CS2040 — AY2025/2026 Semester 1

Midterm

90 minutes / 100 marks

Multiple Choice Questions [40 marks]

Q1 [4 marks]

What is the worst-case time complexity of g(n) (positive n)? Select the best option.

```
void g(int n) {
      if (n <= 3) return;
      if (n % 2 == 1) g(n-1);    // if odd n case
      else g(n / 2);             // watch the parameters carefully
}
```

A. O(log n)
B. O((log n)^2)
C. O(n$^{0.5}$)
D. O(n)
E. O(n log n)
F. O(n^2)

Q2 [4 marks]

Given a non-empty singly linked list of **N** nodes, and a head reference pointing to the first node. The following code is run:

```
ListNode temp = head;
while (temp != null) {
    temp.next = head;
    temp = temp.next;
}
```

i. The list remains unchanged
ii. The program enters infinite loop
iii. The last (**N**-th) node points back to head, making the list circular
iv. The head reference remains the same during execution of the code

Select the best option.

A. Only i
B. Only ii
C. Only iii
D. Only iv
E. ii and iv
F. ii and iii
G. iii and iv

Q3 [4 marks]

Which operation is **faster on average** in a basic linked list (no tail pointer) compared to an array-based list where all elements are in sequence from index 0 onwards?

A. Accessing the last element
B. Insertion at index 0
C. Sorting the elements
D. Random access

Q4 [4 marks]

For a **complete binary tree** of height ***h***, what is the minimum possible number of nodes it has? Assume that a tree with a single node has height 0.

A. $2^h - 1$

B. $2^h + 1$

C. $2^{h-1} + 1$

D. $2^{h+1} - 1$

E. 2^h

F. None of the other options is the correct answer.

Q5 [4 marks]

Which of the following statement regarding stack/queue is true? Choose the best answer.

A. **ALL** of Stack's push, pop, peek operations (properly implemented) can sometimes take **longer/worse than** O(1) time
B. If an array-based FIFO Queue is compared to a linked-list implementation of Queue (both properly implemented), the dequeue operation in the array-based Queue has better time complexity because arrays have random access
C. When an element in a list has to be efficiently compared to the earliest element among all elements from the same list that have previously been processed but have not yet been eliminated, a FIFO Queue may help
D. When an element in a list has to be efficiently compared to the earliest element among all elements from the same list that have previously been processed but have not yet been eliminated, a Stack may help

Q6 [4 marks]

You have a list of students. You want to create a class list such that it is split with females on top and males below, and within each category the names are in alphabetical order.

Which option among those listed below provides the best strategy: First, sort by _____ using _____, then sort by _____ using _____.

A. (gender, quicksort, name, mergesort)
B. (gender, mergesort, name, quicksort)
C. (name, quicksort, gender, mergesort)
D. (name, mergesort, gender, quicksort)
E. None of the other options listed will work.

Q7 [4 marks]

Your boss just attended the l33t4sp33d conference and is obsessed with everything speed related. He heard that radix sort is the fastest sort.

Your boss insists that you must use radix sort, as implemented in lectures, to sort n items, where the labels, used as keys, are unique integers from 1 to n. What is the order of growth of time to sort the **n items**?

After some time in development, the customer requests some **change to the labels**. Your boss still insists that you must use radix sort. You still need to **sort n items**, but the labels are now unique integers **from 1 to n^2**. What is the order of growth of time needed to sort the items with the **new labels**?

A. Initially: O(n), after change: O(n)
B. Initially: O(n), after change: $O(n^2)$
C. Initially: O(n log n), after change: O(n log n)
D. Initially: O(n log n), after change: $O(n (\log n)^2)$
E. Initially: O(n log n), after change: $O(n^2 \log n)$
F. Initially: O(n log n), after change: $O(n^2 (\log n)^2)$

Q8 [4 marks]

Which of the following statement regarding hash table, as taught in lectures, is true? Choose the best answer.

A. Lookups based on a key take amortized O(1) time, i.e. it is not possible for many lookups one after another to all take O(N) time each
B. When deleting an element, the operation first inputs the key into a hash function
C. Repeatedly inserting the same key into an empty hash table N times (N is large) may cause the next operation to perform very poorly
D. When a hash table expands, the existing elements from the old table are all copied to the same index in the new table
E. Inserting an element into a Direct Addressing Table takes **O(1) average** time, since many probes can occur due to collisions but that seldom happens

Q9 [4 marks]

Which of the following statement regarding array-based binary max heap is **false**? Choose the best answer.

A. It is possible that 2 leaves in the binary heap are on different levels of the tree
B. Deletion of the largest element (if exists) removes the element at the lowest valid index
C. Deletion of the largest element (if exists) will always take O(1) time
D. Inserting an element could cause elements to be shifted upwards
E. Finding the largest element will always take O(1) time

Q10 [4 marks]

What is the worst-case time complexity of f(n) (positive n)? Select the best option.

```
int f(int n) {
      int res = 1;
      for (int i = 1; i <= n; i *= 2)
            for (int j = 1; j <= n; j *= 2)
                  for (int k = 1; k <= i; k++)
                        res *= k;
      return res;
}
```

A. O(n)
B. O(n log n)
C. O(n $(\log n)^2$)
D. O(n^3)
E. O(n^3 log n)
F. O(n!)

Analysis Questions [16 marks]

QA1 [8 marks]

You are given the following functions f and g:

```
int f(int n) {
    int sum = 0;
    for (int i = 0; i < n; i = i + 1) {
        for (int j = 0; j < i; j = j + 1) {
            sum += 1;
        }
    }
    return sum;
}
int g(int n) {
    int sum = 0;
    for (int i = 0; i < n; i = i + 1) {
        sum += f(i * i); // calls function f defined earlier
    }
    return sum;
}
```

If it helps, for any positive integer k constant, $\sum_{i=1}^{n} i^k$ is in $O(n^{k+1})$

e.g. $1 + 2 + 3 + 4 + \ldots + n\text{-}3 + n\text{-}2 + n\text{-}1 + n$ is in $O(n^2)$

e.g. $1^2 + 2^2 + 3^2 + 4^2 + \ldots + (n\text{-}3)^2 + (n\text{-}2)^2 + (n\text{-}1)^2 + n^2$ is in $O(n^3)$

e.g. $1^3 + 2^3 + 3^3 + 4^3 + \ldots + (n\text{-}3)^3 + (n\text{-}2)^3 + (n\text{-}1)^3 + n^3$ is in $O(n^4)$

(This is a fact, not a claim)

Claim: g(n) runs in $O(n^2)$ time.

a) [2 marks] The claim is True / False.

b) [6 marks] Provide a brief explanation to support your answer. If your answer is False, you also need to state the correct time complexity of g(n).

QA2 [8 marks]

Selection sort, as implemented in lectures, is run on an array **A** of **N** elements, where each element is a large piece of data. Suppose each element contains **M** items, where **M** can be large. It is NOT known whether **M** or **N** is larger.

Due to each element being very large, moving/copying/swapping/shifting elements takes O(**M**) time. However, comparing two elements is fast since each element's key has been pre-computed, taking O(1) time per comparison.

Claim: The **tightest** time complexity of using selection sort to sort **A**, in terms of both **M** and **N**, is $O(\mathbf{MN}^2)$.

a) [2 marks] The claim is True / False.

b) [6 marks] Provide a brief explanation to support your answer. If your answer is False, you also need to state the correct time complexity.

Essay Questions [44 marks]

QE1 [11 marks]

At Singa Airport, passengers are all standing in one line. They are then supposed to be filtered into two lanes for security screening: anyone carrying **less than x milliliters** of liquids is sent to the **Fast Lane**, while everyone with **$\geq x$ ml** goes to the **Regular Lane**.

You are given the head of a **non-empty singly linked list** of integers, where each node represents a passenger, with the data/value/item being the liquid volume each passenger carries.

Due to manpower shortage, the two lanes will have to be combined into one, with all **Fast Lane** passengers being **served first**, and those in the **Regular Lane behind**. To be fair, the **relative order of passengers within each lane must be preserved**.

Write an algorithm to **re-link the existing nodes** so that all nodes with `val < x` come before all nodes with `val ≥ x`, **stably** (i.e., preserve the original relative order inside each of the two lanes), and then return a reference to the head of the new linked list representing the sequence of passengers in the combined security screening lane.

Your solution should use **constant extra space** to store the liquid volumes, i.e. you can create a constant amount of nodes or other variables if necessary. Solutions that use larger than O(1) extra space to store this data will be awarded low marks.

Example 1
Input: head = 3 → 5 → 8 → 5 → 10 → 2 → 1, x = 5
Output: 3 → 2 → 1 → 5 → 8 → 5 → 10
Explanation: All nodes with value < 5 (3, 2, 1) come before nodes ≥ 5 (5, 8, 5, 10). Within each group, their relative order is unchanged from the input.

Example 2
Input: head = 1 → 4 → 2 → 2 → 7, x = 3
Output: 1 → 2 → 2 → 4 → 7

QE2 [11 marks]

The Great Library of Cossun is due to undergo renovations. Oizne Mak, the librarian, is tasked to move some of the books in the library to the storage room. Each book has a magic number, where the magic number x for each book is a **unique integer** such that $1 \leq x \leq n$ for integer n. There are a total of n books that need to be moved.

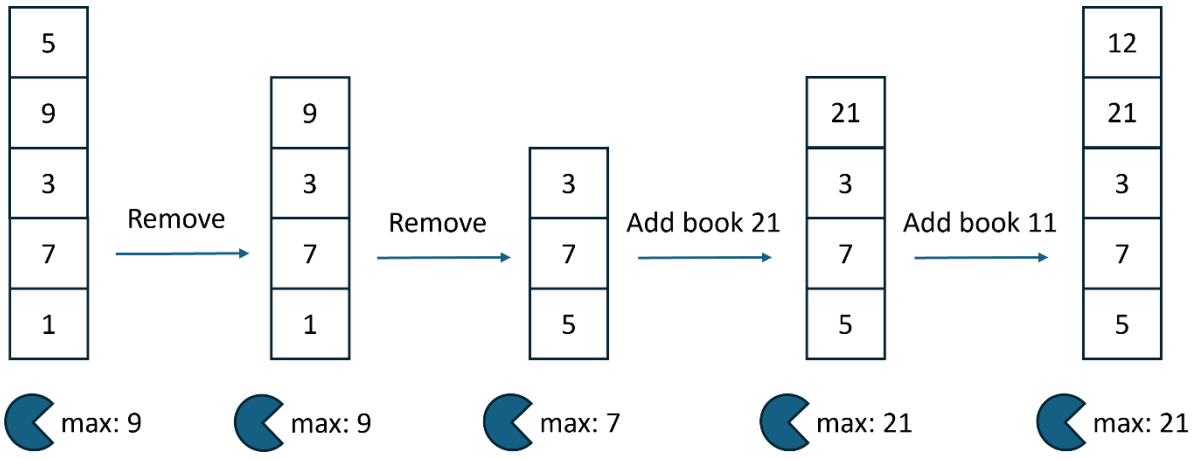
Oizne Mak wants to arrange the books into piles so that it will be easier to transport the books. He will add the books one by one to a pile, sometimes also removing books one by one if he realises that he made a mistake, and those books belong somewhere else.

However, the playful magician Ecneics strikes again! He plays another prank on Oizne Mak and casts a spell, which now requires Oizne Mak to shout the maximum magic number of ALL the books currently in the pile.

Oizne Mak needs to be able to do three things:

i. Add a book to the top of the pile.
ii. Remove the book at the top of the pile.
iii. Shout the maximum magic number of all the books currently in the pile.

Consider the following example:



Above figure shows a pile of books with magic numbers 1, 7, 3, 9, 5 from bottom to top. Removing book 5 from the top does not change the maximum magic number. Subsequent removal of book 9 reduces the maximum magic number to 7. Then, adding book 21 to the top increases the maximum magic number to 21, but subsequently adding book 12 to the top does not cause any changes to the maximum magic number.

Design an efficient algorithm and/or data structure so that Oizne Mak can arrange the books such that you get the **best worst-case time complexity for each operation (i), (ii) and (iii)**.

QE3 [11 marks]

Given a non-empty array **A** of **N** integers (the integers could be positive, negative or zero, and could possibly have very large magnitude), and a positive integer **K**, output how many elements occur at least 2 more times within the next **K** elements (or if there are less than **K** elements after, then up to the end of the array). Design an efficient algorithm to do so.

Example 1
Inputs: **K** = 4, **A** = [1, 2, 3, 1, 1, 2, 2, 1, 2]
Output: 3
Explanation (Don't need to output these) :
Element 1 at index 0 (0-based) has another 2 occurrences of element 1 within the next 4 spaces
Element 1 at index 3 has another 2 occurrences of element 1 within the next 4 spaces
Element 2 at index 5 has another 2 occurrences of element 2 within the next 3 spaces (till end)

Example 2
Inputs: **K** = 5, **A** = [9, 3, 2, 1, 3, 3, 3, 1, 1, 2, 1, 2, 1, 2, 5, 5, 5, 7, 7, 7, 9, 5, 7]
Output: 9
Explanation (Don't need to output these) :
Element 3 at index 1 (0-based) has another 3 occurrences of element 3 within the next 5 spaces
Element 1 at index 3 has another 2 occurrences of element 1 within the next 5 spaces
Element 3 at index 4 has another 2 occurrences of element 3 within the next 5 spaces
Element 1 at index 7 has another 3 occurrences of element 1 within the next 5 spaces
Element 1 at index 8 has another 2 occurrences of element 1 within the next 5 spaces
Element 2 at index 9 has another 2 occurrences of element 2 within the next 5 spaces
Element 5 at index 14 has another 2 occurrences of element 2 within the next 5 spaces
Element 7 at index 17 has another 3 occurrences of element 7 within the next 5 spaces
Element 7 at index 18 has another 2 occurrences of element 7 within the next 4 spaces (till end)

Correct O(**N**) average time or better solutions will obtain the full 11 marks
Otherwise correct O(**N** log **N**) time or better solutions will obtain 8 marks
Otherwise correct O(**NK**) or better solutions will obtain 5 marks

QE4 [11 marks]

Tom is using a submarine to scout a portion of the sea. The submarine can only be dropped once. The submarine sinks vertically till it hits a solid area (the sea floor / seabed), then moves as far right as possible at the current depth (without sinking/rising). Your task is to help Tom find and output the longest possible horizontal distance that can be travelled by the submarine.

The map of the sea and surrounding solid areas is modelled as an **N**-by-**N** 2D-array (i.e. a square grid) (**N** > 3), with each cell containing either 'O' or 'X' : 'X' for solid areas where the submarine cannot pass through, and 'O' for water. There may be solid areas suspended in the middle of the sea. Rows correspond to depths of the sea. Therefore, the top of row 0 is the sea level, where the submarine can be released from. If a cell at row 0 contains water, the submarine may be released at the top-leftmost point in that cell. The bottom-most row and the right-most column are guaranteed to contain only solid areas.

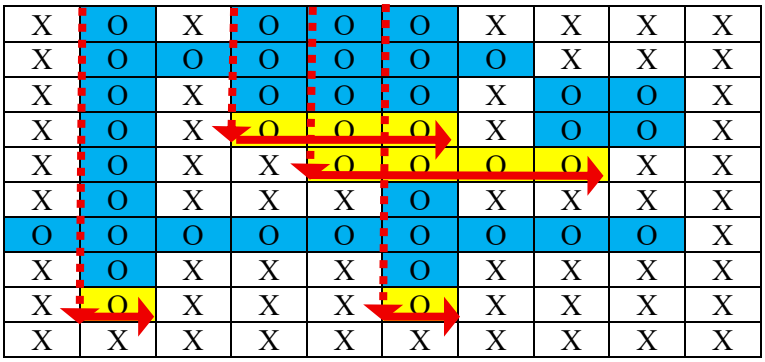
For this question, it is alright to mutate/edit/corrupt the grid, i.e. you may choose to store elements other than 'O' or 'X' in any cell as you wish.

Let the number of cells containing water be **T**, and **T** << **N** (i.e. **T** lower order of growth than **N**) when **N** is large. Given both the integer **N** and the grid **A**, design an efficient algorithm to help Tom.

Example

```
Inputs: N = 10, A = [
            ['X','O','X','O','O','O','X','X','X','X'],
            ['X','O','O','O','O','O','O','X','X','X'],
            ['X','O','X','O','O','O','X','O','O','X'],
            ['X','O','X','O','O','O','X','O','O','X'],
            ['X','O','X','X','O','O','O','O','X','X'],
            ['X','O','X','X','X','O','X','X','X','X'],
            ['O','O','O','O','O','O','O','O','O','X'],
            ['X','O','X','X','X','O','X','X','X','X'],
            ['X','O','X','X','X','O','X','X','X','X'],
            ['X','X','X','X','X','X','X','X','X','X']
      ]
Output: 4
```

Explanation (don't need to output these):



Dropping the submarine at column 1 (0-based) allows it to travel 1 cell to the right
Dropping the submarine at column 3 allows it to travel 3 cells to the right
Dropping the submarine at column 4 allows it to travel 4 cells to the right
Dropping the submarine at column 5 allows it to travel 1 cell to the right
Therefore, 4 (cells to the right) is the longest possible horizontal distance here

Reminder: The grid has **N** rows and **N** columns, but has $\mathbf{N}^2$ cells.

Correct O(**T** + **N**) average time or better solutions **that use O(1) extra space** will obtain the full 11 marks
Otherwise correct O(**T** + **N**) average time or better solutions will obtain 10 marks
Otherwise correct O(**TN**) or O($\mathbf{T}^2$ + **N**) time or better solutions will obtain 7 marks

Correct O($\mathbf{N}^2$) time or worse solutions will obtain low marks…!!