Laboratório 10: Percurso em Grafos

Entrega até domingo, 2/6, às 23:59h

Em sala, iniciamos a discussão sobre grafos e discutimos os percursos de busca em profundidade e busca em largura. Nesse laboratório, iremos implementar os dois percursos utilizando a representação por listas encadeadas. Os arquivos **grafo1.dat** e **grafo2.dat** contêm grafos no formato discutido em aula, ambos grafos **conectados** e **não orientados**. A partir do teste implementado, faça as seguintes questões:

1. (1.0) Termine a implementação da função ***grafoLe*** (de acordo com o material da última aula). Rode o teste inicial e tente visualizar os grafos que foram lidos.

2. (3.0) Implemente o método ***grafoPercorreProfundidadeRec***, que visita todos os nós de um grafo utilizando a ideia de percurso em profundidade, de maneira recursiva, a partir de um nó inicial. **A visita em si deve ser simples, e apenas imprimir o número de cada vértice.**
   Para implementar essa função (e as próximas), lembre-se que você irá precisar de um array auxiliar para marcar os nós já visitados. Uma sugestão (para o método recursivo) consiste em criar o array no método principal, e depois chamar uma função auxiliar que realiza as chamadas recursivas (e recebe/atualiza o array de nós visitados).

3. (3.0) Implemente o método ***grafoPercorreLargura***, que visita todos os nós de um grafo seguindo a ideia de percurso em largura. Nessa implementação, será utilizada uma fila dos nós que devem ser visitados. Como discutido em sala, a fila inicia apenas com o nó origem do percurso, e depois segue retirando os nós dessa fila para visitá-los. Para cada nó, percorre-se a lista de vizinhos, enfileirando aqueles que ainda não foram enfileirados até então. O percurso termina quando não há mais itens na fila. Utilize as funções ***newList***, ***enqueue*** e ***dequeue*** do arquivo esqueleto para implementar a fila de nós a serem visitados. **Imprima o número dos vértices na ordem de visitação.**
   Obs: Nesse método, ao invés de marcar quem já foi visitado, como no percurso em profundidade, devemos marcar quem já foi enfileirado!

4. (3.0) Por fim, implemente a versão do percurso em profundidade sem recursão, definida pelo método ***grafoPercorreProfundidade***. Nesse caso, em vez de fazer chamadas recursivas, você irá colocar os nós a serem visitados em uma pilha. As funções ***newList***, ***push*** e ***pop*** implementam as operações auxiliares para criar e manipular uma pilha. **Imprima o número de cada vértice em sua respectiva ordem de visitação.**
   Obs: Assim como na busca em largura, devemos marcar quem já foi empilhado!

Faça upload do arquivo **grafo.c** no EAD até dia 2 de junho, domingo, às 23:59h. Lembre-se de fazer a entrega mesmo que não tenha chegado ao final do exercício.