Tyler Richard

CPTS 233

Professor Guizani

HW4

1.
    a. Graph: an abstract data type which shows a set of vertices, or nodes, together with a set of paired vertices (edges) in an unordered or ordered fashion.
    b. Vertice: A “node” within a graph.
    c. Edge: A link between two nodes representing a relationship between them. Mathematically that relationship is with reference to a function.
    d. Undirected Graph: A graph with unordered pairs of vertices and undirected edges.
    e. Directed Graph: A graph featuring ordered pairs of vertices and “arrows” for edges, which illustrate the directional nature of the relationship between the two vertices.
    f. Path: A sequence of edges which joins a sequence of distinct vertices.
    g. Loop: An edge that connects a vertice to itself.
    h. Cycle: A closed path from a vertex to itself.
    i. Acyclic: A graph which contains no cycles.
    j. Connected: (with respect to graph) A graph where there is a path from any given point in the graph to any other point in the graph. (with respect to a pair of vertices) A pair of vertices which contain a path between the two.
    k. Sparse: A graph with a number of edges closer to the minimum number possible.
    l. Weight: The “cost” of the path; it’s length, capacity, or the amount of energy it may require.

2. The circumstances in which one would want to use an adjacency matrix as opposed to an adjacency list would be relevant to size. Matrices take up much more memory as they are fairly sparse, so the graph in question would need to be small or have many edges as lists become more inefficient as edges increase.

3. Graphs work best to represent elements which have relationships to other elements within the set. Some examples of what that relationship might be:
    a. Digital connectivity, such as devices in a network
    b. Physical connectivity, like roads between buildings
    c. Actual relationships, like a web of social relationships between people

4. A weighted, connected, cyclic, directed graph.

5. The loop has a weight of 3 and is on vertice 17.

6. Seven vertices, Seventeen edges

7. No, Yes, No

8. A connected, acyclic, directed graph.

9.

Breadth First	**Depth First**
● Uses stack ● Uses vertices ● Slower than DFS ● Moves across a level and then down, vertice by vertice	● Uses Queue ● Uses Edges ● Faster than BFS ● Two stages, following edges from root/parent to leaf

10.

Shortest path by weight from A to C : A - B - D - F - E - C
Shortest path in 12

	A	B	C	D	E	F
	∞	∞	∞	∞	∞	∞
A	**0a**	5a	18a	∞	∞	∞
B		**5a**	14b	7b	∞	∞
C			14b	**7b**	12d	10d
D			14b		11f	**10d**
E			12e		**11f**	
F			**12e**			

11.

a. Vertice with largest degree: MAD2104 with 5 outdegree + 2 indegree = 7 degree
b. Vertice with highest indegree: CDA4101 with 3 indegree
c. Vertice with highest outdegree: MAD2104 with 5 outdegree
d. Toposort output: MAC3311 COP3210
COP4555 COP3337 COP3400 MAD2104 CAP3700
CDA4400 CDA4101 COP3530 MAD3512 MAD3305
COP4225 COP4610 CIS4610 COP5621 COP4540