請勿攜去
Not to be taken away

香 港 中 文 大 學
The Chinese University of Hong Kong

二零二五至二六年度上學期科目期末考試

Course Final Examination 1st Term, 2025-26

科目編號及名稱 Course Code & Title：	CSCI3160 Design and Analysis of Algorithms
時間 Time allowed：	2 小時 hours 0 分鐘 minutes
學號 Student I.D. No. ：	
座號 Seat No. ：	

Please read the following instructions carefully:

1. Please fill in your student ID number and seat number in the corresponding fields above.
2. This question booklet contains 9 questions, worth a total of 100 points. It consists of 4 pages. Please check to make sure no pages are missing.
3. Write all your answers in the answer book.

Exam Questions

1. (20 points) Solutions to True or False questions won't be provided.
2. (10 points) **[From special exercise.]** Question statement hidden.

Solution:

Round-by-Round Distance Table:

Round	**dist(a)**	**dist(b)**	**dist(c)**	**dist(d)**	**dist(e)**
0	0	∞	∞	∞	∞
1	0	-5	-7	-2	-5
2	0	-5	-7	-2	-5
3	0	-5	-7	-2	-5
4	0	-5	-7	-2	-5

3. (10 points) Question statement hidden.

Solution: First apply k-selection to find the k_1-th smallest integer x_1 in $O(n)$ expected time. Then apply k-selection again to find the k_2-th smallest integer x_2 in $O(n)$ expected time. Then scan S once more to report all the numbers $y \in S$ satisfying $x_1 \le y \le x_2$. The last step takes $O(n)$ time.

4. (12 points) **[From special exercise.]** Question statement hidden.

Solution: First, we claim that each clause is satisfied with probability at least $\frac{1}{2}$. This is because for a k-literal clause, it is satisfied with probability $1 - \frac{1}{2^k}$. Then, the claim follows by noticing that $k \geq 1$.

Next, assume there are n clauses in total. Let t_i ($\forall i \in [n]$) be a binary random variable denoting if the i-th clause is true ($t_i = 1$) or false ($t_i = 0$).

Then, the expected number of clauses satisfied is:

$$E[\sum_{i=1}^{n} t_i] = \sum_{i=1}^{n} E[t_i] \geq \frac{1}{2} \cdot n,$$

satisfying the definition of a "$\frac{1}{2}$ approximation ratio in expectation."

5. (5 points) Question statement hidden.

Solution: Consider a graph with two vertices and one edge between them. Our greedy algorithm will pick this edge and add both vertices into the cover set S, resulting in a cover of size 2. But the optimal solution consists of only one vertex.

6. (5 points) Question statement hidden.

Solution:

(a) If $\mathcal{C} = \mathcal{C}^*$, then the sets picked by our algorithm have already covered the entire universe. In that case, the algorithm would not have needed to pick S_i.

(b) Let $\mathcal{C}^*_{out} = \mathcal{C}^* \setminus \mathcal{C}$. The z_{i-1} elements not yet covered by $\{S_1, S_2, ..., S_{i-1}\}$ must be covered by $\mathcal{C}^*_{out}$. It then follows from the averaging argument that at least one of the sets in $\mathcal{C}^*_{out}$ must have a benefit at least $z_{i-1}/|\mathcal{C}^*_{out}| = z_{i-1}/(|\mathcal{C}^* \setminus \mathcal{C}|)$. The claim thus follows from the algorithm's greedy nature.

7. (10 points) Question statement hidden.

Solution: Let $G = (V, E)$ be the input graph.

We use the fact that for any graph G, a subset $I \subseteq V$ is an independent set if and only if $V \setminus I$ is a vertex cover.

So, to find a minimal vertex cover set:

- Run an algorithm that returns a *maximum independent set* $I \subseteq V$.
- Return $C = V \setminus I$ as the vertex cover set.

Since C is a maximum independent set, I must be a minimum vertex cover set. The reduction consists of complementing the output set, which takes linear time. Thus, the reduction is correct and runs in polynomial time.

8. (18 points) Question statement hidden.

Solution:

(a) Consider an MST T which contains e. Removing e breaks the tree into two connected components, say S and $V \setminus S$. Since all the vertices of the cycle cannot still be connected after removing e, at least one edge, say e', in the cycle must cross from S to $V \setminus S$. However, then replacing e by e' gives a tree T' such that cost$(T') \leq$ cost(T). Since T is an MST, T' is also an MST which does not contain e.

(b) If e is the heaviest edge in some cycle of G, then there is some MST T not containing e. However, then T is also an MST of $G - e$, and so we can simply search for an MST of $G - e$. At every step, the algorithm creates a new graph $(G - e)$ such that an MST of the new graph is also an MST of the old graph (G). Hence the output of the algorithm (when the new graph becomes a tree) is an MST of G.

(c) An undirected edge (u, v) is part of a cycle iff u and v are in the same connected component of $G - e$. Since the components can be found by DFS (or BFS), this gives a linear-time algorithm.

(d) The time for sorting is $O(|E| \log |E|)$, and checking for a cycle at every step takes $O(|E|)$ time. Finally, we remove at most $|E|$ edges (more accurately, $|E| - |V| + 1$ edges), and hence the running time is

$$O(|E| \log |E| + |E| \cdot |E|) = O(|E|^2).$$

9. (10 points) **(Discrete Fréchet Distance.)** Question statement hidden.

Solution: Let $L(i, j)$ denote the smallest possible length of the leash, such that the professor and the dog can move from p_1 and q_1 to p_i and q_j. For two points p and q, let $d(p, q)$ denote the distance between them.

Then, the recursive relation we want is:

$$L(i,j) := \begin{cases} d(p_1, q_1) & i = 1 \land j = 1 \\ \max\{d(p_1, q_j), L(1, j-1)\} & i = 1 \\ \max\{d(p_i, q_1), L(i-1, 1)\} & j = 1 \\ \max\left\{d(p_i, q_j), \min\{L(i-1, j), L(i, j-1), L(i-1, j-1)\}\right\} & \text{otherwise} \end{cases}$$

The three first cases correspond to the following three base cases: (1) both Professor Bille and the dog are on the first nodes in their respective paths; (2) Professor Bille stands on the first node (p_1) while only the dog is moving; (3) the dog stands on the first node (q_1) and Professor Bille is the only one who is moving.

In the general case (4), we must use a leash of length at least the distance between p_i and q_j since they could not otherwise stand on p_i and q_j respectively. Furthermore the leash must be long enough for the steps made so far. We consider the following three cases for the previous step and take the best of these (i.e. the smallest): (a) Professor Bille moves; (b) the dog moves; (c) they both move.

The full algorithm is shown in Algorithm 1. Since we store the results to the subproblems in a table L of size $n \times m$ we use $\Theta(nm)$ space. The outer loop always makes n iterations and the inner loop always makes m iterations per iteration of the outer loop. All other operations can be done in constant time, and thus the total time of the algorithm is $\Theta(nm)$.

Algorithm 1:

```
Require: Two sequences of points $P = (p_1, \dots, p_n)$ and $Q = (q_1, \dots, q_m)$
Ensure: Discrete Fréchet distance $L(n, m)$
function FRECHET($P, Q$)
    initialize $L$ as an $n \times m$ table
    for $i \leftarrow 1$ to $n$ do
        for $j \leftarrow 1$ to $m$ do
            if $i = 1$ and $j = 1$ then
                $L[i][j] \leftarrow d(p_1, q_1)$
            else if $i = 1$ then
                $L[i][j] \leftarrow \max(L[i][j-1], d(p_1, q_j))$
            else if $j = 1$ then
                $L[i][j] \leftarrow \max(L[i-1][j], d(p_i, q_1))$
            else
                $L[i][j] \leftarrow \max\big\{d(p_i, q_j), \min\{L(i-1, j), L(i, j-1), L(i-1, j-1)\}\big\}$
    return $L[n][m]$
```

— End —