THE UNIVERSITY OF SYDNEY

Student number:

Room number:

Seat number:

ANONYMOUSLY MARKED
(Please do not write your name on this exam paper)

CONFIDENTIAL EXAM PAPER

This paper is not to be removed from the exam venue.

Computer Science

EXAMINATION

Semester 2- Practice, 2024

COMPX123 Data Structures and Algorithms

EXAM WRITING TIME: 120 minutes
READING TIME: 10 minutes

EXAM CONDITIONS:
This is a RESTRICTED OPEN book examination
- specified materials permitted.

MATERIALS PERMITTED IN THE EXAM VENUE:
One A4 sheet of handwritten and/or typed notes double-sided.
(No electronic devices of any kind are permitted.)

MATERIALS TO BE SUPPLIED TO STUDENTS:
This exam paper booklet.

INSTRUCTIONS TO STUDENTS:
Write your student ID at the top right of this front page.
Write your answers in the spaces provided in the exam paper booklet using blue or black ink. DO NOT ATTACH EXTRA PAGES TO THIS EXAM BOOKLET. If you run out of space in the space provided right after each problem statement, continue your answers on pages 10-16.

For examiner use only:

Problem	1	2	3	4	5	Total
Marks						
Out of	10	10	10	10	20	60

Weeks	3, 8
4	4, 7
7	1, 3, 7, 9
8	2, 6, 7
10	2, 3, 4
11	1, 2, 6

Problem 1.

a) Analyze the time complexity of this algorithm.

```
def COMPUTE(A)
    result ← 0                          O(1)
    for i = 0; i < n; i++ do            ⎤
        if A[i] > i then        O(1)    ⎥ O(n)
            result ← result + A[i]  O(1)⎦
    return result                       O(1)
```

$O(1) + O(n) \cdot [O(1)] = O(n)$

traverse the array

b) Solve the following recursion using unrolling:

$$T(n) = \begin{cases} T(n/2) + O(1) & \text{for } n > 1 \\ O(1) & \text{for } n = 1 \end{cases}$$

n — c 1

$n/2$ — c 1

$n/4$ $\log_2(n)$

$\vdots$

1 — c 1

$\Rightarrow \log_2(n)$

Master: $T(n) = 2^n \rightarrow$ Case 3

$T(n) = \log(n) \rightarrow$ Case 1

Problem 2. We are planning a board games event and we're using one of the shelves in my office to store the games. Unfortunately the shelf only has a certain amount of space S, so we need to carefully pick which games we want to bring. Every game takes some space s_i and has a fun factor f_i that indicates how much fun it is to play that game (for $1 \le i \le n$).

We want to maximize the amount of fun we'll have, so we want to maximize the sum of the fun factors of the games we pick (i.e., $\max \sum_{\text{picked game } i} f_i$), while making sure that the games fit on my shelf, so the sum of the space the games we pick take should be at most S (i.e., $\sum_{\text{picked game } i} s_i \le S$). For simplicity, you can assume that all f_i, s_i, and S are all distinct positive integers.

The strategy of PICKLARGEST is to always pick the game with the highest fun factor until my shelf is full: it sorts the games by their fun factor f_i in decreasing order and adds a game when its required space is less than the remaining space on the shelf.

def PICKLARGEST(all f_i and s_i, S)
    $currentSpace \leftarrow 0$
    $currentFun \leftarrow 0$
    Sort games by f_i and renumber such that $f_1 \ge f_2 \ge ... \ge f_n$
    for $i \leftarrow 1; i \le n;$ i++ **do**
        if $currentSpace + s_i \le S$ **then**
            $currentSpace \leftarrow currentSpace + s_i$ ▷ Pick the ith game
            $currentFun \leftarrow currentFun + f_i$
    return $currentFun$

Show that PICKLARGEST doesn't always return the correct solution by giving a counterexample.

Greedy question.

real value $= \frac{f_i}{s_i}$

(solution to Problem 2 continues here)

Problem 3. Consider the *Dynamic Matrix* ADT for representing an matrix $A = \{a_{i,j}\}_{i,j=1}^{n}$ that supports the following operations:

$\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \to \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

- $O(1)$ CREATE(): creates a 1×1 matrix where $a_{1,1} = 0$.
- $O(1)$ SET/GET(i, j): set or get the value of the entry $a_{i,j}$.
- $O(n)$ INCREASE-SIZE: If the current size of the matrix is $n \times n$, increase it to $n+1 \times n+1$ such that the new entries are set of 0. In other words, A becomes A' such that $a'_{i,j} = a_{i,j}$ if $1 \leq i, j \leq n$, and $a'_{i,j} = 0$ otherwise.

Your task is to come up with a data structure implementation for the Dynamic Matrix ADT that uses $O(n^2)$ space, where n is the size of the matrix, and CREATE, SET, GET take $O(1)$ and INCREASE-SIZE takes $O(n)$ time. Remember to:

a) Describe your data structure implementation in plain English.

b) Prove the correctness of your data structure.

c) Analyze the time and space complexity of your data structure.

Map

array of arrays

when we do increase-size we just

(1) create 2 n+1 empty array
$2 \times O(n)$

(2) copy old array element pointer to new arrays

(3) assign 0's

(solution to Problem 3 continues here)

same points

Problem 4. Let G be a connected undirected graph on n vertices. We say that two distinct spanning trees T and S of G are *one swap away* from each other if $|T \cap S| = n - 2$; that is, T and S differ in only one edge.

For two distinct spanning trees T and S we say that $R_1, R_2, \ldots, R_k$ form a *swapping sequence* from T to S if:

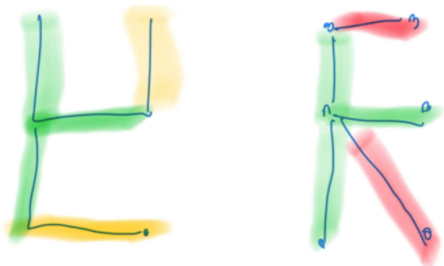
1. $R_1 = T$,
2. $R_k = S$, and
3. for any $1 \leq i < k$, the trees R_i and R_{i+1} are one swap away from each other

Your task is to design a polynomial time algorithm that given G and two spanning trees T and S of G, constructs a minimum length swapping sequence. Remember to:

a) Describe your algorithm in plain English.
b) Prove the correctness of your algorithm.
c) Analyze the time complexity of your algorithm.

$O(n^3)$

T S no cycles



从黄色中 remove one

从红色中 find one that one end point is one of yellow

then do a dfs, if there is no cycle we do it

otherwise find other solution

(solution to Problem 4 continues here)

Problem 5. (Description of hard problem here)

(solution to Problem 5 continues here)

THIS AND FOLLOWING PAGES LEFT INTENTIONALLY BLANK.

Use this and following pages for additional writing space if you run out of space in an answer area. If you want any writing on this page or following pages to be marked, you MUST write "see additional pages" in the relevant answer area and clearly indicate the question number you are answering on the relevant page. Any writing outside of this booklet will not be examined.

(use the rest of the space of rough work)

(use the rest of the space of rough work)

(use the rest of the space of rough work)

(use the rest of the space of rough work)

(use the rest of the space of rough work)