DUNMAN HIGH SCHOOL
Preliminary Examination
Year 6

COMPUTING

9569 / 01

Paper 1

17 September 2020

3 hours

READ THESE INSTRUCTIONS FIRST

An answer booklet will be provided with this question paper. You should follow the instructions on the front cover of the answer booklet. If you need additional 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 125.

1 Given an array of numbers A, count the minimum number of 'bubble sort' swaps (swap between pair of consecutive items) needed to sort the array in ascending order.

For example, if A = [3, 2, 1, 4], we need 3 'bubble sort' swaps to sort A in ascending order i.e.

swap (3, 2) to get [2, 3, 1, 4]
swap (3, 1) to get [2, 1, 3, 4]
swap (1, 2) to get [1, 2, 3, 4]

(a) Devise an $O(n^2)$ algorithm using bubble sort to count the number of 'bubble sort' swaps.

(b) Devise an O(n log n) algorithm to count the number of 'bubble sort' swaps.

You should also explain why each algorithm has its efficiency. [8]

(c) Would you expect insertion sort to perform better or worse than bubble sort in (a)? Explain your answer with respect to the number of comparisons needed using the above example. [5]

(d) Why is quick sort not a suitable algorithm in part (b)? Illustrate your answer with a suitable example. [5]

2 You have been tasked to use a suitable data structure to manage the preliminary exam results of students in Dunman High School. Each student is identified by its centre and index numbers, each of which is 4-digit. For example, Lim Ah Seng's identification number is 30420188. A typical range of students' identification numbers are from 30420001 to 30420450, since each graduating cohort will have about 450 students.

The preliminary examination details to be stored are as follows:

- Subject code (4-digit) eg 9569
- Subject name eg H2 Computing
- Subject grade (1-character eg 'A')

You may assume that each student will have a valid subject grade in the range of ['A', 'B', 'C', 'D', 'E', 'S', 'U', 'T', '0'], where 'T' stands for terminated and '0' stands for absent.

Using suitable examples, evaluate the pros and cons using each of the following data structure to store the required information:

(a) dictionary

(b) hash table

[12]

3 The information in Question 2 will also be stored in a relational database.

(a) Why is a relational database model more suitable than a NoSQL database model for storing the required information? [3]

(b) Draw an ER diagram for a normalised database design. [4]

(c) Produce the specification for the tables. [5]

(d) Using examples in this context, explain the significance of the following terms:

(i) primary key [2]

(ii) foreign key [2]

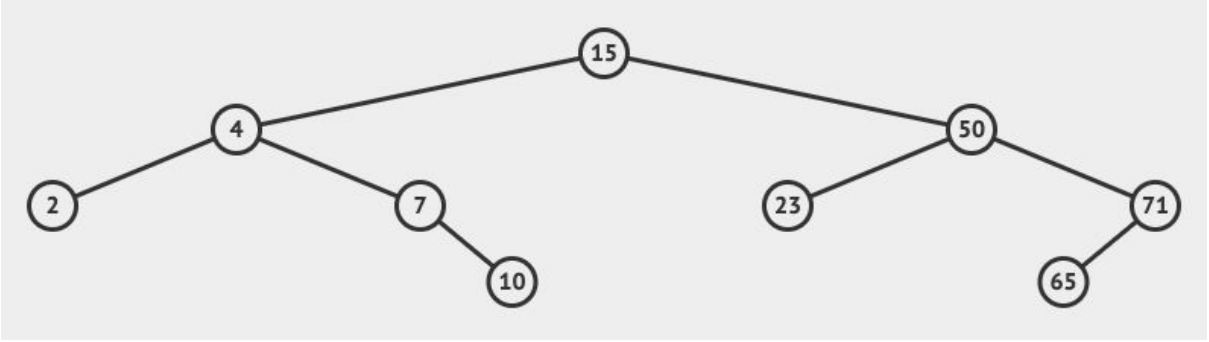
(iii) 1NF [2]

(iv) 2NF [2]

(v) 3NF [2]

(e) What is the relationship between the data structures in Question 2 and a relational database? [4]

4 The following diagram shows the contents with some data inserted.



(a) State two possible insertion orders of data to this BST. [2]

(b) Generalise how data can be inserted to produce a balanced BST. [3]

(c) The BST is to be implemented using a 1D array T. Write pseudocode to show how data can be represented in T with suitable initial values for an empty BST. [4]

(d) Devise a recursive algorithm to insert to this BST. [5]

(e) Devise a recursive algorithm to check if T is a BST. [3]

(f) Convert the recursive algorithm in part (e) to an iterative one using a suitable data structure which you should name and justify. [5]

(g) Devise an algorithm to output the items in T that are within a given range [a, b] inclusive in ascending order. [4]

(h) Devise an algorithm to output the contents of the leaves of T in descending order. [4]

(i) Despite its memory overhead, why is recursion often used in BSTs? [3]

(j) Why is recursion less often used in linked lists? [2]

5 A student came up with the following Python program to implement a linked list data structure:

```
01 class Node:
02   def __init__(self, data):
03     self.data = data
04     self.link = None
05
06 def insert(data):
07   global head
08   if head == None: # empty linked list
09     head = Node(data)
10   else: # insert to front
11     new_node = Node(data)
12     new_node.link = head
13     head = new_node
14
15 def display():
16   global head
17   curr = head
18   while curr:
19     print(curr.data)
20     curr = curr.link
21
22 # main
23 head = None
24 insert("Ali")
25 insert("Tom")
26 insert("Mary")
27 display()
```

(a) What will be the output of line 27? [1]

(b) Comment on the programming paradigms used and identify any potential pitfalls in the above program.. [5]

(c) Why is OOP appropriate in the implementation of data structures such as linked lists?[2]

(d) The above program maintains an unordered linked list. Devise an algorithm to insert to an ordered linked list. [5]

(e) A linked list can be ordered or unordered. Draw a UML class diagram to illustrate the concepts of encapsulation, inheritance and polymorphism. [5]

(f) Explain how polymorphism is applied to the insert() rather than the display() method in this context. [3]

6 On 14 September 2020, it was reported that GrabCar was fined S$10,000 for a 4th user data privacy violation. The Personal Data Protection Commission (PDPC) said the update risked the personal data of 21,541 drivers and passengers, including profile pictures, names and vehicle plate numbers.

GrabCar rolled back the app to the previous version within about 40 minutes and took other remedial action, PDPC said.

On Aug 30, 2019, GrabCar notified the PDPC that profile data of 5,651 GrabHitch drivers was exposed to the risk of unauthorised access by other GrabHitch drivers for a "short period of time on the same day" through the Grab app.

Grab's investigations traced the cause of the breach to a deployment of an update to the app on the same day. The purpose of the update was to address a potential vulnerability discovered within the Grab app.

In PDPC's findings, the application programming interface URL which allowed GrabHitch drivers to access their data, had contained a "userID" portion that could potentially be manipulated to allow access to other drivers' data. According to GrabCar, there was no evidence that this vulnerability was exploited.

To fix the vulnerability, the update removed the "userID" from the URL, which shortened it to a hard-coded "users/profile". However, it failed to take into account the URL-based caching mechanism in the app, which was configured to refresh every 10 seconds.

The mechanism served cached content in response to data requests, so as to reduce the load of direct access to GrabCar's database.

With the update, all URLs in the Grab app ended with "users/profile". Without the "userID" in the URL, which directed data requests to the correct GrabHitch driver's accounts, the caching mechanism could no longer differentiate between drivers.

Thus, the mechanism provided the same data to all GrabHitch drivers for 10 seconds before new data was retrieved from GrabCar's database and cached for the next 10 seconds.

PDPC said GrabCar did not put in place "sufficiently robust processes" to manage changes to its IT system that may put personal data it was processing at risk.

"This was a particularly grave error given that this is the second time the (GrabCar) is making a similar mistake, albeit with respect to a different system," he said.

In a statement in response to Reuters' query, Grab said: "To prevent a recurrence, we have since introduced more robust processes, especially pertaining to our IT environment testing, along with updated governance procedures and an architecture review of our legacy application and source codes."

In 2019, GrabCar was ordered to pay a financial penalty of S$16,000 after it sent out more than 120,000 marketing emails to customers containing the name and mobile phone number of another customer.

The PDPC had found that GrabCar “failed to make reasonable security arrangements” to detect the errors in their database when sending out the emails.

Source: Reuters/CNA/lk

Adapted from
https://www.channelnewsasia.com/news/business/grab-car-hitch-pdpc-personal-data-risk-fine-13108144

(a) For each of the following, suggest two ways in which the data leaks could have been exploited by a malicious hacker with reference to

(i) profile pictures, names and vehicle plate numbers of drivers and passengers [2]

(ii) name and mobile phone number of customers [2]

(b) What could have caused

(i) the URL related data leak?

(ii) the marketing emails related data leak?

(iii) the repeated data leaks?

You should provide a balance of technical and human related reasons. [6]

(c) How can Grab ensure and assure PDPC that sufficiently robust processes have been put in place? [3]