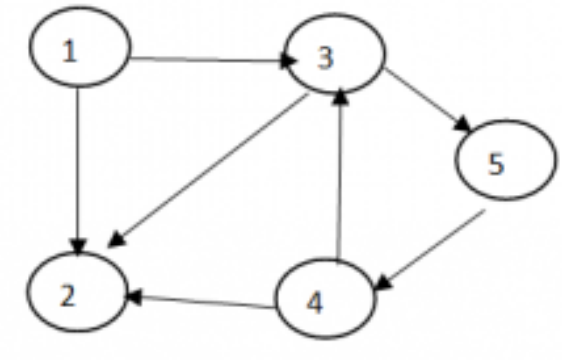
ACH2024 ALGORITMOS E ESTRUTURAS DE DADOS II

Semestre 2021-1 - Exercício prático 1 – caminhos simples em grafos – t.02

Estagiário PAE: Wesley Ramos dos Santos (wesley.ramos.santos@usp.br)

Descrição do EP: A partir do modelo disponibilizado no e-disciplinas (*ep1-modelo.txt*), implementar as seguintes operações para computar caminhos simples em listas de adjacências.

Um caminho simples entre dois vértices **x**,**y** é uma sequência de arestas adjacentes que inicia em **x** e termina em **y** sem repetir vértices. Por exemplo, dados os vértices origem **x**=1 e destino **y**=2, o caminho [1,3,2] é do tipo simples, enquanto que o caminho [1,3,5,4,3,2] não é (porque passa duas vezes pelo 3).



1. O objetivo do trabalho é implementar de forma correta e completa a função **caminhos_max_d** utilizando uma estrutura de grafo em listas de adjacências (ver *typedef* do código exemplo fornecido), e demais funções auxiliares para realização da operação solicitada. Não modifique as definições de *typedef* do código, ou seu programa será invalidado.
2. A função recebe como entrada um grafo **g** de **n** vértices numerados de 1 a **n** (ou seja, desprezando-se a posição zero do vetor) representado por listas de adjacências, dois vértices **x**,**y** e uma distância máxima **d**, retorna uma lista ligada sem cabeça contendo todos os caminhos simples existentes entre **x**,**y** com até **d** arestas de distância. O grafo pode ser tanto do tipo dirigido (como no exemplo acima) como não-dirigido.

   NO* caminhos_max_d(VERTICE* g, int n, int x, int y, int d)

Exemplo. No grafo ilustrado acima, supondo-se **x**=1, **y**=2 e **d**=3, deve ser retornada uma lista contendo todos os caminhos simples de 1 até 2 com distância máxima de 3 arestas. Isso significa retornar a concatenação das listas [1,2] e [1,3,2], que são os únicos caminhos simples possível dentro do limite d=3. O outro caminho simples possível para este par de vértices, que seria [1,3,5,4,2], não é incluído na resposta porque possui tamanho superior ao limite **d** estipulado.

3. O retorno da função deve ser NULL caso não haja nenhum caminho válido entre o par de vértices fornecido, ou deve representar uma lista ligada **sem cabeça** contendo todos os caminhos válidos encontrados em qualquer ordem, em uma sequência única (ou seja, um caminho após o outro). Observe que cada um dos caminhos individuais dentro da lista deve obrigatoriamente começar em **x** e terminar em **y**, pois sem esta informação não será possível distinguir um caminho do próximo.
4. No exemplo acima, dado **x**=1, y=2 e **d**=3, a resposta seria uma única lista [1,2,1,3,2] que é a concatenação dos dois caminhos encontrados (o primeiro em amarelo). Outra resposta igualmente válida para este exemplo seria [1,3,2,1,2], pois não importa qual caminho aparece primeiro dentro da lista - desde que os vértices dentro de cada caminho estejam na sequência correta.

5. RESTRIÇÕES DE IMPLEMENTAÇÃO:
    a. Não **use nenhum vetor** na sua implementação além do vetor g fornecido. Se necessitar de estruturas auxiliares, use sempre listas ligadas de implementação dinâmica.
    b. Não use variáveis globais. A função implementada deve definir localmente todas as variáveis e estruturas auxiliares, ou chamar funções auxiliares que o façam também em um escopo local.
6. Não exiba nenhuma mensagem na tela, nem solicite que o usuário pressione nenhuma tecla etc. Apenas implemente a função solicitada.
7. A função *main*() serve apenas para seus testes particulares, e não precisa ser entregue. Caso você prefira mantê-la no corpo do programa, pede-se apenas que *main*() seja a última função do programa, ou seja, que não haja nenhum código abaixo dela.
8. Seu programa será corrigido de forma *automática*, e por isso você não pode alterar as assinaturas da função solicitada, nem os tipos de dados ou especificações (*typedef*) do modelo fornecido.
9. O EP deve ser desenvolvido individualmente. Não tente emprestar sua implementação para outros colegas, em copiar deles, pois isso invalida o trabalho de **todos** os envolvidos.
10. 0 programa deve ser compilável no ambiente Windows com Codeblocks 13.12 ou superior. Será aplicado um desconto de até 30% na nota do EP caso ele não seja imediatamente compilável nesta configuração.

O que/como entregar:

- A entrega será via upload no sistema e-disciplinas antes da data estipulada.
- Entregue apenas um arquivo texto com o código da função principal e funções auxiliares que ela invoca.
- A extensão do arquivo deve ser .cpp - **favor não compactar**.
- Preencha as funções nroUSP do código exemplo disponível para que você seja identificado.
- Na primeira linha do código, escreva um comentário "//" com seu nome.

Prazos etc.:

O EP deve ser depositado no prazo definido na atividade cadastrada no sistema. Não serão aceitos EPs entregues depois do prazo, independentemente do motivo. Entregas no último dia são assim por conta e risco do aluno, e nenhum tipo de imprevisto de última hora (e.g., problemas de saúde, indisponibilidade de rede etc.) pode ser usado como justificativa para o atraso. O EP é uma atividade para ser desenvolvida ao longo de várias semanas, não nos últimos dias antes da entrega.

É responsabilidade do aluno que fez o *upload* do arquivo verificar se o mesmo foi corretamente recebido pelo sistema. Atrasos/falhas na submissão invalidam o trabalho realizado. Após o *upload*, verifique se você consegue abrir o arquivo depositado, e certifique-se de que é a versão correta do programa e que não está corrompido.

Critérios de avaliação:

A função será testada com uma série de chamadas repetidas e consecutivas, com diversas configurações de grafos e parâmetros de entrada. É assim importante assegurar que o seu programa funciona desta forma (por exemplo, chamando-o dentro de um laço *for*), e não apenas para um teste individual. Um teste é considerado correto se o resultado da soma for exatamente como o esperado, ou incorreto em caso contrário. Erros de alocação de memória ou compilação invalidam o teste, assim como a ausência de funções auxiliares necessárias para a execução do programa.

Este EP deve ser desenvolvido obrigatoriamente por *todos* os alunos de AED2. Sua nota é parte integrante da 1ª. avaliação e *não é* passível de substituição.