COMS 3110 - Homework 4

Name: Erroll Barker.

[redacted]

Assignment: Homework 4

1. Adjacency List Data Structure

I used the following Java type to represent the adjacency list in my graph class:

Map<T, List<Pair<T, Integer>>>

This maps each vertex T to a list of pairs, where each Pair contains a neighboring vertex and the weight of the directed edge to that neighbor. This structure supports efficient access to neighbors and allows Dijkstra's algorithm to process the weighted graph efficiently.

2. Edge Weight Assignment

When converting an image to a graph, each pixel is treated as a node, and directed edges are added to its four immediate neighbors (up, down, left, right). The weight of each edge is based on the brightness of the destination pixel:

- If the destination pixel is white (0xFFFFFFFF), the weight is set to 1.
- If the destination pixel is black or non-white, the weight is set to 1000.

This design ensures that Dijkstra's algorithm naturally prefers paths through white space, helping it identify horizontal and vertical whitespace between lines or characters in the image.

3. Use of Dijkstra’s Algorithm

To detect horizontal and vertical whitespace lines, I constructed a single instance of WeightedAdjacencyList and used Dijkstra’s algorithm a constant number of times — once per row and once per column. For each row, I computed the shortest path from the leftmost to the rightmost pixel. If the path cost was below a computed threshold (such as the 20th percentile of all row costs), the row was considered whitespace. The same strategy was applied to columns, using top-to-bottom paths. This limits the number of Dijkstra calls to a linear number relative to image height and width and complies with the assignment's constant-call constraint.

4. Heap Implementation Used

For the priority queue in Dijkstra’s algorithm, I used the standard PriorityQueue class from the java.util package. It supports efficient insertions and retrieval of the minimum element, which aligns with the requirements for Dijkstra’s algorithm. Because it is part of the Java Standard Library, no third-party license or attribution is required.