Relatório de funcionamento do Algoritmo

Douglas José Tertuliano do Santos – 11513928

Matheus Pires Fernandes - 11518681

Descrição do código:

- Função InicializarArquivo()

Esta função apenas abre os arquivos .txt e faz a leitura deles. Os dados desses arquivos são colocados no grafo no meio do escopo da função *main.*

- Funções PintarNode e PintarEdge()

Apenas colorem os nós e a arestas com a cor desejada e passada como parâmetro.

- Função Busca()

Esta função recebe um grafo como argumento e marca todos os vértices como “Não visitados”, depois colore-os de branco. Esta função também percorre o grafo e verifica se cada vértice já foi visitado. Caso não tenham sido visitados, é chamada a função BuscaProf().

Dentro dessa função também se faz a verificação do maior componente. Se o grafo gerado for maior que o maior já encontrado, então o Grafo Maior recebe o grafo de aux.

- Função BuscaProf()

Esta função recebe o vértice analisado no momento como argumento e visita todos os adjacentes do mesmo. Este vértice é adicionado na pilha de nós. Então é verificado se este tem adjacentes.

Para cada nó adjacente a este, é verificado se ele possui o atributo “NãoVisitado”, pois caso já tenha sido analisado seus adjacentes, não necessita analisa-lo. Caso ele realmente não tenha sido visitado, marca a aresta (a aresta entre o nó passado como argumento e o nó adjacente em questão) como visitada.

Após isso, é chamado a BuscaProf() para o vértice adjacente ao vértice inicialmente analisado.

Quando já tiver sido analisado todos os vértices adjacentes ao vértice inicialmente analisado, ele é desempilhado e também marcado com a cor Preta, para mostrar que já foi completamente analisado.

- Função biggerComponent()

Esta função recebe por parâmetro o grafo, o nó de origem, o nó de destino (que é um adjacente ao nó de origem) e um ArrayList com o nome dos nós que já foram inseridos no novo grafo. Cria duas variáveis booleanas, *origin e destiny*, inicializadas como *false*, para ajudar a conferir se precisa repetir a origem e destino. Depois tem um *foreach* para percorrer os nós repetidos e se o nó de origem tiver sido repetido a variável para isso recebe *true*; a mesma coisa é feita para a variável booleana para o nó de destino. Se o nó de origem ainda não tiver sido repetido, ele adiciona ao grafo da função e depois adiciona à lista de nós repetidos; a mesma coisa é feita para o nó de destino.

- Função BuscaL()

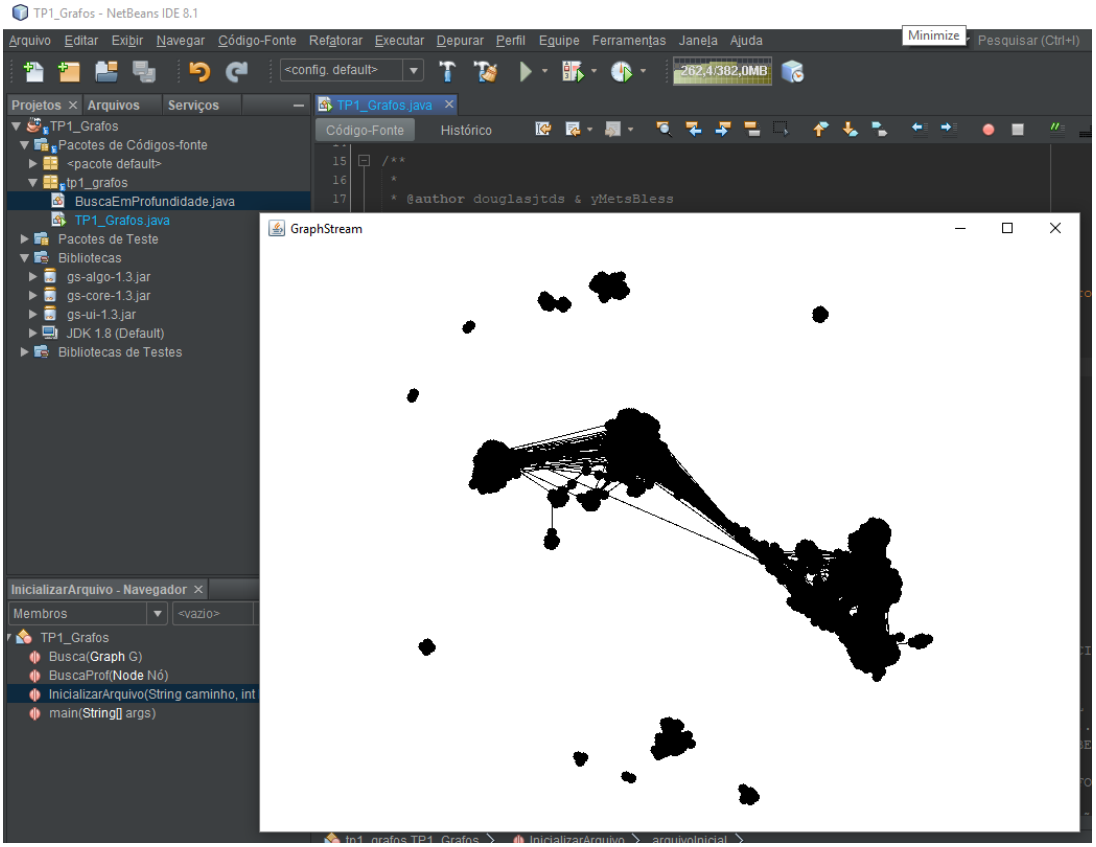
Recebe um grafo como parâmetro e marca todos os vértices como “NãoVisitado”. Pinta os nós não visitados de vermelho. Instancia uma fila de nós e pra cada nó n no grafo ela chama a função busca largura SE o grafo ainda tiver algum nó não visitado ou então SE não tiver um atributo “ui.color” setado; ou seja chama a função BuscaLargura() para cada nó não visitado.

- Função BuscaLargura()

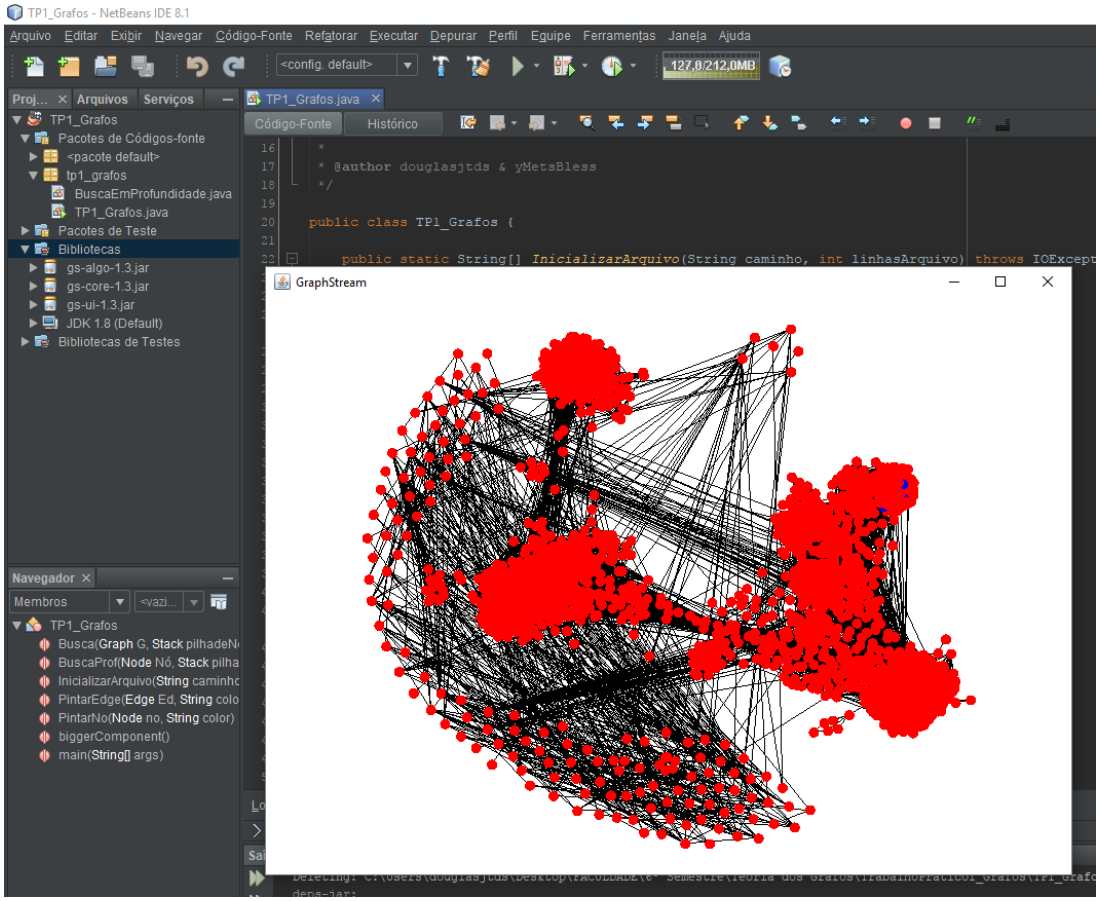
Marca o nó analisado como “ExaminandoAdjacencias” e pinta ele de azul. Adiciona ele na pilha de nós. Após isso entra num *While* que só irá sair quando a pilha de nós estiver vazia. Dentro deste While excluímos da fila o nó que já foi visitado e pintamos ele de preto. Se ainda tempos um nó adjacente a ele, o contador recebe o atributo “comprimento” e um acréscimo, senão, continua como 0. Depois temos mais um while para percorrer todos os nós adjacentes e se o nó ainda tem atributo de “não visitado”, o algoritmo visita, seta, como “ExaminandoAdjacencia”, pinta de azul e pinta a aresta entre eles de amarela.

Imagens dos grafos gerados:

- Grafo inicial:



- **Somente o maior componente**



O código completo pode ser baixado em:
<https://github.com/douglasjtds/TrabalhoPratico1_Grafos>