INF/UFG 2025-1 — AED2 — Prof. André L. Moura — Projeto Final

Tema: Sistema de navegação primitivo

Valor: (10,0 pontos)

O produto final é um sistema com em interface visual, que, dados dois pontos *s* e *t*, encontra um caminho mais curto de *s* para *t*, conforme o algoritmo de Dijkstra. O código pode ser escrito em linguagens como Java, JavaScript, Python, C etc.

Seguem adiante os requisitos funcionais e não funcionais do Sistema de Navegação Primitivo, com a respectiva pontuação.

1. Requisitos Funcionais (RF)

Descrição	Pontuação
RF01 – Permitir importar mapas reais (de arquivos .txt ou xlml etc., contendo coordenadas de vértices e arestas ref. a ruas, avenidas, rodovias etc.) e converter para grafos.	1,00
RF02 – Disponibilizar opção de enumerar os vértices e rotular as arestas com pesos (distâncias).	0,25
RF03 – Permitir ao usuário fazer/desfazer seleção de um vértice de origem e um de destino, com marcação, em cores distintas, para origem e destino.	0,25
RF04 – Calcular e exibir, em cor diferenciada, rota indicativa do menor caminho entre dois vértices.	2,00
RF05 – Permitir a criação e edição de grafos, adicionando/removendo vértices e arestas, com auxílio de clique do mouse.	1,50
RF06 – Suporte a diferentes tipos de grafos (ponderado, não direcionado/direcionado). Ou seja, tratar vias com mão única e mão dupla.	1,75
RF07 – Exibir estatísticas sobre a execução do algoritmo (tempo de processamento, número de nós explorados, custo total).	1,00
RF08 – Permitir copiar para a área de transferência a imagem do grafo em qualquer momento.	0,25
Total de pontos	**8,00**

2. Requisitos Não Funcionais (RNF)

Descrição	Pontuação
RNF01 – Os vértices e as arestas do grafo devem ter cores distintas.	0,25
RNF02 – Diferenciar aresta representativa de via de mão única da aresta de via de mão dupla.	0,25
RNF03 – A execução do programa deve ser otimizada para grandes grafos (milhares de vértices e arestas).	0,25
RNF04 – O tempo de resposta para cálculos deve ser inferior a 2 segundos para grafos médios (~500 nós).	0,25
RNF05 – Uso eficiente de memória para evitar sobrecarga em grafos extensos.	0,25
RNF06 – Interface intuitiva e de fácil manipulação para usuários não técnicos.	0,25
RNF07 – Suporte aos sistemas operacionais Windows ou Linux.	0,25
RNF08 – Código modular e bem documentado para facilitar a manutenção.	0,25
Total de pontos	**2,00**

3. Estruturas de Dados Necessárias

3.1. Representação do Grafo

O grafo pode ser representado de diferentes maneiras, dependendo do tamanho e da densidade:

- Matriz de Adjacência ($O(V^2)$ de espaço) → Melhor para grafos densos.
- Lista de Adjacência ($O(V + E)$ de espaço) → Melhor para grafos esparsos (mais eficiente para Dijkstra).

3.2. Fila de Prioridade (Heap Mínima)

O algoritmo de Dijkstra frequentemente utiliza uma fila de prioridade para processar os vértices na ordem das menores distâncias conhecidas.

- A estrutura ideal para isso é o Heap Mínima.
- A cada iteração, o vértice com menor distância acumulada é extraído da fila.

Isso garante O(lg V) para inserção e remoção.

3.3. Estrutura para Armazenamento das Distâncias (Vetor de Distâncias)

Um vetor ou array que armazena a menor distância conhecida de cada nó para o nó de origem. Inicialmente, todas as distâncias são definidas como infinito, exceto a distância do nó de origem, que é 0.

3.4. Vetor de Predecessores

Um vetor que armazena o nó anterior no caminho mais curto para cada nó, o que permite reconstruir o caminho quando o algoritmo termina.

3.5. Conjunto de Nós Visitados

Um conjunto ou vetor para marcar os nós que já foram processados (ou visitados) e cujas distâncias finais já foram determinadas. Isso impede que o algoritmo processe um nó mais de uma vez.

4. Artefatos do Projeto

a) Documentação do Projeto (conforme formalismos da Engenharia de Software).

b) Códigos-fontes.

c) Arquivo “Read-me”, com instruções de instalação e execução.

d) Instalador do programa.

5. Equipe de trabalho
Grupo de 4 (quatro) ou 5 (cinco) membros.

6. Forma de entrega
Compactar todos os artefatos do projeto e atribuir nome ao arquivo **.zip** no seguinte formato: PF_<primeiroNomeAluno><ultimoNomeAluno>.zip.
Exemplo: PF_JoaoNaves.zip

Postar o arquivo .ZIP no SIGAA.

7. Cronograma

7.1. Data de entrega
18/06/2025

7.2. Datas de apresentação
24/06/2025 (curso de Sistemas de Informação)
25/06/2025 (curso de Engenharia de Software)

Importante:
1. Não serão aceitas entregas por e-mail ou fora da data de entrega.
2. Na documentação do projeto, deve-se constar, em ordem alfabética, o nome de cada integrante do grupo.

Anexo I — Exemplo de interface gráfica