Lab: Ordenação Topológica

Descrição

Dado um dígrafo não-ponderado, encontre uma ordenação topológica se o grafo for um DAG. Considere a visitação por ordem crescente de índice. O aluno deverá implementar um programa que leia a ordem do grafo, uma sequência de arestas e imprima a ordenação topológica, se dag, ou imprima nada se houver ciclo.

Exemplos:

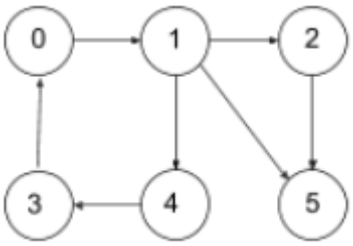

Figure 1: Grafo1

Entrada

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

Saída

Entrada

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

Saída

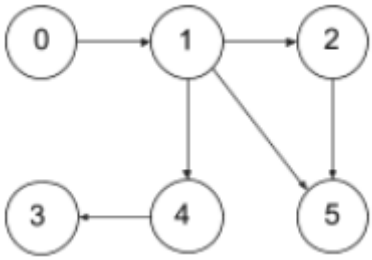

Figure 2: Grafo2

```
0 1 4 3 2 5
```

Importante: Você deverá submeter ao run.codes um arquivo .zip com os arquivos .c e .h do template alterados.

Informativo sobre as avaliações no geral

Para a avaliação final de cada exercício, serão considerados diversos aspectos do código, como a formatação, a lógica e o uso correto das estruturas de dados. Já o Run.Codes servirá apenas de apoio para automatização de alguns testes mais simples.

Importante: O Run.Codes ficará aberto para submissão por apenas 1 ou 2 dias, forçando com que os testes e depurações do código sejam realizados antes da submissão à plataforma. Veja o Exercício 0 (Hello World), para mais detalhes.

Importante: Apenas o último código-fonte enviado na plataforma, após o término do prazo de submissão, será considerado para avaliação.

O trabalho que não cumprir qualquer item da seção 'Requisitos exigidos' receberá **nota zero**, mesmo que a funcionalidade principal esteja implementada corretamente.

Fique atento aos prazos de submissão e envie antes do horário máximo permitido. A plataforma pode sobrecarregar, fazendo com que a submissão seja impedida.

Alguns dos itens avaliados

A seguir é listado alguns dos itens avaliados que reduzem a nota:

- Compilando o código com as flags `-Wall -Wextra -std=c11 -pedantic` nenhuma mensagem de `warning` deve ser retornada
- Nome de variáveis não condizentes com seu uso
- Boas práticas de programação e formatação
- *Memory leak*

Requisitos exigidos

1. Código-fonte com encoding UTF-8 sem BOM;
2. Código-fonte compilando com sucesso;
3. Indentação;
4. Seguir estritamente o solicitado, incluindo as entradas e saídas para os casos de teste;
5. Código-fonte no padrão C11 da linguagem C.

Exemplo de item (4): enunciado informa que o exercício deve ser resolvido usando laços, porém o aluno implementa a solução com recursão.