## Hand in instructions:

* For Part A, hand in your AdjList project folder.
  + Close all files before closing your solution.
  + Remove all test code from your Program.cs files (don’t delete the file though).
  + Hand in to Brightspace
* For Part B and C, please format your work in a clear and legible manner on the pages below. Place in the root folder of your solution.
* You are required to work alone on this assignment.

## Part A: Coding [26 marks]

Submitted in the separate project file

Problem: In class, we coded the matrix implementation of a graph. Another implementation is an adjacency list. In an adjacency list implementation, edges are stored in a list. One list is created for each vertex in the graph. For example, given the following graph, edges would be stored as follows:



This is an Arraylist of Arraylists.

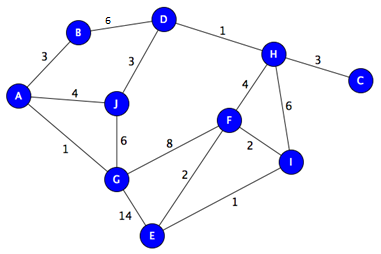
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| A |  | List 🡪 |  | A-D | A-E | A-B |
| B |  | List 🡪 |  | B-E |  |  |
| C |  | List 🡪 |  | C-B |  |  |
| D |  | List 🡪 |  | D-G |  |  |
| E |  | List 🡪 |  | E-F | E-H |  |
| F |  | List 🡪 |  | F-C | F-H |  |
| G |  | List 🡪 |  | G-H |  |  |
| H |  | List 🡪 |  | H-I |  |  |
| I |  | List 🡪 |  | I-F |  |  |

This is the vertex list.

Code the following:

* A class called AAdjList that implements AGraph. Use C# ArrayLists to store the edges. The arraylists can be stored in an Arraylist.
* A class called UGraphAL (undirected graph) that implements AAdjacencyList (not doing directed implementation).

## Part B: Algorithm Traces [15 marks]



1. Using Kruskal’s algorithm, show the first 3 iterations for minimum spanning tree (may be more than one correct solution). Again, show all of your work as demonstrated in class. [5 marks]

## Solution for Kruskal’s algorithm first three iterations:

* We have total 10 nodes (vertices) and 16 edges (links). For minimum spanning tree we have to find “9” optimum edges.
* First we will list out all our edges in ascending Order.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Edge | Weight | | A – G | 1 | | D – H | 1 | | I – E | 1 | | F – E | 2 | | F – I | 2 | | H – C | 3 | | A – B | 3 | | D – J | 3 | | A – J | 4 | | H – F | 4 | | B – D | 6 | | J – G | 6 | | H – I | 6 | | G – F | 8 | | G – E | 14 | | |  |  | | --- | --- | | Forest | Vertices in forest | | 0 | A | | 1 | B | | 2 | C | | 3 | D | | 4 | E | | 5 | F | | 6 | G | | 7 | H | | 8 | I | | 9 | J | |

## First iteration

A picture containing different, several

Description automatically generatedDiagram

Description automatically generated



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Edge | Weight | | A – G | 1 | | D – H | 1 | | I – E | 1 | | F – E | 2 | | F – I | 2 | | H – C | 3 | | A – B | 3 | | D – J | 3 | | A – J | 4 | | H – F | 4 | | B – D | 6 | | J – G | 6 | | H – I | 6 | | G – F | 8 | | G – E | 14 | | |  |  | | --- | --- | | Forest | Vertices in forest | | 0 | A, G | | 1 | B | | 2 | C | | 3 | D | | 4 | E | | 5 | F | | 6 | G | | 7 | H | | 8 | I | | 9 | J | |

## Second iteration

A picture containing different, several

Description automatically generatedDiagram

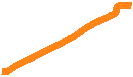
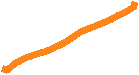
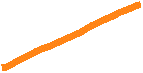
Description automatically generated



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Edge | Weight | | A – G | 1 | | D – H | 1 | | I – E | 1 | | F – E | 2 | | F – I | 2 | | H – C | 3 | | A – B | 3 | | D – J | 3 | | A – J | 4 | | H – F | 4 | | B – D | 6 | | J – G | 6 | | H – I | 6 | | G – F | 8 | | G – E | 14 | | |  |  | | --- | --- | | Forest | Vertices in forest | | 0 | A, G | | 1 | B | | 2 | C | | 3 | D, H | | 4 | E | | 5 | F | | 6 | G | | 7 | H | | 8 | I | | 9 | J | |

## Third iteration

## Diagram Description automatically generated



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Edge | Weight | | A – G | 1 | | D – H | 1 | | I – E | 1 | | F – E | 2 | | F – I | 2 | | H – C | 3 | | A – B | 3 | | D – J | 3 | | A – J | 4 | | H – F | 4 | | B – D | 6 | | J – G | 6 | | H – I | 6 | | G – F | 8 | | G – E | 14 | | |  |  | | --- | --- | | Forest | Vertices in forest | | 0 | A, G | | 1 | B | | 2 | C | | 3 | D, H | | 4 | E, I | | 5 | F | | 6 | G | | 7 | H | | 8 | I | | 9 | J | |

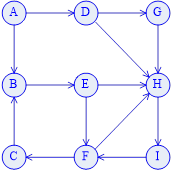
## Part C: Traversals Traces [9 marks]

For the unweighted directed graph represented by the following:

G1 = (V1, E1)

V1 = {A, B, C, D, E, F, G, H, I}

E1 = {(A, B), (A, D), (A, D), (B, E), (C, B), (D, G), (E, F), (E, H), (F, C), (F, H), (G, H), (H, I), (I, F)}



For both questions, show all of your work

1. Show the depth first traversal beginning at G. Assume that neighbors are added to the stack alphabetically. [5 marks]

depth first traversal:

A picture containing schematic

Description automatically generated



First, push G in the stack.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| G |  |  |  |  |  |  |  |  |  |

Pop G, mark it visited(add in the visited list). Enumerate neighbours of G.

Visited:- G

Push neighbours in stack alphabetically (push H).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| H |  |  |  |  |  |  |  |  |  |

Pop H, mark it visited(add in the visited list). Enumerate neighbours of H.

Visited:- G, H

Push neighbours of H in the stack alphabetically (push I).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| I |  |  |  |  |  |  |  |  |  |

Pop I, mark it visited(add in the visited list). Enumerate neighbours of I.

Visited:- G, H, I

Push neighbours of I in the stack alphabetically (push F).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| F |  |  |  |  |  |  |  |  |  |

Pop F, mark it visited(add in the visited list). Enumerate neighbours of F.

Visited:- G, H, I, F

Push neighbours of F in the stack alphabetically which are not visited (push C).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| C |  |  |  |  |  |  |  |  |  |

Pop C, mark it visited(add in the visited list). Enumerate neighbours of C.

Visited:- G, H, I, F, C

Push neighbours of C in the stack alphabetically (push B).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| B |  |  |  |  |  |  |  |  |  |

Pop B, mark it visited(add in the visited list). Enumerate neighbours of B.

Visited:- G, H, I, F, C, B

Push neighbours of B in the stack alphabetically (push E).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| E |  |  |  |  |  |  |  |  |  |

Pop E, mark it visited(add in the visited list). Enumerate neighbours of E.

Visited:- G, H, I, F, C, B, E

All neighbours of E are visited, so we will not going to push anything in stack.

1. Show the breadth first traversal beginning at B. Assume that neighbors are added to the queue alphabetically. [4 marks]

## breadth first search:

## A picture containing schematic Description automatically generated



## 

First, Enqueue B in the queue.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| B |  |  |  |  |  |  |  |  |  |

Dequeue B and mark it visited(add in the visited list).

Visited:- B

Enumerating neighbours of B. Enqueue neighbours alphabetically if it is not visited(Enqueue E).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| E |  |  |  |  |  |  |  |  |  |

Dequeue E and mark it visited(add in the visited list).

Visited:- B, E

Enumerating neighbours of E. Enqueue neighbours alphabetically if it is not visited(Enqueue F and H).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| F | H |  |  |  |  |  |  |  |  |

Dequeue F and mark it visited(add in the visited list).

Visited:- B, E, F

Enumerating neighbours of F. Enqueue neighbours alphabetically if it is not visited(Enqueue C).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| H | C |  |  |  |  |  |  |  |  |

Dequeue H and mark it visited(add in the visited list).

Visited:- B, E, F, H

Enumerating neighbours of H. Enqueue neighbours alphabetically if it is not visited(Enqueue I).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| C | I |  |  |  |  |  |  |  |  |

Dequeue C and mark it visited(add in the visited list).

Visited:- B, E, F, H, C

There are no any unvisited neighbours of C.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| I |  |  |  |  |  |  |  |  |  |

Dequeue I and mark it visited(add in the visited list).

Visited:- B, E, F, H, C, I

There are no any unvisited neighbours of I.

1. Using Dijkstra's algorithm, calculate the shortest path from D to E. Show all of your work as demonstrated in class. [10 marks]

Submitted in the separate excel file

## Shortest Path

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Pass | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |  | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | Initial | | | 1 | | | 2 | | | 3 | | | 4 | | | 5 | | | 6 | | | 7 | | | 8 | | | 9 | | |
| Vertex | d | p | k | d | p | k | d | p | k | d | p | k | d | p | k | d | p | k | d | p | k | d | p | k | d | p | k | d | p | k |
| A |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| B |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| C |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| D |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| E |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| F |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| G |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| H |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| I |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Submitted in the separate excel file