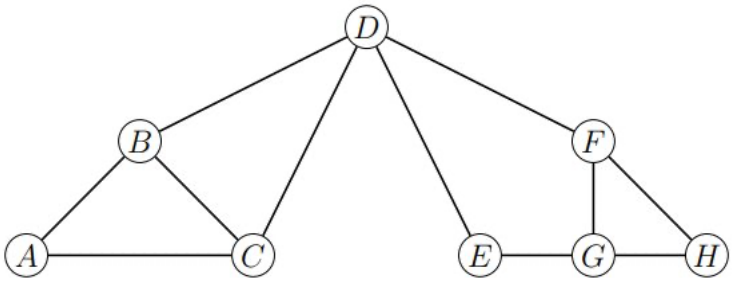
CSCE 3110 Assignment 6

Due: 12/01, 11:59PM

1. (20 points) Given an array A with n elements, please design and analyze an efficient algorithm to count the number of pairs $(A[i], A[j])$, $1 \leqslant i < j \leqslant n$ for which $A[i] + 1 = A[j]$. For example, for the array [2, 4, 3, 5], there are two pairs (2, 3) and (4, 5).
   Solution: Scan the input from left to right and insert each element into a universal hash table. Before inserting $A[j]$, check if $A[j] - 1$ is present in the hash table. This takes $O(1)$ time per element to insert, $O(1)$ expected time to find - overall running time is $O(n)$ expected.

2. (20 points) For the following questions, assume that the graph is represented using adjacency lists, and that **all adjacency lists are sorted**, i.e., the vertices in an adjacency list are always sorted alphabetically.



   (a) give the adjacency lists

   (b) use DFS to find a path from A to H

   (c) Define the edge type while searching with DFS

A: $\{B, C\}$

B: $\{A, C, D\}$

C: $\{A, B, D\}$

D: $\{B, C, E, F\}$

E: $\{D, G\}$

F: $\{D, G, H\}$

G: $\{E, F, H\}$

H: $\{F, G\}$

$$A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow G \rightarrow F \rightarrow H$$

Tree Edges (T):

- A → B, A → C, B → C, B → D, C → D, D → E, E → G, F → H

Back Edges (B):

- C → A, D → B, F → D, G → H

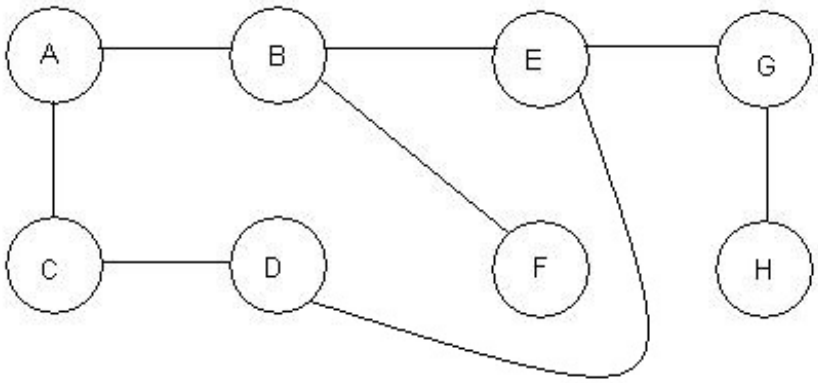
Ignored (Parent Edges):

- B → A, C → B, D → C, E → D, G → F, F → G, H → F

3. (20 points) The graph below is represented as undirected graph with labeled nodes (**A–H**) and edges. Starting from node **A**, show the sequence of traversal using:
    a. **Depth-First Search (DFS):** Visit as deeply as possible before backtracking, while preferring neighbors in **alphabetical order**.

b. **Breadth-First Search (BFS):** Explore all immediate neighbors of the current node before moving deeper, while preferring neighbors in **alphabetical order**.

Include **step-by-step explanations** or **diagrams** for both traversal processes.



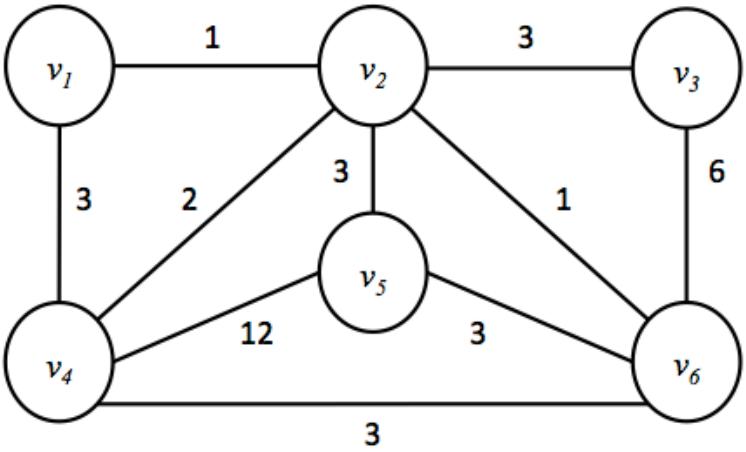
DFS Order:

A → B → E → D → C → G → H → F

BFS Order:

A → B → C → E → F → D → G →H

4. (30 points) assume that the graph is represented using adjacency lists, and all adjacency lists are sorted, i.e., the vertices in the adjacent list are always in ascending order by vertex IDs. Also assume some stable sorting algorithm is applied to sort the edges by weight. Name edges with their starting and ending vertices, for example, the edge from vertex v_1 to vertex v_2 is named (v_1, v_2) and has cost 1.



1) Run Kruskal's algorithm on this graph, showing the action of Kruskal's algorithm, i.e., either accept or reject an edge into MST.

Order	***Edge***	***Weight***	***Action***
1	(v_1, v_2)	1	Accepted
2	(v_2, v_6)	1	Accepted
3	(v_2, v_4)	2	Accepted
4	(v_1, v_4)	3	Rejected
5	(v_2, v_3)	3	Accepted
6	(v_2, v_5)	3	Accepted
7			
8			
9			
10			

2) Select v_4 to start and run Prim's algorithm **without** using heap on this graph. Fill the provided table using the edges with weights in the order selected by Prim's algorithm.

Order	Edge	Weight
1	(v_2, v_4)	2
2	(v_1, v_2)	1
3	(v_2, v_6)	1
4	(v_2, v_3)	3
5	(v_2, v_5)	3
6		
7		

8		
9		
10		

3) What is the cost (weight) of the minimum spanning tree?

Solution:

The cost (weight) of the minimum spanning tree is 1 + 1 + 2 + 3 + 3 = 10.