MC202 — ESTRUTURAS DE DADOS

Laboratório 15 — Grafos 2

Introdução

Luciano ouviu dizer que entre quaisquer duas pessoas do mundo existe uma conexão formada de no máximo 6 graus de separação, por exemplo:

No grafo acima, a aresta entre Luciano e Maria quer dizer que ambos se conhecem e assim por diante. O que ele ouviu quer dizer é que o menor caminho entre quaisquer duas pessoas do mundo contém no máximo 6 arestas.

Luciano achou isso estranho. Como tinha acesso a informações de grupos da sua universidade e dos participantes desses grupos, resolveu verificar a informação para os alunos de sua universidade. Mas a quantidade de informação é muito grande, então ele não conseguirá fazer isso sozinho.

É aqui que você entra. Você deve fazer um programa que recebe as listas de participantes dos grupos da universidade e descobre, para cada um, a distância entre esse aluno e o aluno da universidade com maior grau de separação. Suponha que, se duas pessoas estão em um mesmo grupo, elas se conhecem. Além disso, se, para algum aluno, houver outro(s) aluno(s) que não têm conexão com o primeiro, então você deve listar esse(s) aluno(s).

Entrada

A entrada do programa começa com dois valores inteiros n e k, em que n é a quantidade total de alunos e k é a quantidade total de grupos. Em seguida, cada linha começa com a quantidade de pessoas em um grupo e o identificador das pessoas que pertencem a esse grupo. O identificador de cada pessoa é um valor de **0** a n**-1**.

```
8 4
1 1
2 2 3
4 3 4 5 6
3 6 7 0
```

Saída

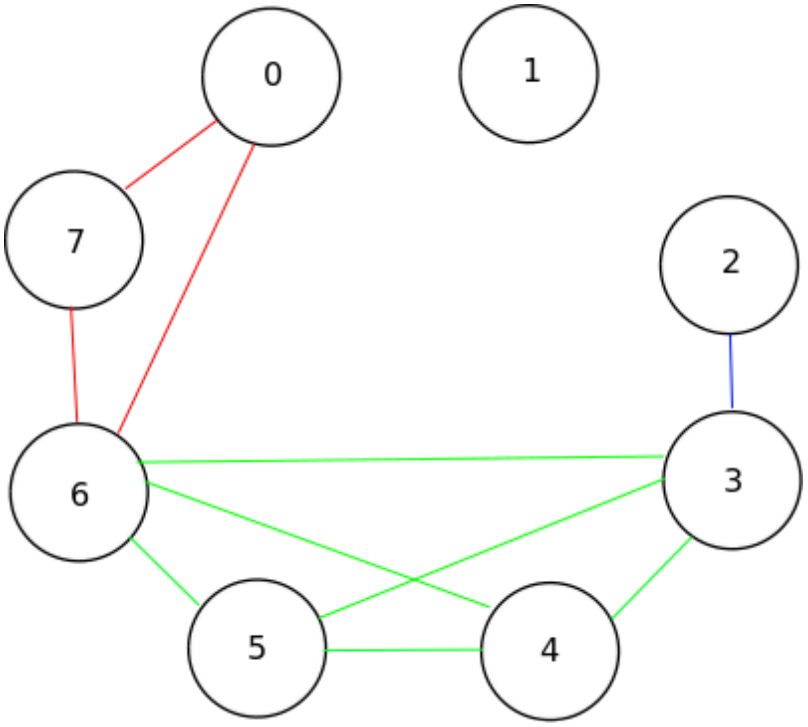
A saída do programa consiste em n linhas (enumeradas de **0** a n**-1**), onde a linha i-ésima contém de forma ordenada a lista de identificadores de pessoas as quais a pessoa de

identificador i não possui nenhuma conexão, seguido de um hífen e a maior distâncias (tamanho do menor caminho) entre a pessoa identificada por i e todas as outras com quem ela tem uma conexão.

Se todas as pessoas estiverem conectadas, a lista antes do hífen estará vazia e não conterá nenhum espaço (por exemplo:- 3).

```
1 - 3
0 2 3 4 5 6 7 - 0
1 - 3
1 - 2
1 - 2
1 - 2
1 - 2
1 - 3
```

Abaixo temos uma imagem para ilustrar o relacionamento entre os nós. Na imagem arestas da mesma cor representam relacionamento que surgiu de um mesmo grupo.



Detalhes de implementação

Para encontrar a distância de um nó aos demais será obrigatório a implementação da busca em largura como mostrado em aula.

Critérios específicos

- Para as turmas E e F, este laboratório tem peso 5.
- Para as turmas G e H, este laboratório tem peso 2.
- Deverão ser submetidos os seguintes arquivos:
    - grafo.c (implementação do grafo)
    - grafo.h (interface do grafo)
    - lab15.c (programa principal cuja solução utiliza o grafo)
- Tempo máximo de execução: 2 segundos.

Testando

Para compilar com o Makefile fornecido e testar um caso de teste:

```
make
./lab15 < arq01.in > arq01.out
diff arq01.res arq01.out
```

onde `arq01.in` é a entrada (casos de testes disponíveis no SuSy), `arq01.out` é a saída do seu programa e `arq01.res` é a solução provida para a entrada arq01.in. O Makefile também contém uma regra para testar todos os testes de uma vez; nesse caso, basta digitar:

```
make testar_tudo
```

Observações gerais

No SuSy, haverá 3 tipos de tarefas com siglas diferentes para cada laboratório de programação. Todas possuirão os mesmos casos de teste. As siglas são:

1. **SANDBOX:** Esta tarefa serve para testar o programa no SuSy antes de submeter a versão final. Nessa tarefa, tanto o prazo quanto o número de submissões são ilimitados, porém arquivos submetidos aqui **não serão corrigidos**.
2. **ENTREGA:** Esta tarefa tem limite de **uma única submissão** e serve para entregar a versão final dentro do prazo estabelecido para o laboratório. Não use essa tarefa para testar o seu programa: submeta aqui quando não for mais fazer alterações no seu programa.
3. **FORAPRAZO:** Esta tarefa tem limite de **uma única submissão** e serve para entregar a versão final após o prazo estabelecido para o laboratório, mas com nota

reduzida (conforme a ementa). O envio nesta tarefa irá substituir a nota obtida na tarefa ENTREGA apenas se o aluno tiver realizado as correções sugeridas no feedback ou caso não tenha enviado anteriormente em ENTREGA.

Observações sobre SuSy:

- Versão do GCC: C-ANSI 4.8.2 20140120 (Red Hat 4.8.2-15).
- Flags de compilação:
  `-ansi -Wall -pedantic-errors -Werror -g -lm`
- Utilize comentários do tipo `/* comentário */`;
  comentários do tipo `//` serão tratados como erros pelo SuSy.

Além das observações acima, esse laboratório será avaliado pelos critérios gerais:

- Indentação de código e outras boas práticas, tais como:
  - uso de comentários (apenas quando forem relevantes);
  - código simples e fácil de entender;
  - sem duplicidade (partes que fazem a mesma coisa).
- Organização do código:
  - tipos de dados criados pelo usuário e funções bem definidas e tão independentes quanto possível.
- Corretude do programa:
  - programa correto e implementado conforme solicitado no enunciado;
  - inicialização de variáveis sempre que for necessário;
  - dentre outros critérios.