Final (A)

Important: write down the letter in the above parentheses on the cover of your answer sheets NOW.

1. (18%) For a connected unweighted undirected graph $G = (V, E)$ and $v \in V$, let $\texttt{Far}(G, v)$ be an algorithm which computes a farthest vertex u from v. I.e., $\delta(u, v) = \max_{v' \in V} \delta(v', v)$. The diameter $D(G)$ of G is defined as $\max_{u,v \in V} \delta(u, v)$.
   (a) (5%) Describe a $O(|V| + |E|)$-time implementation of $\texttt{Far}(G, v)$.
   (b) (5%) Give an algorithm to compute $D(G)$ in $O(|V||E|)$-time. Briefly explain why it has time complexity $O(|V||E|)$.
   (c) (8%) Prove or disprove the following claim: If G is a tree, then the following algorithm output $D(G)$.
      - Let v be an arbitrary vertex in V. $\texttt{p=Far}(G, v)$. $\texttt{q=Far}(G,\texttt{p})$. $\texttt{BFS}(G,\texttt{q})$. return $\texttt{p.d}$.
2. (10%) Fact 1: A directed graph $G = (V, E)$ has a cycle if every vertex in G has in-degree greater than 0.
   Fact 2: If a vertex v in a directed graph $G = (V, E)$ has in-degree 0, then the following statement is true: G has a cycle if and only if $G_v = (V \backslash \{v\}, E \backslash \{(u, v) : u \in V\})$ has a cycle.
   By using these facts, give a non-recursive $O(|V| + |E|)$ algorithm to determine if a directed graph $G = (V, E)$ contains a cycle. Notice: no points for a DFS-based algorithm.
3. (10%) Let s be a stack which supports `empty`, `push`, `pop` and `multipop`. Show that $|s|$, the number of elements in s, is a potential function, and we can use this fact to prove that all four kinds of operations are in amortized $O(1)$-time.
4. (10%) Given that $d(\alpha, \alpha) = \alpha$ and $d(\alpha, \beta - 1) \le d(\alpha, \beta) \le d(\alpha + 1, \beta)$. Show the following claim is true: for every $c \in \{0, \ldots, n-2\}$, we have $\sum_{i=2}^{n-c} (d(i, i+c) - d(i-1, i+c-1) + 1) = O(n)$.
5. (10%) Turtle tower: there are n turtles $t_1, \ldots, t_n$. For $i \in \{1, \ldots, n\}$, t_i has weight w_i and strength s_i. WLOG, we assume $s_1 \le \cdots \le s_n$. Prove or disprove: if there exists a permutation $\pi : \{1, \ldots, n\} \to \{1, \ldots, n\}$ such that $s_{\pi(i)} \ge \sum_{j=1}^{i} w_{\pi(j)}$ for $1 \le i \le n$, then $s_i \ge \sum_{j=1}^{i} w_j$ for every $i \in \{1, \ldots, n\}$. Note: π is a permutation iff $\pi(i) = \pi(j) \iff i = j$.
6. (10%) The below implementation of Dijkstra's algorithm is modified from `http://www.csie.ntnu.edu.tw/~u91029/`. Analyze its time complexity in terms of $|V|$ and $|E|$.

```
struct Node {int v, d;}; // v: vertex, d: distance
bool operator<(const Node& n1, const Node& n2) {return n1.d > n2.d;} // for min heap use

int* dijkstra_without_decrease_key(int n, int s, int **w) { // n: |V|, s: source, w: adjacency matrix
    int *d = new int[n]; // d: length of the shortest paths, return value
    bool visit[n]; // visited == true, unvisited == false
    priority_queue<Node> PQ; // min-heap: extract-min and insertion are in O(logn)-time
    for(int i=0; i<n; i++) visit[i] = false;  // nodes are unvisited
    for(int i=0; i<n; i++) d[i] = 0x3FFFFFFF; // Infinity: 0x3FFFFFFF

    d[s] = 0;  PQ.push( (Node){.v=s, .d=d[s]} );

    for (int i=0; i<n; i++) {
        int a = -1; // so many students just do not modify this statement.
        while (!PQ.empty() && visit[a = PQ.top().v]) PQ.pop(); // find unvisited a with min distance
        if (a == -1) break; // if no such a, break the for-loop

        visit[a] = true; // visit a
        for (int v=0; v<n; v++) // for every v in V
            if (!visit[v] && d[a] + w[a][v] < d[v]) { // if (a,v) in E, then relax (a,v)
                d[v] = d[a] + w[a][v];
                PQ.push( (Node){.v=v, .d=d[v]} );
            }
    }
    return d;
}
```

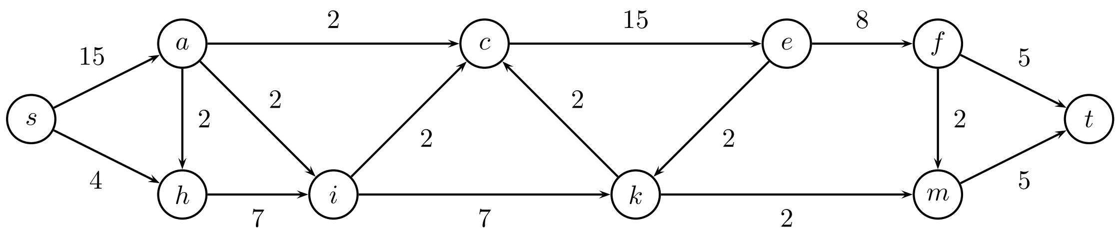
7. (10%) Give an algorithm that determines whether a weighted directed graph contains a negative cycle in $O(|V||E|)$-time. Hint: you have to modify the original Bellman-Ford algorithm, since it detects only the negative cycles reachable from the source.

8. (10%) Let $G = (\{v_1, \ldots, v_5\}, E)$ be a weighted directed graph without negative cycles, and $D^i(u, v)$ be the length of the shortest path from u to v passing only vertices in $\{v_1, \ldots, v_i\}$. Given D^2 as below, compute D^3. (Each mistake deducts 2 points.)

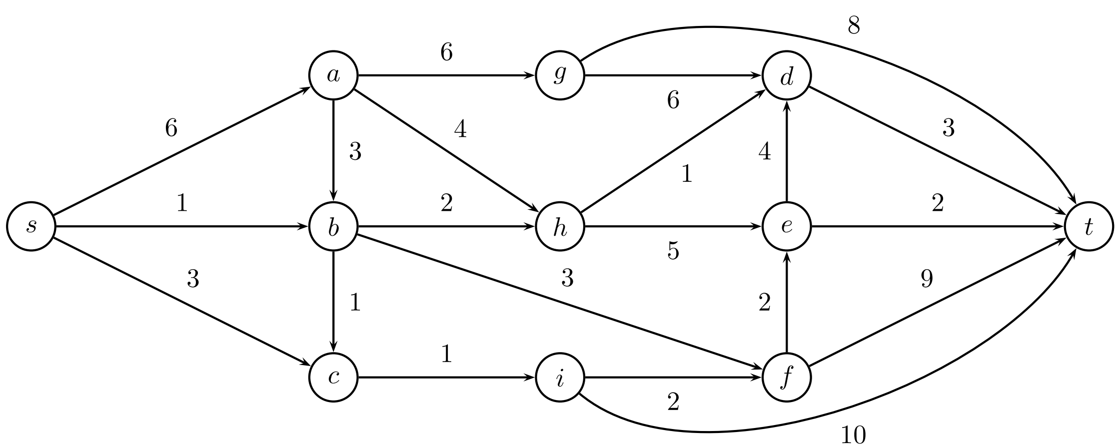
$D^2(u, v)$

$u \backslash v$	v_1	v_2	v_3	v_4	v_5
v_1	0	∞	-1	∞	∞
v_2	6	0	5	∞	∞
v_3	∞	∞	0	∞	9
v_4	-3	-9	-4	0	∞
v_5	∞	∞	∞	8	0

9. (15%) Use Edmond-Karp algorithm to compute the maximum flow f of the network below where s is the source and t is the sink. Notice: a flow is a function mapping edges into non-negative reals. (10 points for illustrating the whole process.)



10. (15%) Consider the directed graph G below. Compute three edge-disjoint paths p_1, p_2, p_3 from s to t such that the total length is minimized. Note: 5 points for the minimum total length.



11. (10%) Let $\Phi = (x_1 \vee x_2 \vee x_3) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_4}) \wedge (\overline{x_1} \vee x_2 \vee \overline{x_4}) \wedge (\overline{x_2} \vee x_4 \vee \overline{x_3}) \wedge (\overline{x_4} \vee x_3 \vee \overline{x_2}) \wedge (x_1 \vee \overline{x_2} \vee x_4)$. Find an assignment satisfying Φ. Illustrate the process.

Final (A)