國立臺灣師範大學資訊工程學系
九十九學年度第二學期
博士班資格考

考試科目：演算法
總分一百分
請在答案卷作答，在題目卷上作答不予計分

1. **(10pts)** Given the input array $A = [13, 19, 9, 5, 8, 7, 4, 21, 2, 6, 11]$, and we would like to use RANDOMIZED QUICKSORT to sort the elements in A into increasing order.

   (a) **(5pts)** Show the sequence of pivots that results in the worst-case running time.
   (b) **(5pts)** Show the sequence of pivots that results in the best-case running time.

2. **(10pts)** Given n integers in the range of 1 to k, describe an algorithm that is used to pre-process the integers so that we can access the number of integers that fall into a range $[a, b]$ in $O(1)$ time. Your preprocessing algorithm should perform in $O(n+k)$ time.

3. **(20pts)** Given two square n x n matrices $A = (a_{ij})$ and $B = (b_{ij})$, an ordinary algorithm SQUARE-MATRIX-MULTIPLY(A, B) uses three loops to compute the product $C = A \cdot B$, where $c_{ij} = \sum_{k=1}^{n} a_{ik} \cdot b_{kj}$, and it takes $\Theta(n^3)$ time. Alternatively, Strassen's algorithm solves the matrix multiplication problem by dividing the matrices as follows:

$$\begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \times \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}.$$

If we have

$$\begin{aligned}
m_1 &= (a_{11} + a_{22})(b_{11} + b_{22}) \\
m_2 &= (a_{21} + a_{22})b_{11} \\
m_3 &= a_{11}(b_{12} - b_{22}) \\
m_4 &= a_{22}(b_{21} - b_{11}) \\
m_5 &= (a_{11} + a_{12})b_{22} \\
m_6 &= (a_{11} - a_{21})(b_{11} + b_{12}) \\
m_7 &= (a_{12} - a_{22})(b_{21} + b_{22}),
\end{aligned}$$

the product C is then given by

$$C = \begin{bmatrix} m_1 + m_4 - m_5 + m_7 & m_3 + m_5 \\ m_2 + m_4 & m_1 + m_3 - m_2 - m_6 \end{bmatrix}.$$

(a) **(5pts)** How many scalar multiplications would be performed in finding the product of two 4 x 4 matrices using SQUARE-MATRIX-MULTIPLY(A, B)?

(b) **(5pts)** How many scalar multiplications would be performed in finding the product of two 4 x 4 matrices using Strassen's algorithm?

(c) **(5pts)** Write the recurrence for computing the time complexity of Strassen's algorithm.

(d) **(5pts)** Professor Gibson wishes to develop a matrix-multiplication algorithm that is asymptotically faster than Strassen's algorithm. His algorithm will use the divide-and-conquer method, dividing each matrix into pieces of size $n/4$ x $n/4$, and the divide and combine steps together will take $\Theta(n^2)$ time. He needs to decide how many sub-problems that his algorithm has to create in order to beat Strassen's algorithm. If his algorithm creates a sub-problems, the recurrence for the running time becomes $\mathrm{T}(n) = a\mathrm{T}(n/4) + \Theta(n^2)$. What is the largest integer value of a for which Professor Gibson's algorithm would be asymptotically faster than Strassen's algorithm?

4. **(10pts)** What does dynamic programming have in common with divide-and-conquer? What is a principal difference between the two techniques?

5. **(5pts)** There are two standard ways to represent a graph $G = (V, E)$: the adjacency list and the adjacency matrix.

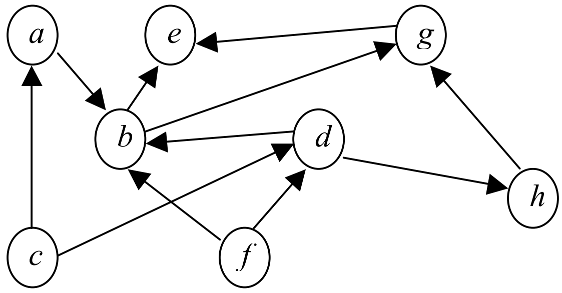
   (a) **(2pts)** Please give the space complexity of these two representations respectively (in terms of V and E).

   (b) **(3pts)** Please compare the two representations and describe how you choose between them.

6. **(5pts)** When an adjacency-matrix representation is used, most graph algorithms require time $\Theta(V^2)$, but there are some exceptions. Show that determining whether a directed graph contains a sink – a vertex with in-degree $|V| - 1$ and out-degree 0 – can be determined in time $O(V)$, even if an adjacency-matrix representation is used.

7. **(5pts)** Given a graph $G = (V, E)$ and a source vertex s. Define the shortest-path distance $d(s, v)$ from s to v as the minimum number of edges in any path from vertex s to vertex v, or else ∞ if there is no path from s to v. Please give an efficient algorithm (including the graph representation) to compute the shortest-path distance from s to all reachable vertices.

8. **(7pts)** Topological sort:
   (a) **(3pts)** Given a directed graph $G = (V, E)$, please describe the goal of topological sort, i.e. what does topological sort aim to find?
   (b) **(1pt)** In what condition, it is not possible to have a successful topological sort?
   (c) **(3pts)** Please re-draw the following graph in a topologically sorted manner.



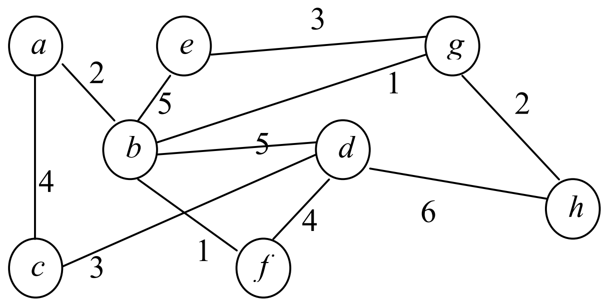
9. **(5pts)** Strongly connected components:
   (a) **(3pts)** Please give the definition of a strongly connected component of a directed graph $G = (V, E)$.
   (b) **(2pts)** How does the number of strongly connected components of a graph change if a new edge is added?

10. **(8pts)** Find the minimum spanning tree using Kruskal's algorithm:
   (a) **(2pts)** Please give the definition of the minimum spanning tree.
   (b) **(2pts)** Complete step 4 and 5 in Kruskal's algorithm.

```
MST-KRUSKAL(G, w)
1  A ← ∅
2  for each vertex v ∈ V[G]
3     do MAKE-SET(v)
4
5
6     do if FIND-SET(u) ≠ FIND-SET(v)
7        then A ← A ∪ {(u, v)}
8             UNION(u, v)
9  return A
```

   (c) **(2pts)** Find a minimum spanning tree in the following graph.

   (d) **(2pts)** Let (u, v) be a minimum-weight edge in a graph G. Show that (u, v) belongs to some minimum spanning tree of G.



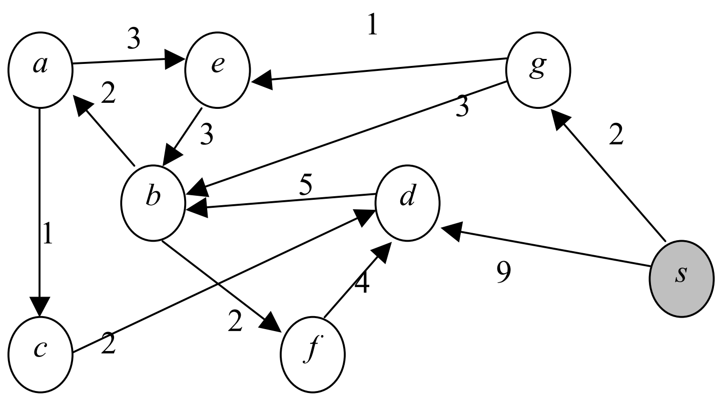
11. **(9pts)** Find the single-source shortest path on a weighted, directed graph $G = (V, E)$ using Dijkstra's algorithm

```
DIJKSTRA(G, w, s)
INITIALIZE-SINGLE-SOURCE(G, s)
S ← ∅
Q ← V[G]
while Q ≠ ∅
  do u ← EXTRACT-MIN(Q)
     S ← S ∪ {u}
     for each vertex v ∈ Adj[u]
        do RELAX(u, v, w)
```

(a) **(3pts)** Analyze the time complexity if we maintain the priority queue Q as a linear array. Explain how you compute the complexity in detail, not just give the result.

(b) **(3pts)** Analyze the time complexity if we implement Q by a binary heap. Explain how you compute the complexity in detail, not just give the result.

(c) **(3pts)** Apply Dijkstra's algorithm to find the shortest-path distance from s to all other nodes in the following graph.



12. **(6pts)** Find all-pairs shortest path on a weighted, directed graph using Floyd-Warshall algorithm.

FLOYD-WARSHALL(W)
$n \leftarrow \text{rows}[W]$
$D^{(0)} \leftarrow W$
for $k \leftarrow 1$ to n
  do for $i \leftarrow 1$ to n
    do for $j \leftarrow 1$ to n
      $d_{ij}^{(k)} \leftarrow \min(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)})$
return $D^{(n)}$

(a) **(1pts)** Please compute the time complexity.
(b) **(2pts)** Please explain the meaning of $d_{ij}^{(k)}$.
(c) **(3pts)** Run the Floyd-Warshall algorithm on the following graph. Show the matrix $D^{(k)}$ in each iteration.