ANDERSON SERANGOON JUNIOR COLLEGE
JC2 Preliminary Examination 2024
Higher 2

COMPUTING 9569/01

Paper 1 (Written) Suggested Answers **11 September 2024 (Wednesday)**

3 hours

READ THESE INSTRUCTIONS FIRST

An answer booklet will be provided with the question paper. You should follow the instructions on the front cover of the answer booklet. If you need additional writing paper ask the invigilator for a continuation booklet.

Answer **all** questions.
Approved calculators are allowed.

The number of marks is given in brackets [ ] at the end of each question or part question.
The total number of marks for this paper is 100.

This document consists of **6** printed pages and **0** blank page.

1 A gym's membership management system uses Object Oriented Programming (OOP) in its design. The gym offers three types of membership: Basic, Pro and Family.

For every gym member, its member ID, name, mobile number and membership start date need to be stored. The system should also allow for the annual renewal of membership.

Basic members are only allowed non-peak hours gym usage daily. There are no restricted hours for Pro and Family memberships.

Pro members are entitled to 10 complimentary training sessions with a qualified personal trainer of choice.

For Family members, up to three family members' names can be stored.

Basic members will be charged a monthly fee, while Pro members will be charged 1.3 times this rate. Each Family member will be charged 1.5 times this rate, with each additional family member being charged 0.5 times this rate. Membership fees are charged via a `calculate_fee()` method at the start of each membership cycle.

(a) For the Basic base class, explain how encapsulation can be achieved. [2]

- Encapsulation - grouping of data and methods within a class entity with protected access
- Define all attributes of the class as private to prevent direct access to the data from outside the class.
- Provide public methods to access and modify the private attributes to allow controlled access to the data

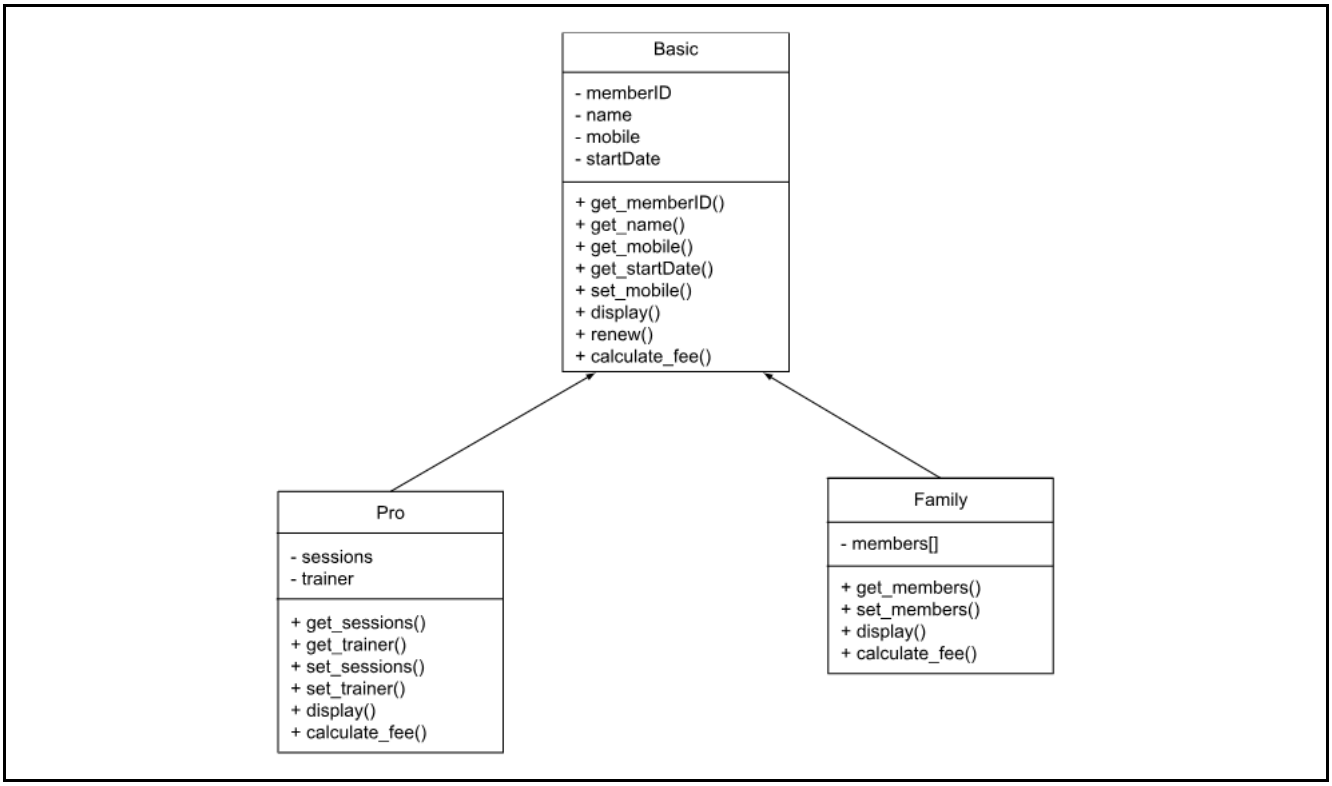
(b) Explain the rationale of creating the Pro and Family subclasses. [2]

- Inheritance - ability of subclasses to adopt data(state) and methods(functionality) from superclass
- Leverage by reusing common data/state and methods/functionality from superclass without reinventing the wheel or unnecessary code duplication (better readability and maintainability)
- Override and/or Implement data and methods specific to the subclasses allowing them to have new/extended state and functionality

(c) How does this class relationship demonstrate polymorphism? [3]

- Polymorphism - ability to invoke different methods(behaviour) using the same method name, promoting code generalisation
- By having different classes implementing their specific calculate_fee() method, different computation of membership fees can be effected using the same method call without the use of conditional checks for membership types/classes
- eg b1, p1, f1 representing objects of Basic, Pro and Family classes respectively
  members = [b1, p1, f1]
  for member in members:
  member.calculate_fee()

(d) Draw a class diagram for this system showing the base class and the derived classes, including relevant attributes and methods necessary for the system to function. [6]

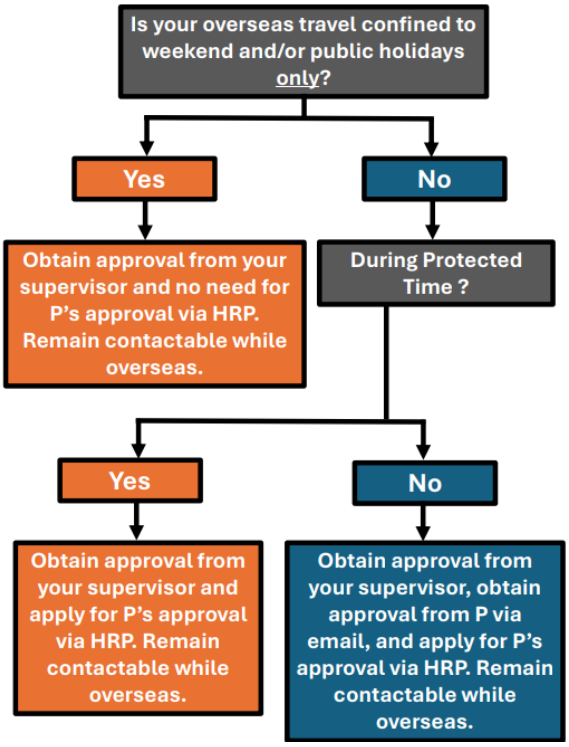


(e) The gym business owner wishes to launch a referral promotion for members to refer their friends to join the gym. Each successful referral will extend an existing member's current membership validity period by one month. Describe the necessary change(s) that can be made to the class diagram to accommodate this updated requirement. [2]

- add membership end date attribute and referral() method to superclass Basic
- upon every successful referral, invoke referral() method to extend membership end date by one month
- modify calculate_fee() method to trigger fee payable using membership end date

accept other reasonable answers

2 ABC School has the following workflow for teachers to apply for vacation leave.



Note:

- Each teacher has a reporting supervisor
- HRP (Human Resource Portal) is an online platform for teachers to access services such as leave application

(a) Convert the workflow to a decision table, showing all possible conditions and actions. [3]

Conditions				
Weekend and/or public holiday?	Y	Y	N	N
During protected time?	Y	N	Y	N
Actions				
Obtain supervisor approval	X	X	X	X
Obtain P approval via email				X
Apply P approval via HRP			X	X

(b) Simplify your decision table in (a) by removing redundancies. [2]

Conditions			
Weekend and/or public holiday?	Y	N	N
During protected time?	-	Y	N

Actions			
Obtain supervisor approval	X	X	X
Obtain P approval via email			X
Apply P approval via HRP		X	X

(c) Translate the decision table to pseudocode using appropriate variables. [4]

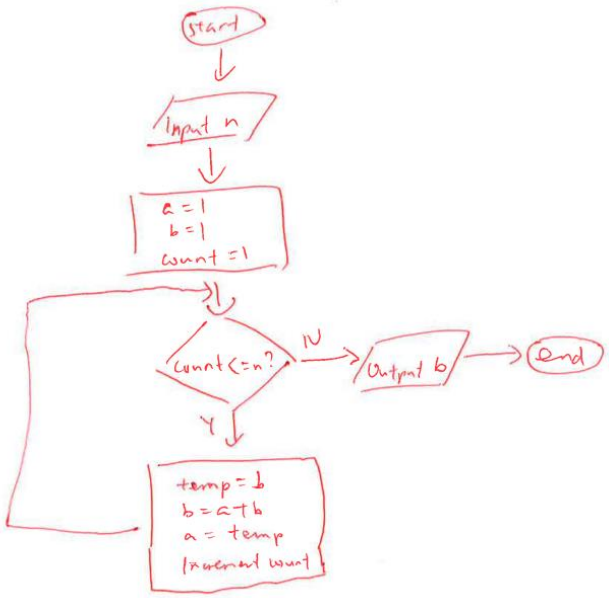
```
FUNCTION ApplyVacationLeave(isWeekendOrHoliday, isProtectedTime)
    // Initialize actions
    obtainSupervisorApproval = true // Always obtain supervisor approval
    obtainPApprovalViaEmail = false
    applyPApprovalViaHRP = false

    IF isWeekendOrHoliday
        obtainPApprovalViaEmail = true
    ELSE IF isProtectedTime
        applyPApprovalViaHRP = true
    ELSE
        obtainPApprovalViaEmail = true
        applyPApprovalViaHRP = true
    ENDIF

    // Return required actions
    RETURN [obtainSupervisorApproval, obtainPApprovalViaEmail,
applyPApprovalViaHRP]
ENDFUNCTION
```

3 The Fibonacci sequence is a series of numbers in which each number is the sum of the two previous numbers: 1, 1, 2, 3, 5, 8, …

(a) Draw a program flowchart to iteratively compute the nth Fibonacci number, n > 0. [3]



(b) The following pseudocode shows an algorithm to iteratively compute the nth Fibonacci number:

```
01 FUNCTION fibonacci_iterative(n)
02     IF n <= 2
03         return n
04     ENDIF
05     a = 1
06     b = 1
07     FOR i FROM 2 TO n
08         temp = a + b
09         a = b
10         b = temp
11     ENDFOR
12     RETURN b
13 ENDFUNCTION
```

Identify (specify line numbers) and rectify two logic errors with the above algorithm. [4]

- line 03 incorrectly returning initial values
- correction: return 1
- line 07 incorrect starting value in for loop
- correction: FOR i FROM 3 to n

(c) Produce the pseudocode for a `fibonacci_recursive(n)` algorithm to determine the nth Fibonacci number recursively. [3]

```
FUNCTION fibonacci_recursive(n):
   IF n <= 2
      RETURN 1
   ELSE
      RETURN fibonacci_recursive(n - 1) + fibonacci_recursive(n - 2)
   ENDIF
```

```
ENDFUNCTION
```

(d) Why is recursive Fibonacci less efficient than iterative Fibonacci for large n? [2]

- Redundant calculations: recursive approach recalculates the same Fibonacci numbers multiple times. For example, to calculate F(5), it calculates F(4) and F(3). But to calculate F(4), it again calculates F(3). This redundancy grows exponentially as n increases.
- Call stack overhead: each recursive call adds a new frame to the call stack. For large n, this can lead to stack overflow errors if the recursion depth exceeds the available stack space.

(e) Suggest a way to overcome the issue identified in (d). [3]

- memoisation to store computed fibonacci numbers for future retrieval
- before performing any calculation, check if result is already in the memo. eliminating redundant calculations
- while still recursive, significantly reduces the number of recursive calls, lowering the risk of stack overflow

4 (a) What advantage does binary search and merge sort have in common compared to linear search and bubble sort? [3]

- Both binary search and merge sort utilise divide and conquer approach, involving breaking down the problem into smaller subproblems, solving them, and then combining the results, giving significantly better performance on large datasets.
- For searching, binary search can eliminate half of the remaining elements with each comparison, while linear search must check each element sequentially.
- For sorting, merge sort divides the array into smaller subarrays, sorts them, and then merges them back together, while bubble sort makes multiple passes through the entire array, swapping adjacent elements.
- Both binary search and merge sort efficiency involve logarithmic time complexity due to this divide characteristic compared to the linear one in linear search and bubble sort

(b) For a small dataset which is almost sorted, explain if quick sort is a suitable sorting algorithm. If not, suggest and justify a suitable sorting algorithm. [3]

- Quick sort is not ideal for small, almost sorted datasets due to potential worst-case performance (O(n^2)) and overhead from partitioning.
- Insertion sort is more suitable because it's adaptive, performing efficiently (O(n) in best case) on nearly sorted data with minimal comparisons and swaps.
- For small datasets, insertion sort's simplicity and lower overhead often outweigh the average-case benefits of more complex algorithms like quick sort.

(c) For a large dataset with duplicates, explain if quick sort is a suitable algorithm. If not, suggest and justify a suitable sorting algorithm. [3]

- Quick sort can be inefficient for datasets with many duplicates, potentially leading to unbalanced partitions and O(n^2) worst-case performance.
- Merge sort is more suitable for large datasets with duplicates, offering consistent O(n log n) time complexity regardless of data distribution.
- Merge sort efficiently handles duplicates without special treatment, maintains stability, and performs reliably on large datasets, though it requires O(n) additional space.

(d) Explain why the efficiency of merge sort is O(n lg n). [4]

- Divide and Conquer: Merge sort repeatedly divides the input array into halves until individual elements are reached.
- Logarithmic Depth: The number of divisions required to reach individual elements is $\log_2(n)$, where n is the number of elements.
- Linear Merging: Each level of merging combines subarrays in linear time, proportional to the number of elements being merged.
- Total Complexity: The algorithm performs $\log_2(n)$ levels of merging, each taking O(n) time, resulting in a total time complexity of O(n log n).

(e) For the purpose of illustration, using the integers 1 to 15 inclusive and each only once (i.e. no duplicates), give and explain a **best case** dataset scenario for quick sort if the middle element is selected as the pivot.

For example, for integers 1 to 5 inclusive and no duplicates, a **worst case** dataset scenario for quicksort is [5, 4, 3, 2, 1] if the first element is selected as the pivot because the pivot will always have maximum number of elements in the left array and minimum number or no elements in the right array, resulting in the maximum number of recursive calls and worst case efficiency of quadratic time complexity $O(n^2)$.

[3]

- Best-case dataset: [8, 4, 2, 1, 3, 6, 5, 7, 9, 12, 10, 11, 14, 13, 15]
- Explanation: The middle element (8) is the median, creating perfectly balanced partitions. This pattern continues recursively, with each subarray's middle element being its median.
- Result: Balanced partitioning at each step leads to minimal recursion depth ($\log_2 n$), with each element used as pivot only once, achieving the best-case O(n log n) time complexity.

5 Every year, there is a large number of people balloting for a fixed and limited number of National Day Parade (NDP) tickets. For NDP2024, successful applicants were notified via email. While official figures are not disclosed, about or more than 15,000 spectators were reported by mainstream media.

Moving forward, to prevent the problems of bounced emails and phishing, the organising committee would like to provide a web service to allow applicants to check their application status online.

While there is no perfect data structure, after careful consideration, the organising committee is contemplating the use of a hash table to store successful applicants' identification and their status.

(a) Justify why a hash table would be a suitable data structure in this context. [3]

- Maps potentially large address key space to fixed hash table size with ideally O(1) lookup performance and practically O(small k) performance if number of collisions is small given well designed hashing function
- For both successful and unsuccessful applicants, search performance can be from best case O(1) (no collision) to O(k) (with collision)
- Fast lookup time: Hash tables offer O(1) average-case time complexity for lookups, allowing for near-instant status checks regardless of the number of applicants.
- Space efficiency: Hash tables provide a good balance between memory usage and performance, storing only necessary information (ID and status) without excessive overhead.
- Scalability: Hash tables can easily accommodate a growing number of applicants without significant performance degradation, making them suitable for future NDPs with potentially larger applicant pools.

(b) What would be a suitable hash table size and why? [2]

- Prime number slightly larger than 30,000: This accounts for the reported 15,000 spectators and allows for potential growth, while being prime ensures more uniform distribution and reduces the risk of clustering.
- Load factor consideration: This size maintains a load factor of about 0.5, balancing space efficiency and performance. It leaves room for additional entries without requiring frequent resizing, while keeping collision probabilities low.

(c) To handle collision resolution, the developer decides to implement quadratic probing. What problem does quadratic probing address? [2]

- Primary clustering: Unlike linear probing, quadratic probing helps prevent the formation of long contiguous blocks of occupied slots.
- Quadratic increments spread out collisions more evenly, reducing likelihood of performance degradation that occurs when many elements cluster together.

(d) Devise an algorithm to perform hash table search using quadratic probing collision resolution strategy. [5]

```
FUNCTION hash_table_search(hash_table, ID):
    table_size = length(hash_table)
```

```
    hash_value = hash(ID) % table_size
    original_hash = hash_value

    i = 0
    WHILE True
        IF hash_table[hash_value] == None: # Not found
            RETURN "Unsuccessful"
        ELSE IF hash_table[hash_value][0] == ID: # Found
            RETURN "Success!"
        ELSE # Collision
            i = i + 1
            hash_value = (original_hash + i * i) % table_size # Quadratic probing
    ENDWHILE
ENDFUNCTION
```

(e) Compare and contrast the advantages and disadvantages of using an array over a hash table to store successful applicants' identification and ballot status information. [3]

Lookup Time:
- Array: O(n) average and worst-case time complexity for lookups, as it may require scanning the entire array.
- Hash Table: O(1) average-case time complexity for lookups, offering near-instant access.
- Advantage: Hash table significantly outperforms for large datasets.

Memory Usage:
- Array: Can be more memory-efficient if the applicant IDs are sequential and dense, as it doesn't require additional structures.
- Hash Table: May use more memory due to its internal structure and the need for empty slots to manage collisions.
- Advantage: Array can be more space-efficient in specific scenarios.

Flexibility and Maintenance:
- Array: Simpler to implement but less flexible. Adding or removing entries can be inefficient, especially if maintaining sorted order.
- Hash Table: More complex but highly flexible. Easily handles dynamic data with efficient insertions and deletions.
- Advantage: Hash table offers better adaptability to changing data.

In summary, while arrays are simpler and can be memory-efficient in specific cases, hash tables generally provide superior performance and flexibility for the NDP ticket application.

6 A theatre has an existing computer system that stores customer bookings in a spreadsheet. The following are some sample entries recorded:

CustomerName	Email	ShowName	ShowTime	Duration	Seats	Amount
Amil Bin Osman	amilbo@email.com	Miss Saigon	2024-08-30 1930	160	F09	$128.00

Bridgette Smith	bridgettes@email.com	Playing with Fire	2024-09-06 2000	90	G07, G08, G09, G10	$220.00
Chen Tiong Kai	chentk@email.com	Miss Saigon	2024-08-30 1930	160	J20, J21, J22	$196.00

(a) Explain the problems arising using this method of storing booking records. [2]

- Data redundancy, which increases risk of inconsistencies (e.g. updating timing of a show across all relevant records may not be done properly).
- Unable to enforce data integrity constraints (e.g. a seat may be doubly booked for the same show)

(b) State why the table is not in First Normal Form. [1]

- The 'seats' column contains multiple seat entries for some of the records, which violates 1NF, which requires each field to contain only atomic values.

A relational database in Third Normal Form is to be used for the updated booking system. The database contains three tables: `Customer`, `Show` and `Booking`.

(c) Create an entity-relationship diagram for the database. [3]

Customer --<- Booking ->-- Show

(d) One of the tables is designed as follows:

`Customer(CustomerID, CustomerName, Email)`

Give the table specification for the other two tables in the database, clearly identifying all primary and foreign keys required. [3]

- Show(ShowID, ShowName, ShowTime, Duration)
- Booking(CustomerID#, ShowID#, Seats, Amount)

foreign key

assume one customer can only make one booking for each show (not unreasonable)

(e) In the nomalised database design above, explain why the `Booking` table requires a composite key. [2]

- Uniqueness: The combination of CustomerID and ShowID uniquely identifies each booking. This ensures that each customer can have only one booking for a particular show, preventing duplicate entries and maintaining data integrity.
- Relationship representation: The composite key effectively represents the many-to-many relationship between customers and shows. It directly links each customer to the shows they have booked, without needing an additional artificial / system-generated unique identifier.

(f) A practical alternative to a composite key is to use a system-generated primary key. Discuss the pros and cons of this approach. [2]

Pros
- Simplicity: A single-column primary key is simpler to manage and reference, especially in queries and when creating relationships with other tables. It provides a straightforward, unique identifier for each booking.
- Flexibility: This approach allows for multiple bookings by the same customer for the same show, which might be necessary in some scenarios (e.g., booking tickets for friends or family).

Cons:
- Additional storage: Introducing an extra column (BookingID) increases the storage requirements, albeit minimally in most cases.
- Less natural representation: The system-generated ID is an artificial construct that does not inherently represent the booking relationship between customers and shows, potentially making the data model less intuitive at a glance.

7 Recently, CNA reported that "The Ministry of Education (MOE) has taken legal action against "relevant contractors" following a Mobile Guardian cyberattack that affected 13,000 users from 26 secondary schools."
Source: https://www.channelnewsasia.com/singapore/mobile-guardian-cybersecurity-breach-attack-legal-action-contractors-chan-chun-sing-4597791

(a) From the article:

> About one in six of the affected users lost some data due to the cybersecurity breach suffered by the device management app, Minister for Education Chan Chun Sing said in parliament on Tuesday (Sep 10).
>
> Less than 5 per cent were unable to recover all their data as their devices had not been backed up before the Aug 4 breach, he added.

Explain the significance of backup and the difference between backup and archive in this context. [3]

- Data protection: Backups are crucial for protecting against data loss in cybersecurity incidents. The fact that only 5% of users lost all their data highlights how effective backups can be in mitigating the impact of breaches.
- Disaster recovery: Regular backups enable faster recovery and minimize disruption to students' education. Without backups, the impact on the affected schools could have been much more severe.

Backup vs. archive:
- Backup: Copies of current data intended for short-term recovery after data loss or corruption.
- Archive: Long-term storage of historical data, often for compliance or reference purposes.

In this case, backups were more relevant for quickly restoring recent student work.

(b) From the same article:

> "Prior to the Aug 4 incident, Mobile Guardian suffered a data breach in April due to poor password management practice. A glitch was also reported in July due to human error."

(i) Suggest two ways to harden the password management system in a networked environment. [2]

Implement Multi-Factor Authentication (MFA):

- Require at least two forms of authentication for all user logins.
- This could include something the user knows (password), something they have (security token or smartphone app), and/or something they are (biometric data).
- MFA significantly reduces the risk of unauthorized access even if passwords are compromised.

Use a centralized Password Management System:

- Deploy an enterprise-grade password manager that generates, stores, and manages complex passwords.
- Enforce strong password policies (length, complexity, regular changes) through the system.
- Implement role-based access control to limit password visibility and sharing.
- Utilize features like automatic password rotation and real-time monitoring for suspicious activities.

(ii) What could be a likely 'human error' and suggest one possible preventive and one possible corrective measure. [3]

- Likely human error: Misconfiguration of access controls or security settings, potentially granting unauthorized users access to sensitive data or systems.
- Preventive measure: Implement a "four-eyes principle" for critical system changes, requiring two authorized personnel to review and approve configuration modifications before they are applied.
- Corrective measure: Establish an automated rollback mechanism that can quickly revert system configurations to a known-good state in case of detected misconfigurations or unauthorized changes.

(c) The article continues…

"IMPACT ON STUDENTS

The 13,000 personal learning devices that were remotely wiped out represented about 8 per cent of devices used by the secondary school population.

MOE deployed 300 additional IT engineers and staff to help students, and provided instruction sheets to those who wanted to troubleshoot on their own. All devices were restored for use last month.

Schools provided hardcopy resources and supported students who were emotionally affected, said Mr Chan.

Deadlines were extended and weighted assessments were postponed where needed, he added.

At the school level, adjustments have been made according to the school's specific circumstances and needs."

Suggest two other support measures to mitigate the impact on students. [2]

- Peer support network: Establish a "tech buddy" system where students with restored devices or stronger tech skills assist their peers. This promotes collaborative problem-solving and reduces the emotional stress on affected students.
- Temporary device loan program: Implement a short-term device loaning system where unaffected students or the school lends devices to those whose laptops are still being restored. This ensures continuous access to digital learning resources for all students.

(d) The article continues…

The attack surface is "wide" and it is not possible to "defend everywhere with the same resources, with the same level of focus", he said.

At the national level, critical information infrastructure gets the most resources, and the level of security in other areas would vary depending on the system, said Mr Chan, describing it as a tiered and risk-based approach.

"It would not be practical to try to achieve the same level of security for all systems, he said."

In the context of schools, propose and justify how you would adopt a tiered and risk-based approach for online services used by students and teachers. Highlight the roles of firewall (filtering function), intrusion detection system (IDS), intrusion prevention system (IPS), encryption, digital signature, and authentication. [7]

- Tier 1 (Highest Security): Critical administrative systems
  - Strict firewall rules to limit access to authorized IP ranges
  - IPS for real-time threat prevention
  - Strong encryption (e.g., AES-256) for data at rest and in transit
  - Multi-factor authentication for all users
  - Digital signatures for all official communications

- Tier 2: Student information systems and grade databases
  - Firewall configured to allow access only from school networks
  - IDS for continuous monitoring and alerting
  - Encryption for sensitive data fields

    - Role-based access control with strong password policies
- Tier 3: Learning management systems and educational platforms
    - Firewall allowing broader access but filtering malicious content
    - Basic IDS for anomaly detection
    - SSL/TLS encryption for all communications
    - Single sign-on (SSO) authentication integrated with school accounts
- Tier 4 (Lowest Security): Public-facing school websites
    - Web application firewall to protect against common web attacks
    - Basic encryption (HTTPS) for all pages
    - Content filtering to prevent inappropriate material

- Network segmentation: Implement VLANs to separate different tiers, limiting potential breach impacts

- Regular security audits: Conduct periodic penetration testing and vulnerability assessments, prioritizing higher tiers

- User education: Provide tiered security training, with more in-depth programs for those accessing higher-security systems

- Incident response plan: Develop a comprehensive plan with specific procedures for each tier, ensuring quick and appropriate responses to potential breaches

END OF PAPER