Peter M Piper

An interactive and demonstrative program that implements directed, undirected, weighted, and unweighted graphs and provides functions including BFS, DFS and Prims to analyze properties of the graphs.

Data Structures 2

Graph

**Program Description and Design**

This program implements a visual interface for exploring different properties of unweighted, weighted, undirected and directed graphs. Users can add or update vertices and directed edges with weights. Graphs can be saved to a file and reopened with interactive dialogues. The adjacency matrix for the graph is displayed and updated as the graph is modified. The graph can be toggled between weighted, unweighted, directed and undirected. A GUI is provided to display the graph and results of analysis functions. These include breadth first search, depth first search, and Prim’s algorithm to calculate a minimum spanning tree. Additionally functions are provided to determine whether the graph is connected and to calculate shortest paths.

**Test Graphs**

* Undirected and connected
* Undirected and disconnected
* Directed with cycles
* Directed acyclic and weighted
* Directed strongly connected unweighted
* Directed weakly connected unweighted
* Prim’s algorithm graph

**Working Features**

* Add/Update vertex
* Delete vertex
* Add/Update edge
* Delete edge
* New graph
* Open graph
* Save graph
* SaveAs graph
* Quit
* Adjacency matrix display
* Adjacency matrix navigation buttons
* Breadth first search (directed)
* Depth first search (directed)
* Toggle weighted/unweighted
* Toggle directed/undirected
* Display vertices with labels
* Display edges with arrows and weights

**Known Bugs**

* Minimum spanning tree not displaying edges correctly
* Weights and edges are displayed in duplicate for undirected graphs
* IsConnected? function produces incorrect results based on MST
* Problems with screen refreshing
* Cancel button on save dialogue resets graph
* Disable functions when not available

**Not Implemented**

* Shortest path
* Is Acyclic?
* Find topological order
* Strongly/Weakly connected

**Future Improvements**

* Animate BFS, DFS path traversals
* Application to demonstrate practical use of graph
* Drag and drop interaction with vertices and edges

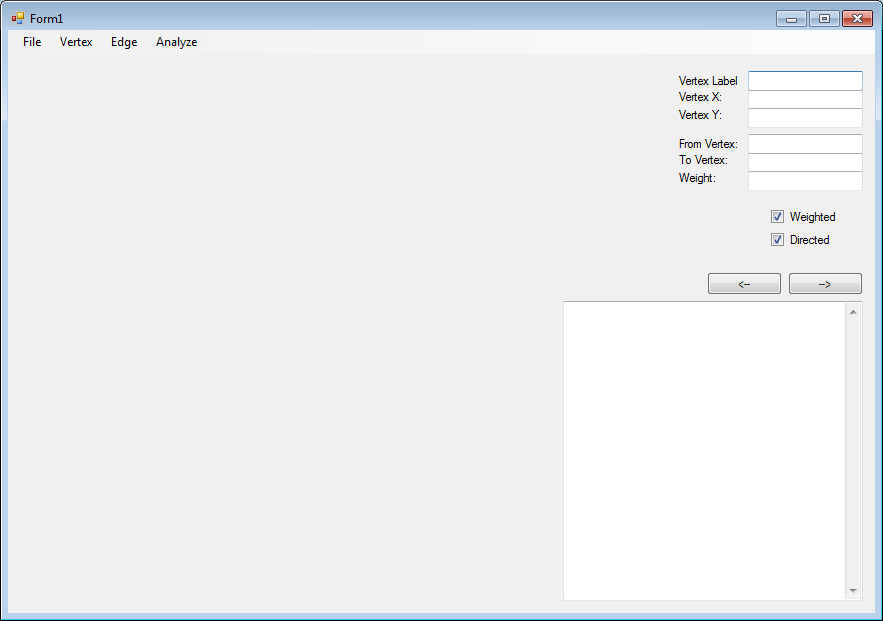
**Test Plan**

**Reason for Test Case \_\_\_\_\_\_\_\_ Input Values\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_Expected Output**

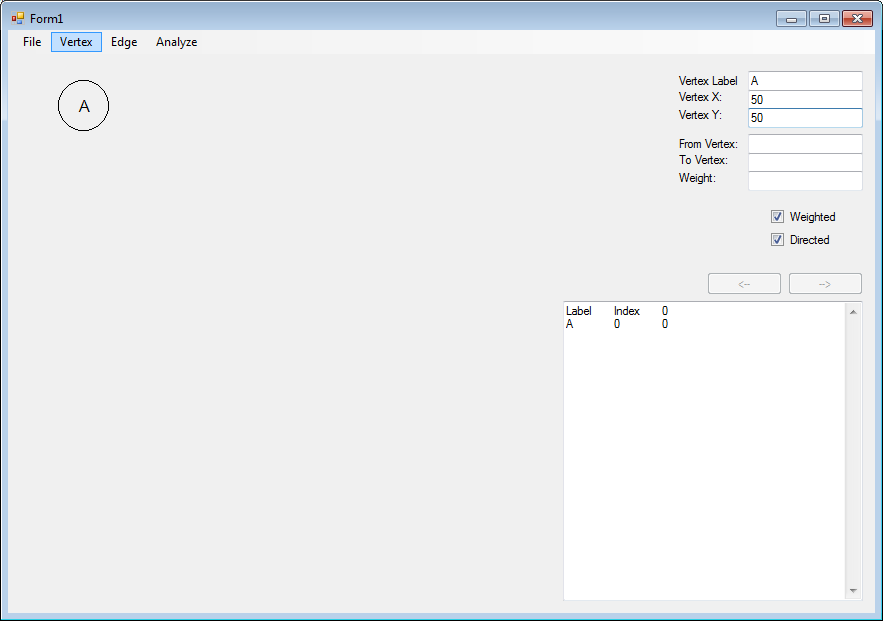
|  |  |  |
| --- | --- | --- |
| Add vertex | Label: A, x: 50, y: 50 | Vertex is displayed,  Adjacency matrix:  Label Index 0  A 0 0 |
| Add vertex – duplicate | Label: A, x: 50, y: 50 | Single Vertex is displayed,  Adjacency matrix:  Label Index 0  A 0 0 |
| Update vertex | Label: A, x: 150, y: 150 | Vertex is moved to new position,  Adjacency matrix unchanged |
| Add second vertex | Label: B, x: 100, y: 50 | Second vertex displayed,  Adjacency matrix:  Label Index 0 1  A 0 0 0  B 1 0 0 |
| Add edge - directed | From: A, To: B, Weight: 3 | Edge from A to B displayed with arrow and weight,  Adjacency matrix:  Label Index 0 1  A 0 0 3  B 1 0 0 |
| Add edge – invalid vertex | From: A, To: C, Weight: 3 | Message box shown with error: Vertex Not Found! |
| Add edge – undirected | From: A, To: B, Weight: 3 | Edge from A to B displayed with weight,  Adjacency matrix:  Label Index 0 1  A 0 0 3  B 1 3 0 |
| Add edge – unweighted, directed | From: A, To: B, Weight: 3 | Edge from A to B displayed with arrow and without weight,  Adjacency matrix:  Label Index 0 1  A 0 0 1  B 1 0 0 |
| Add edge – unweighted, undirected | From: A, To: B, Weight: 3 | Edge from A to B displayed without arrow or weight,  Adjacency matrix:  Label Index 0 1  A 0 0 1  B 1 1 0 |
| Delete vertex – with edge | Label: A | Vertex A is removed, edge from A to B is removed, B is displayed,  Adjacency matrix:  Label Index 0  B 0 0 |
| Save As… | Enter filename in dialogue box | File is created |
| Open… | Select saved file | Graph is displayed |
| Save | None | Graph is saved in same file |
| New | None | Graph is cleared, adjacency matrix is cleared |
| Quit | None | Program exits |
| IsConnected? | None | Message box appears with connection status |
| Breadth-first search | Open acyclic directed graph, From: D, To: B | Vertices and edges display path in red from D to B |
| Depth-first search | Open acyclic directed graph,  From: D, To: B | Vertices and edges display path in red from D to C to A to B |
| Breadth-first search #2 | Open directed cyclic graph, From: A, To: D | Vertices and edges display path in red from A to B to C to D |
| Depth-first search #2 | Open directed cyclic graph, From: A, To: D | Vertices and edges display path in red from A to C to D |

**Sample Outputs**

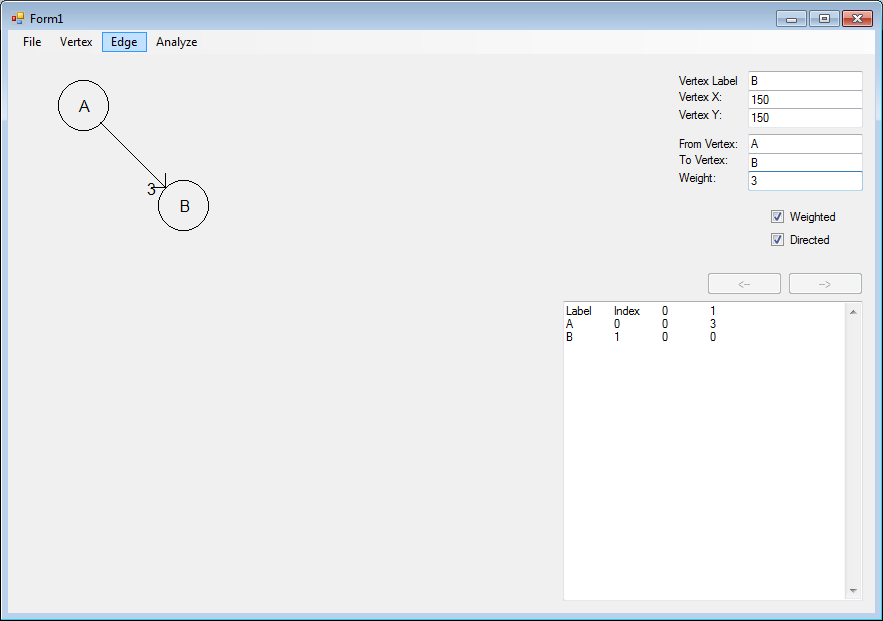
New Graph



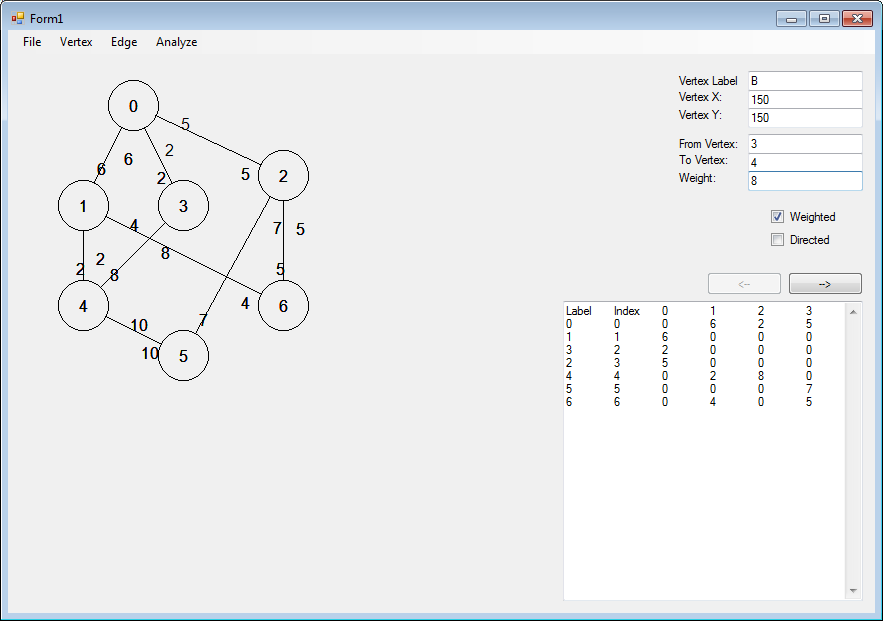
Add Vertex



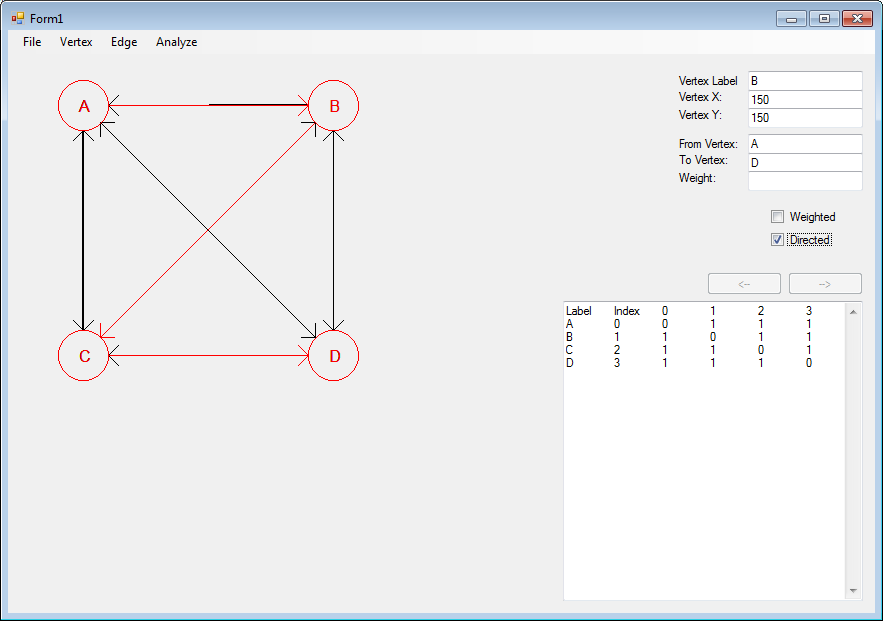
Add Second Vertex with Directed Edge



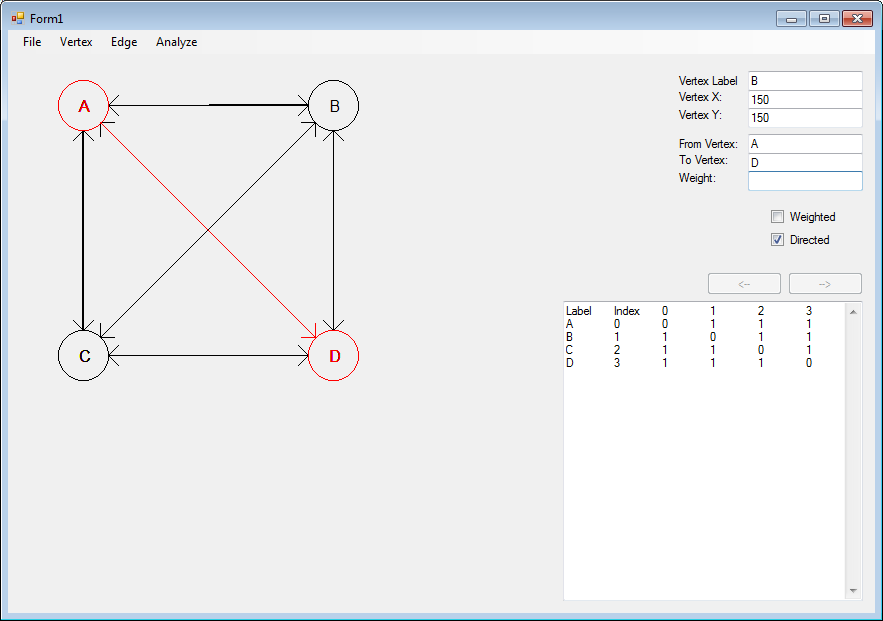
Prim’s Algorithm Graph



Breadth-First Search



Depth-First Search



Minimum Spanning Tree

