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

Course Examination 1st Term, 2022-2023

Course Code & Title : CSCI3160 Design and Analysis of Algorithms

Time allowed : 2 hours 0 minutes

Student I.D. No. : .... Seat No. : ....

NOT TO BE TAKEN AWAY

Note 1: Write all your answers in the answer book.
Note 2: You do not need to describe any algorithms in lectures, tutorials, or regular exercises.
Note 3: Full mark of the paper is 115.

List of Acronyms:

APSP: all-pairs shortest paths
SCC: strongly connected component
SSSP: single-source shortest path

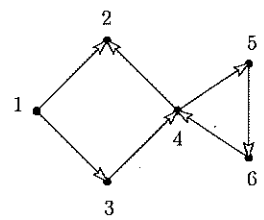
Problem 1 (20 marks, 2 for each statement). Determine whether each of the following statements is true (if you think statement $i \in [1, 10]$ is true, write "i: T" in the answer book; otherwise, write "i: F"):

1. $n^{1.04-c} = O(n^{1.04}/(\log_2 n)^{1000})$ for any constant $c > 0$.
2. Consider any *connected* graph $G = (V, E)$ (i.e., G has only one connected component). If an algorithm has running time $O(|E|^c)$ on G where $c > 0$ is a constant, then the running time is polynomial in $|V|$.
3. Let Σ be an arbitrary finite alphabet. If we encode each letter in Σ with the same number of bits (ensuring that the letters' codewords are distinct), then the encoding must be a prefix code.
4. For the rod cutting problem, consider the price table below:

length	1	2	3	4	5	6	7	8
price	3	5	8	9	10	17	17	20

The maximum revenue attainable from a rod of length 8 is 22.

5. Let $G = (V, E)$ be the graph shown below.



(1, 3, 4, 2, 6, 5) can be the sequence of nodes visited by a depth-first search on G (with restarts).

6. Let G be a simple directed graph. Let u and u' be two vertices in the same SCC of G. Then, for any vertex v in G, there is a path from u to v if and only if there is a path from u' to v.

7. For solving the SSSP problem on graph $G = (V, E)$ where all edges have *positive* weights, Bellman-Ford's algorithm has the same time complexity as Dijkstra's algorithm.

8. In this question, APSP and SSSP problems concern only simple directed graphs $G = (V, E)$ with no negative cycles. Suppose that we already know that no algorithm can solve the APSP problem in $O(|V|^{2.99})$ time. Then, we can assert that no algorithm can solve the SSSP problem in $O(|V|^{1.99})$ time.

9. Let $G = (V, E)$ be a simple directed graph where every edge carries a weight that may be negative. Suppose that G has no negative cycles. Then, for any distinct vertices $u, v \in V$, every shortest path from u to v must be simple (i.e., no vertex can appear on the path more than once).

10. Consider the boolean expression $x_1 \vee x_2 \vee ... \vee x_n$ where each x_i ($i \in [1, n]$) is a boolean variable. Suppose that $n \geq 4$. If we independently set each x_i to 0 or 1 with an equal probability, the expression evaluates to 1 with probability at least 15/16.

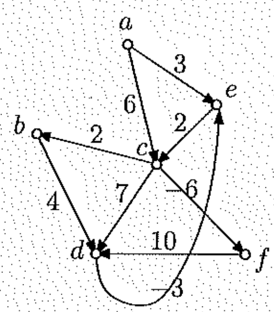
Problem 2 (10 marks). Let S be a set of n integers, and k_1, k_2 be arbitrary integers satisfying $1 \leq k_1 \leq k_2 \leq n$. Suppose that S is given in an array. Give an $O(n)$ expected time algorithm to report *all* the integers whose ranks in S are in the range $[k_1, k_2]$. Recall that the rank of an integer v in S equals the number of integers in S that are at most v.

Problem 3 (10 marks). Define function $f(x)$ — where $x \geq 0$ is an integer — as follows:

- $f(0) = 0$
- $f(1) = 1$
- $f(x) = f(x-1) + f(x-2)$.

Give an algorithm to calculate $f(n)$ in $O(n)$ time (you can assume that $f(x)$ fits in a word for all $x \leq n$).

Problem 4 (15 marks). Consider the weighted directed graph $G = (V, E)$ below.



Suppose that vertices a, b, c, d, e, and f have IDs 1, 2, 3, 4, 5, and 6, respectively. Find the values of $spdist(a, d \mid\leq k)$ for $k = 0, 1, ..., 6$. Recall that $spdist(a, d \mid\leq k)$ is the shortest length of all paths from a to d that use only intermediate vertices with IDs at most k.

Problem 5 (15 marks). Given a string s of length n and each integer $i \in [1, n]$, we use $s[i]$ to denote the i-th character of s and $s[1:i]$ to denote the prefix of s with length i. Let t be another string of length m satisfying $m \leq n$. A *subsequence occurrence* of t in s is defined as a sequence of m integers $(p_1, p_2, ..., p_m)$ such that

- $1 \leq p_1 < p_2 < ... < p_m \leq n$;
- $s[p_i] = t[i]$ for each $i \in [1, m]$.

For example, if $s =$ `sophisticated` and $t =$ `sit`, then the following are all the subsequence occurrences of t in s: (1, 5, 7), (1, 5, 11), (1, 8, 11), and (6, 8, 11).

(a) (10 marks) For any $i \in [1, n]$ and $j \in [1, m]$ such that $i \geq j$, define $count(i, j)$ as the number of subsequence occurrences of $t[1:j]$ in $s[1:i]$. For example, if $s =$ `sophisticated` and $t =$ `sit`, then $count(13, 3) = 4$ and $count(8, 2) = 3$.

Give a recurrence that relates $count(i, j)$ to $count(i-1, j)$ and $count(i-1, j-1)$. You need to explain why your recurrence is correct.

(Hint: Consider whether $i = j$ and independently whether $s[i] = t[j]$.)

(b) (5 marks) Give an algorithm to compute $count(n, m)$ in $O(nm)$ time.

Problem 6 (15 marks). Let $G = (V, E)$ be a simple directed graph given in the adjacency-list format. You can assume $V = \{1, 2, ..., n\}$. Answer the following questions:

(a) (5 marks) Design an algorithm to decide in $O(|V| + |E|)$ time whether G is a *chain*, namely, all the edges in E form a path that goes through all the n vertices.

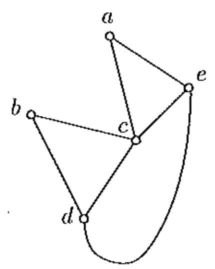
(b) (10 marks) Design an algorithm to decide in $O(|V| + |E|)$ time whether G is *semi-connected*, namely, for any vertices $u, v \in V$, G has a path from u to v or a path from v to u.

You need to analyze the running time of your algorithms.

Problem 7 (15 marks). Let M be an $n \times n$ matrix where n is a power of 2. Each row of M is sorted (from left to right) in ascending order, and each column is sorted (from top to bottom) in ascending order. An example with $n = 4$ is $M = \begin{bmatrix} 1 & 20 & 60 & 80 \\ 10 & 25 & 65 & 90 \\ 12 & 37 & 70 & 95 \\ 23 & 40 & 83 & 100 \end{bmatrix}$. M is stored in an array such that, for any $i \in [1, n]$ and $j \in [1, n]$, $M[i, j]$ can be retrieved in $O(1)$ time. Given an arbitrary integer q, design an algorithm to determine whether q is in M using $O(n)$ time. You need to analyze the running time of your algorithm.

You get 5 marks if your algorithm runs in $O(n^{\log_2 3})$ time.

Problem 8 (15 marks). Let $G = (V, E)$ be a simple undirected graph. A *5-cycle* is a cycle with 5 edges. We say that a subset $D \subseteq E$ is a *5-cycle destroyer* if removing the edges of D destroys all the 5-cycles in G, namely, $G' = (V, E \setminus D)$ has no 5-cycles. For example, if G is the graph below, there is only one 5-cycle *acbdea*; a 5-cycle destroyer is $\{\{d, e\}\}$, and so is $\{\{d, e\}, \{c, d\}\}$.



Let D^* be a 5-cycle destroyer with the minimum size. Design an algorithm to find a 5-cycle destroyer of size $O(|D^*| \cdot \log |V|)$ in time polynomial to $|V|$. You need to analyze the running time and approximation ratio of your algorithm.

-End-