Instruções sobre o Trabalho 3

1000608/CCS025 — Teoria dos Grafos

Cândida Nunes da Silva

2º Semestre de 2018

1 Problema

Resolva o problema Ir e Vir da regional da Maratona de Programação de 2010, respeitando o formato de entrada e saída especificado no enunciado. A entrada deve ser lida da entrada padrão (teclado) e a saída deve ser escrita na saída padrão (terminal).

2 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 "t3-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).

Serão disponibilizados arquivos com diversas entradas (t3.in) e as respectivas saídas esperadas (t3.sol). É **imprescindível** que o **algoritmo** implementado esteja correto, isto é, retorne a solução esperada para **qualquer** entrada do arquivo de testes.

É **desejável** que a implementação seja eficiente. Serão consideradas **ineficientes** as implementações que demandem mais do que 1 segundo para executar para o caso de teste disponibilizado.

3 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 "t3.in" seja o arquivo com os casos de teste, a linha de comando:

```
shell$ ./t3-nomesn < t3.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, "t3.my.sol". A respectiva linha de comando seria:

```
shell$ ./t3-nomesn < t3.in > t3.my.sol
```

Após a execução, a comparação pode ser feita usando o comando `diff` do Unix. Por exemplo, se o arquivo "t3.sol" contém as saídas esperadas, a linha de comando:

```
shell$ diff t3.sol t3.my.sol
```

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

4 Entrega e Prazos

A data para a entrega do projeto é dia 30 de outubro. Cada aluno deve entregar via moodle seu único arquivo fonte com nome no padrão requerido até essa data. Lembre-se que é **imprescindível** que o código compile em ambiente Unix e que a entrada e a saída sejam feitas pela entrada e saída padrão.

5 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.

É recomendado que se esforcem para que o programa compute corretamente e em tempo adequado as soluções do primeiro caso de teste do online judge, que é exatamente o caso de teste disponibilizado no moodle. Boa parte da nota do trabalho depende do resultado para esse caso de teste.