisa alocar professores para as disciplinas a serem oferecidas no próximo semestre do curso. Recentemente houve grandes mudanças no quadro docente disponível para ministrar as disciplinas com o objetivo de cortar custos. E também com esse mesmo objetivo a universidade determinou que paga a cada docente apenas o valor base para lecionar um único curso por semestre. Foi aberta aos docentes a possibilidade de lecionar mais de um curso por semestre, porém a partir do segundo curso seria uma "colaboração voluntária" por parte do docente para com a universidade. Em vista destas mudanças, o coordenador já não está mais certo se o corpo docente consegue oferecer todas as disciplinas necessárias no próximo semestre.

Para avaliar se o corpo docente está adequado, e pedir apoio aos seus superiores para resolver a questão em caso negativo, o coordenador pediu a cada docente que manifestasse se aceitava contribuir voluntariamente ministrando mais de uma disciplina. Nenhum docente concordou com a contribuição voluntária, portanto cada um aceita ministrar no máximo uma disciplina. O coordenador também pediu a cada docente que listasse quais das disciplinas que precisam ser ofeceridas o docente pode ministrar de forma satisfatória. Com base nas listas de cada docente, o coordenador precisa agora determinar o número máximo de disciplinas que pode ser atendido pelo corpo docente atual e precisa da sua ajuda para implementar um programa que calcule esse valor.

2 Entrada

A entrada deve ser lida da entrada padrão (teclado). A primeira linha de cada caso de teste contém três inteiros N, K e M, separados por um espaço em branco, que representam, respectivamente, quantos são os docentes do corpo docente ($1 \leq N \leq 250$), quantas são as disciplinas a serem oferecidas ($1 \leq K \leq 250$), e a soma dos tamanhos das listas entregues pelos docentes ($1 \leq M \leq 500$). Cada uma das M linhas subsequentes de cada caso de teste contém dois inteiros A e B ($0 \leq A < N$ e $N \leq B < N + K$), separados por um espaço em branco, indicando que o docente A pode ministrar a disciplina B.

3 Saída

A saída deve ser escrita na saída padrão (terminal). Para cada caso de teste seu programa deve imprimir uma linha com o número máximo de disciplinas que pode ser atendido pelo corpo docente atual.

4 Exemplo

Entrada	Saída
4 5 12 0 4 0 7 1 4 1 5 1 6 1 8 2 4 2 5 2 7 3 5 3 6 3 8	4

Entrada	Saída
9 9 21 0 9 0 10 0 11 1 9 1 10 1 12 2 12 2 14 3 12 3 13 3 14 3 15 4 12 4 14 5 12 5 14 6 14 6 16 6 17 7 17 8 17	7

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 "ex07-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 "ex07.in" seja o arquivo com os casos de teste, a linha de comando:

```
shell$ ./ex07-nomesn < ex07.in
```

executa o programa para todos os casos de teste de uma só vez, retornando todas as saídas em sequência para o terminal. 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, "ex07.my.out". A respectiva linha de comando seria:

```
shell$ ./ex07-nomesn < ex07.in > ex07.my.out
```

Após a execução, a comparação pode ser feita usando o comando `diff` do Unix. Por exemplo, se o arquivo "ex07.out" contém as saídas esperadas, a linha de comando:

```
shell$ diff ex07.out ex07.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.