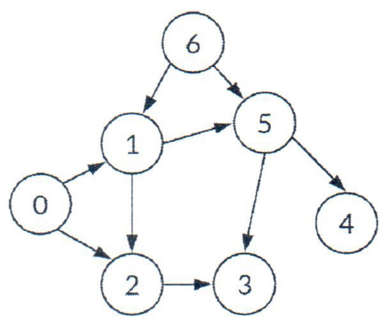
Algorithms
Comprehensive Exam
(Fall 2019)

SHORT QUESTIONS **(Answer any six questions, each carrying 8 points)**

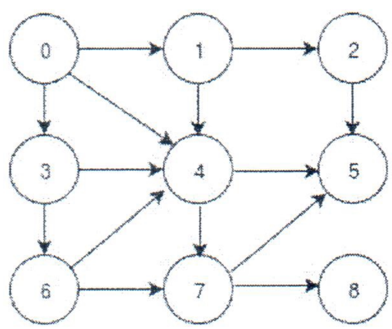
1. (a) Give a big-O (upper bound), a big-Omega (lower bound), and a big-Theta (tight bound) estimations, respectively, for $T(n)=3n^3 + 20n^2 + 1000$, where n is a positive integer.

   (b) Solve $T(n) = \log(n!)$.

2. The following graph shows a sequence of tasks with their dependencies on other tasks. A directed edge from vertex $\boldsymbol{u}$ to vertex $\boldsymbol{v}$ indicates task $\boldsymbol{u}$ must be completed before task $\boldsymbol{v}$ can start. Apply DFS (depth-first search) to find an ordering of the tasks that conforms to the given dependencies. (Show each step.)



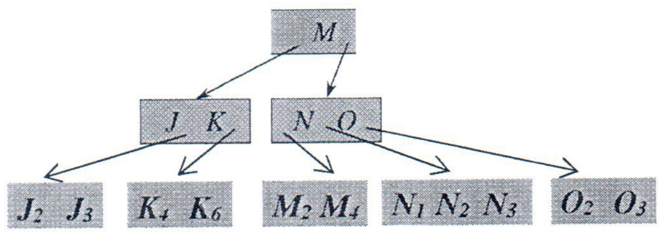
3. Give the visited node order for (a) breadth-first graph search and (b) depth-first graph search, starting with node 0 shown below.



4. For any n-key B-tree of height h and with the minimum node degree of $t \geq 2$, prove that h is no larger than $\log_t \frac{n+1}{2}$. (Hint: consider the number of keys stored in each tree level.)

5. Show your construction of an optimal Huffman code for the set of 7 frequencies: **a**: 3, **b**: 12, **c**: 5, **d**: 20, **e**: 16, **f**: 34, **g**: 18.

6. Given the initial B-tree with the minimum node degree of $t = 3$ below, show the results (a) after deleting the key of M_2, (b) followed by inserting L, with $K_i < L < M_j$ for all i and j, (c) then by deleting J_2, and (d) then by deleting O_2. (Show the result after each deletion and after each insertion.)



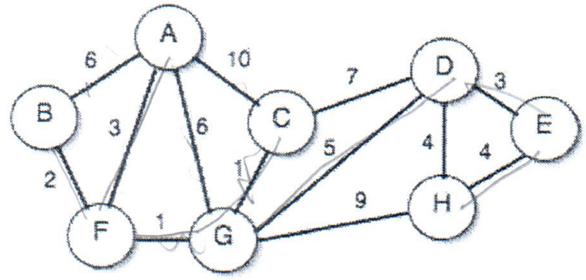
7. Many problems have been proved to be NP-complete. To prove NP-completeness, it is necessary in general to demonstrate two proof components. What are the two proof components to show a problem being NP-complete?

   Given that the Hamiltonian-cycle problem (HAM-CYCLE) belongs to NP-completeness, how do you prove that the traveling-salesman problem (TSP) is NP-complete?

   TSP has a 2-approximation solution in polynomial time based on establishing a minimum spanning tree (MST) rooted at the start/end vertex (in polynomial time following MST-PRIM), if the graph edge weights observe triangle inequality. Sketch a brief proof to demonstrate that such a solution satisfies 2-approximation.

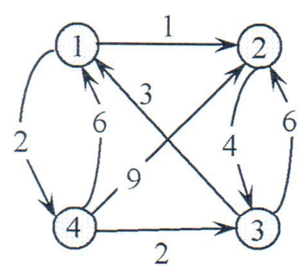
LONG QUESTIONS (Answer any four questions, each carrying 13 points)

1. Apply (a) Prim's algorithm and (b) Kruskal's algorithm to compute the minimum spanning tree of the given graph, respectively. For each algorithm, write down the edge picked in every step via its two end vertices; e.g., Step 1: (v1, v2), etc.



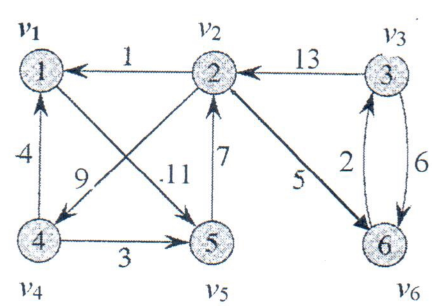
2. The Floyd-Warshall algorithm (*FW*) obtains all pairs of shortest paths in a weighted directed graph. Given the graph given below, what is the recursive equation of $d_{i,j}^{(k)}$ for the shortest-path weight of any path between *i* and *j* with intermediate vertices $\in \{1, 2, 3, \ldots, k\}$?

   Derive all distance matrices $D^{(k)}$ following *FW* so that the $d_{i,j}^{(n)}$ element of final matrix $D^{(n)}$ denotes $\delta(i, j)$ for every vertex pair $\langle i, j\rangle$ for all $i, j \in \{1, 2, 3, 4\}$.



3. The Dijkstra's algorithm (*DS*) solves the single-source shortest-path problem in a weighted directed graph $G = (V, E)$ without negative weighted edges or cycles, by edge relaxation at one vertex at a time until all vertices are examined. Given the graph G below, follow *DS* to find shortest paths from vertex v_1 to all other vertices, with all predecessor edges shaded and estimated distance values from v_1 to all vertexes provided at the end. Also list the sequence of vertices at which relaxation takes place.

   What is the time complexity of *DS* for a general graph $G = (V, E)$, when candidate vertices are kept in an array?



4. Use a hash table with size $m = 7$ to store the set of keys: $\{2, 3, 4, 7, 9, 11, 12\}$
   (a) Pick a hash function $h(k) = k \bmod 7$ to hash. If there exists collision in one entry, use a double hashing method with the probe sequence of $h(k, i) = (h_1(k) + i * h_2(k)) \bmod 7$ to handle collision, where $h_1(k) = h(k) = k \bmod 7$, and $h_2(k) = k \bmod 5 + 1$.
   Draw the table with keys hashed to each entry.

   (b) For the double hashing method above, can we change $h_2(k) = k \bmod 5 + 1$ to $h_2(k) = k \bmod 5$? Please explain.

5. Given the matrix-chain multiplication problem for four matrices sized 20×60, 60×15, 15×30, 30×50, follow the tabular, bottom-up method in the procedure of MATRIX-CHAIN-ORDER below to construct two tables of $m[i, j]$, for all $1 \le i, j \le 4$, and $s[i, j]$, for all $1 \le i \le 3$ and $2 \le j \le 4$. Construct the two tables, with their entry values shown.

```
MATRIX-CHAIN-ORDER(p)
n = p.length - 1
let m[1..n, 1..n] and s[1..n-1, 2..n] be new tables
for i = 1 to n
    m[i, i] = 0
for l = 2 to n                // l is the chain length
    for i = 1 to n - l + 1
        j = i + l - 1
        m[i, j] = ∞
        for k = i to j - 1
            q = m[i, k] + m[k + 1, j] + p_{i-1} p_k p_j
            if q < m[i, j]
                m[i, j] = q
                s[i, j] = k
return m and s
```

Good Luck!