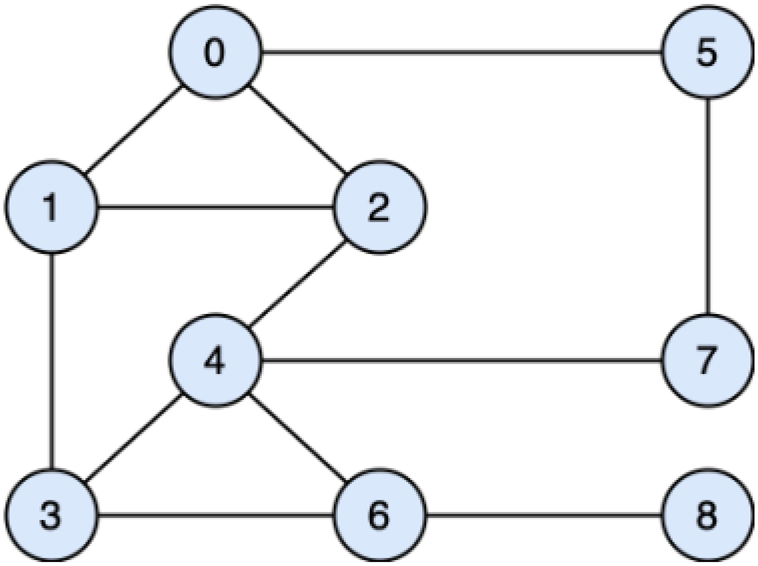
Due Date: 24th Nov 2024
20% penalty for 1 day late
40% penalty for 2 days late
Submission not allowed afterwards

CS2009: Design and Analysis of Algorithms (Fall 2024)
Assignment 4
Total Marks: 100

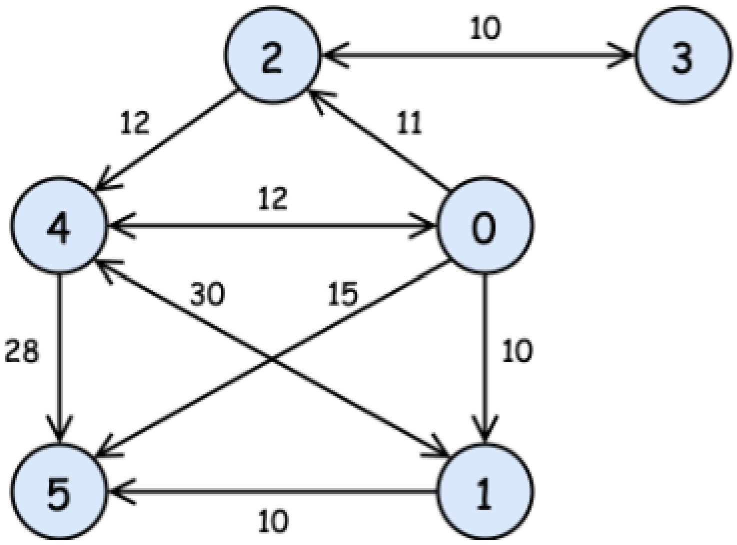
1. Consider the following graph:



a) Write the vertices of the above graph in the order in which they would be visited in a depth first and breadth first traversal starting at node 0. Assume neighbors are visited in numerical order. [14 Points]
b) Suppose the above graph as a tree, how many strong components do we have? [2 Points]
c) Analyze the complexity of BFS algorithm (use Big-Oh notation) from the following pseudocode. [4 Points]

```
mark s as explored, all other vertices as unexplored
D := a queue or stack data structure, initialized with s
while D is not empty do
 remove the vertex from the front/top of D, call it v
 for edge (v, w) in v's neighborhood do
  if w is unexplored then
   mark w as explored
```

2. Find the weighted shortest path from vertex 3 to vertex 5 in the *digraph* (directed graph) below.



a) Create a table below with each vertex and its corresponding outgoing vertices. [2 Points]

b) Start with the following default values in Table 1 below (fill out the answers from part-1 in the "Outgoing" column). How would these values change after exploring vertex 3? Next, Create Table 2 with same columns as Table 1 and provide your response. What is now the unexplored vertex with the smallest distance from vertex 3? [2 Points]

Table 1: Default values

Vertex	Outgoing	Distance from 3	Previous	Explored
0		INFINITY	null	no
1		INFINITY	null	no
2		INFINITY	null	no
3		INFINITY	null	no
4		INFINITY	null	no
5		INFINITY	null	no

c) How would these values change after exploring vertex 2? Create Table 3 with your response. What is now the unexplored vertex with the smallest distance from vertex 3? [2 Points]

d) How would these values change after exploring vertex 4? Create Table 4 with your response. What is now the unexplored vertex with the smallest distance from vertex 3? [2 Points]

e) How would these values change after exploring vertex 0? Create Table 5 with your response. What is now the unexplored vertex with the smallest distance from vertex 3? [2 Points]

f) How would these values change after exploring vertex 11? Create Table 6 with your response. What is now the unexplored vertex with the smallest distance from vertex 3? [2 Points]

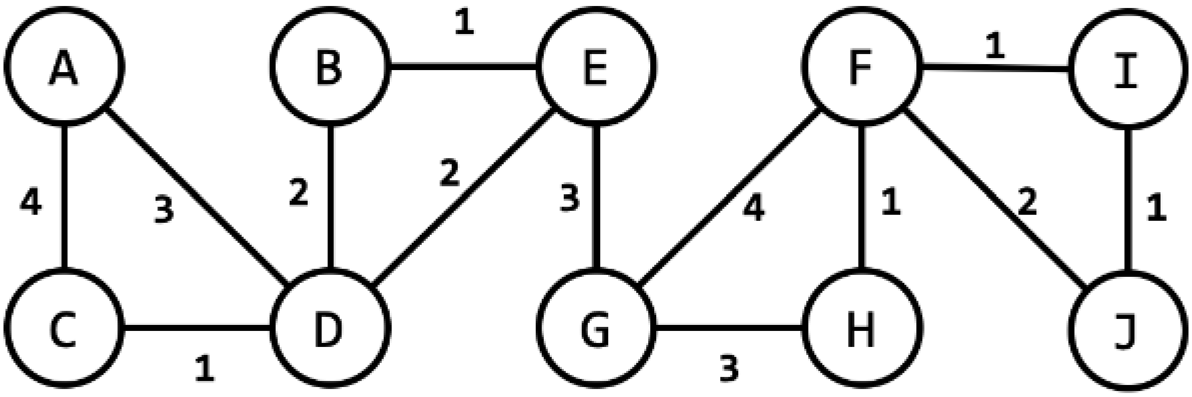
g) How would these values change after exploring vertex 5? Fill out Table 7 with your response. [2 Points]

h) What is the weighted shortest path from vertex 3 to 5? What is the total distance of this path? [2 Points]

i) Analyze the complexity of SPF algorithm (use Big-Oh notation) from the following pseudocode. [4 Points]

```
for each vertex v
   distance[v] = Infinity
   previous[v] = null
   explored[v] = false
distance[s] = 0   // s is the source
repeat N times
   let v be unexplored vertex with smallest distance
   explored[v] = true
   for every u: unexplored neighbor(v)
      d = distance[v] + weight[v,u]
      if d < distance[u]
         distance[u] = d
         previous[u] = v
```

3. Identify the edges on a minimum spanning tree for this graph following using Prim's and Kruskal’s algorithm. [6 Points]

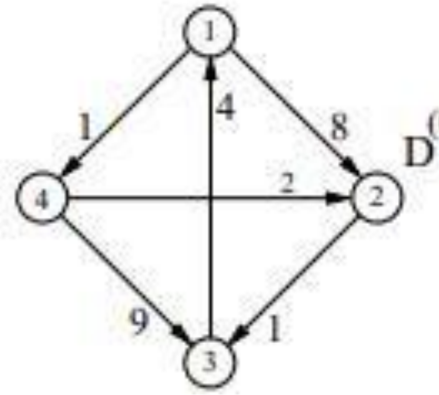


–a) Based on your understanding of Prim's algorithm, how can we efficiently implement the step which involves finding min-weight edge with one endpoint in *T*? [1 Point]

b) Based on your understanding of Kruskal's algorithm, how can we efficiently implement the step which involves finding the next min-weight edge in G? [1 Point]

c) Once the next min-weight edge (v,w) is found, how can we efficiently check if adding it to the MST would create a cycle? [1 Point]

4. Using Floyd-Warshall, find all pairs shortest path following Figure (D^0 weight matrix is also provided). Discuss the its complexity as well [10 Points]



$$D^{(0)} = \begin{bmatrix} 0 & 8 & ? & 1 \\ ? & 0 & 1 & ? \\ 4 & ? & 0 & ? \\ ? & 2 & 9 & 0 \end{bmatrix}$$

? = infinity

5. Word search puzzle problem: Given the following 2d, 4x5 grid of letters

O F O O T
V O Q U O
E O I H O
R T G H F

a) Find the word "foot" in the grid.

b) The word may be formed in any direction - up, down, left or right (not diagonals!) but all of the letters in a word must occur consecutively. Assuming the grid starts in the upper left corner with position (0,0) , and that the row is the first coordinate, “foot” would be found at 3 places in the grid: [(0,1) to (0,4)] ,[(4,4) to (0,4)] , and [(0,1) to (4,2)].

c) Assume you are provided with the above word search puzzle grid. Write a algorithm to efficiently search all occurrences of the word. Display all the coordinates (row and column) of the starting and ending positions of word occurrence. [20 Points]

6. Compare the **Quick Hull** algorithm and **Chan’s Algorithm** for computing the convex hull of a set of points in a 2D plane. You will implement both algorithms and evaluate their **runtime** and **memory usage** on a small-scale dataset of 20 randomly generated points. Using this dataset, you will analyze the practical performance of both algorithms, comparing how efficiently they compute the convex hull in terms of both time and space. You are expected to plot the **runtime** and **memory usage** as the number of points increases and discuss your findings based on these observations. The comparison should include a detailed analysis of the performance of each algorithm, considering factors like the size of the input set and the resulting convex hull. The assignment will require you to implement both algorithms, measure the associated metrics, and provide a report summarizing your results, including graphical representations of the performance comparison. [15 Points]

7. Consider two sets of points:
   a. Set A: (1,1), (4,1), (1,4), (3,3), (2,5)
   b. Set B: (6,6), (9,6), (6,9), (8,8), (7,10)

First, compute the convex hulls of **A** and **B** separately. Then, merge the hulls using the divide-and-conquer approach. Describe how the merging step works geometrically. [6 Points]