HWA CHONG INSTITUTION
C2 PRELIMINARY EXAMINATION 2025

COMPUTING
Higher 2

15 Sept 2025 **Paper 1 (9569 / 01)** **1400 -- 1700 hrs**

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 answer 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.

1 **(a)** Describe the purpose of Internet Protocol (IP) and state the format of an IPv4 address. [2]

(b) Describe **one** disadvantage of packet switching and how the problem can be handled. [2]

(c) Explain the term Domain Name Server (DNS) and how it works. [5]

Soln
(a) IP is responsible for **routing** packets of data towards their destination.

IPV4 address is **32 bits** or **4 groups of denary numbers** between 0 and 255, separated by dots when written in dotted-decimal notation.

(b) It requires more **time for reassembly** at the destination device as packets may arrive in random order.

To handle this, a **sequential number** is attached to each packet, and efficient algorithms are used to sort the packets into the original order.

(c) DNS server **translates domain name** in human language to **IP addresses** in machine language.

DNS server first checks if the requested domain name is available in its **cache**. If not found in its cache, it sends requests to the **local DNS** by the Internet Service Provider. If still not found, DNS searches from a **hierarchy of distributed database** to locate the domain name. Once found, DNS server **sends the IP address** back to the user.

2 **(a)** An airline website is designed for customers to book air tickets.

(i) State the purpose of data validation and give **one** example of how the website implements it on the customers' date of birth. [2]

(ii) State the purpose of data verification and give **one** example of how the website implements it on the bookings. [2]

(iii) Describe **two** differences between a web application and a native application. [2]

(b) Give **two** purposes of using digital signature and describe how it works. [8]

(c) Describe how a firewall protects computer networks and one limitation of a firewall. [2]

Soln

(a) (i) ensure that the data entered is **sensible and reasonable**

- Range check: year < 2026.
- Format check: all digits, e.g. yyyymmdd
- Length check: 8 characters, e.g. yyymmdd,
- Presence check: not left blank

(ii) ensure the data entered **matches the original resource**.
Proofread of customers' data entered before submission.

(iii)

Web Applications	**Native Applications**
Deployment and maintenance (updates) for a web-based application require deployment on a single set of server machines.	Deployment and any maintenance/patch are done on individual client machines separately.
Web applications can be accessed from anywhere (most locations), so there is no location constraint.	As desktop are confined to a standalone machine, so they can be only accessed from the machines they are deployed in.
Web applications are platform-independent, they can work in different types of platforms with the only requirement of a web browser.	Desktop applications need to be developed separately for different platform machines. (Windows, Linux, Unix, Mac etc)
Web applications are at higher security risks as they are inherently designed to increase accessibility.	Desktop applications, on the other hand, have better authorization and administrators have better control, hence more secure.
Web applications rely heavily on internet connectivity, for their operation.	Desktop applications don't require the internet for their operations. Some applications just require internet connectivity at the time of update.

(b) purpose:

- Authentication: the message was created by the known sender
- Non-repudiation: the sender cannot deny having sent the message
- Integrity: the message is not altered in transit

Process:

- The sender uses a **hash algorithm** to create a **hashed version of the message**
- The sender uses **its private key** to encrypt the hash to the **digital signature**
- Both the message (encrypted or not) and the digital signature are **sent to the receiver**
- The receiver uses the **sender's public key** to decrypt the digital signature back to the sender's version of hash
- The receiver uses the same hash algorithm to **create a new hash** from the received message
- If the two hashes **match**, it means the data is not altered and is sent by the known sender

(c) Firewall monitors and controls all incoming and outgoing network traffic based on a set of security rules to prevent unauthorized access from entering a private network.

Limitations:
Hackers can bypass firewall by inserting malicious attacks inside legitimate programs, for example, emails
Firewall cannot protect against internal attacks, for example, virus in one computer in the network
The setting of firewall may block some legitimate programs

3 A system processes user input and performs encoding and validation using character and number representations.

(a) The ASCII code (in denary) for the character 'A' is 65.

(i) Express 'D' in 7-bit binary. [1]

(ii) Express 'D' as a hexadecimal value. [1]

(iii) State the range of values that are common to both ASCII and Unicode. [1]

(b) A user inputs the number 365, which must be stored in a 7-bit binary format. Explain the problem, and how this could be resolved. [2]

(c) Give **one** example of a software or platform that relies on Unicode and explain its benefit in that context. [2]

Soln

(i)	1000100
(ii)	44
(iii)	Values 0 to 127
(b)	The maximum value for 7-bit unsigned binary is 127. 365 exceeds this and needs 9-bits, so it cannot be stored. This can be resolved by using a larger data type, such as 16-bit binary format.
(c)	Example: WhatsApp or Chrome. Unicode allows the app to support messages in multiple languages and emojis globally.

4 A collectible toy store sells Libaba figurines that come in many series. Each series is released in a different year, and within a series release, multiple toy figurines may be produced. The following unnormalised table is used to track sales of these toys:

`LibabaSale`

SaleID	SaleDate	ToyID	ToyName	SeriesName	ReleaseYear	NumSold
S101	1/4/2025	T001	Polar Pixie	Starborn	2024	1
		T002	Misty Mewling	Fairytails	2025	3
S102	3/4/2025	T003	Nutty Nimlet	Arborealis	2023	1
S103	5/4/2025	T002	Misty Mewling	Fairytails	2025	2
		T004	Jungle Jinx	Arborealis	2023	2

(a) State **one** data redundancy and **one** anomaly issue present in the `LibabaSale` table above. [2]

(b) Identify **one** transitive dependency that exist in the table. [2]

(c) Normalise the `LibabaSales` table into Third Normal Form (3NF) and draw the Entity-Relationship (E-R) diagram showing the relationships between the resulting tables. [3]

(d) A table description can be expressed as:

```
TableName (Attribute1, Attribute2, Attribute3, ...)
```

The primary key is indicated by underlining one or more attributes. Foreign keys are indicated by using a dashed underline.

Write table descriptions for the tables obtained from part **(c)**. [4]

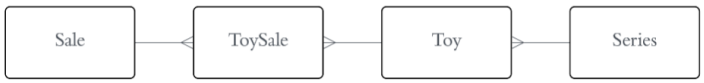
(e) Write an SQL query to display each SeriesName and the total number of Libaba sold, grouped by series and ordered from highest to lowest total. [6]

(f) Explain **one** advantage and **one** disadvantage of archiving sales data. Determine whether weekly archiving is appropriate. [4]

(g) Explain what constitutes personal data under PDPA and give **two** examples of personal data that the toy shop may have of their customers. [2]

(h) What should the toy shop do when attaining personal data from their customers? [2]

Soln

(a)	Repetition of ToyID and ToyName, SeriesName, ReleaseYear. (data redundancy) Update anomalies if a toy's name changes (must update in many rows).
(b)	ReleaseYear is dependent on SeriesName, while SeriesName dependent on ToyID. Hence, ReleaseYear is transitively dependent on ToyID.
(c)	 1 series can have many toys, as seen for the 2 toys from Arborealis. The table shows many sales to many toys which need to be normalised using a joint table.
(d)	Sale (SaleID, SaleDate) ToySale (SaleID, ToyID, NumSold) Toy (ToyID, ToyName, SeriesName) Series (SeriesName, ReleaseYear)
(e)	SELECT Toy.SeriesName, SUM(ToySale.NumSold) AS TotalSold FROM ToySale ts INNER JOIN Toy t ON ts.ToyID = t.ToyID GROUP BY Toy.SeriesName ORDER BY TotalSold DESC;
(f)	Advantages: (state 1) 1. Reduces active database size, improving local system performance. 2. Keeps data safe by storing it separately, often off-site. 3. Enhances security, as archived data can be better protected than on local devices. 4. Cost-effective, since archives can be stored on cheaper long-term storage media. Disadvantages: (state 1) 1. Less accessible, as archived data may not be readily available for reporting or day-to-day use. 2. Extra management effort, as additional processes are needed for indexing, securing, and maintaining archives. 3. Risk of data loss/corruption if archives are not properly maintained. 4. Technology obsolescence, as data stored on outdated media (e.g. tape, CD) may become difficult to access with modern hardware/software. Older file formats may also be incompatible with new software. Weekly archiving is inappropriate as it might be too frequent. Stock-take or reviews might require recent sales data, which requires accessing of archives. Making local back-up weekly would be a better alternative, or annual archive instead.
(g)	Personal data refers to data that can **identify an individual**, whether on its own or when combined with other information. Examples: name, contact number, email address, home/delivery address
(h)	Consent obligation: ask customer for consent Notification obligation: notify why they need the data

5 A car dealership is building a system to manage its vehicle inventory. The company sells Internal Combustion Engine (ICE) vehicle and Electric Vehicles (EV)**.** They want to store and manage information about each car's key specifications, and allow the system to estimate how far the car can travel based on its current fuel or charge level.

For every vehicle, the following data is recorded:

- `BMY`: the brand, model and year of manufacture of the car
- `mileage`: total distance driven in kilometers
- `trip_mileage`: distance driven since the engine was last started in kilometers
- `estimated_distance`: distance which the vehicle can be driven.

All vehicles reset the `trip_mileage` to zero when the engine of the vehicle starts.

For an Internal Combustion Engine vehicle, the following data is recorded:

- `fuel_capacity`: maximum capacity of fuel in liter
- `current_fuel`: current capacity of fuel in liter
- `fuel_consumption`: distance driven in kilometer per liter of fuel

Each internal combustion engine vehicle resets the `trip_mileage` to zero when the engine of the vehicle starts. The vehicle computes the `estimated_distance`, based on the `fuel_consumption` and the `current_fuel` amount of the vehicle, when it is driven.

For an electric vehicle, the following data is recorded:

- `battery_capacity`: maximum energy the battery can store in kWh
- `current_charge`: current battery level as a percentage of total
- `energy_consumption`: energy used in kWh to travel one kilometer

Each electric vehicle resets the `trip_mileage` to zero and check the `current_charge` value to ensure it is above certain threshold when the engine of the vehicle starts. The vehicle computes the `estimated_distance`, based on the `energy_consumption` and the `current_charge` value of the vehicle, when it is driven.

(a) State the purpose of a superclass and subclass. [2]

(b) State the purpose of polymorphism. Give **one** example of polymorphism from the vehicle inventory system. [2]

(c) Draw a class diagram for the described situation, showing:

- any derived classes and inheritance from the base class
- the properties needed in the base and any derived classes
- suitable methods, in each class, to support the system. [7]

(d) State **one** reason for a method of a class to be private. Give **one** example from the vehicle inventory system. [2]

Soln

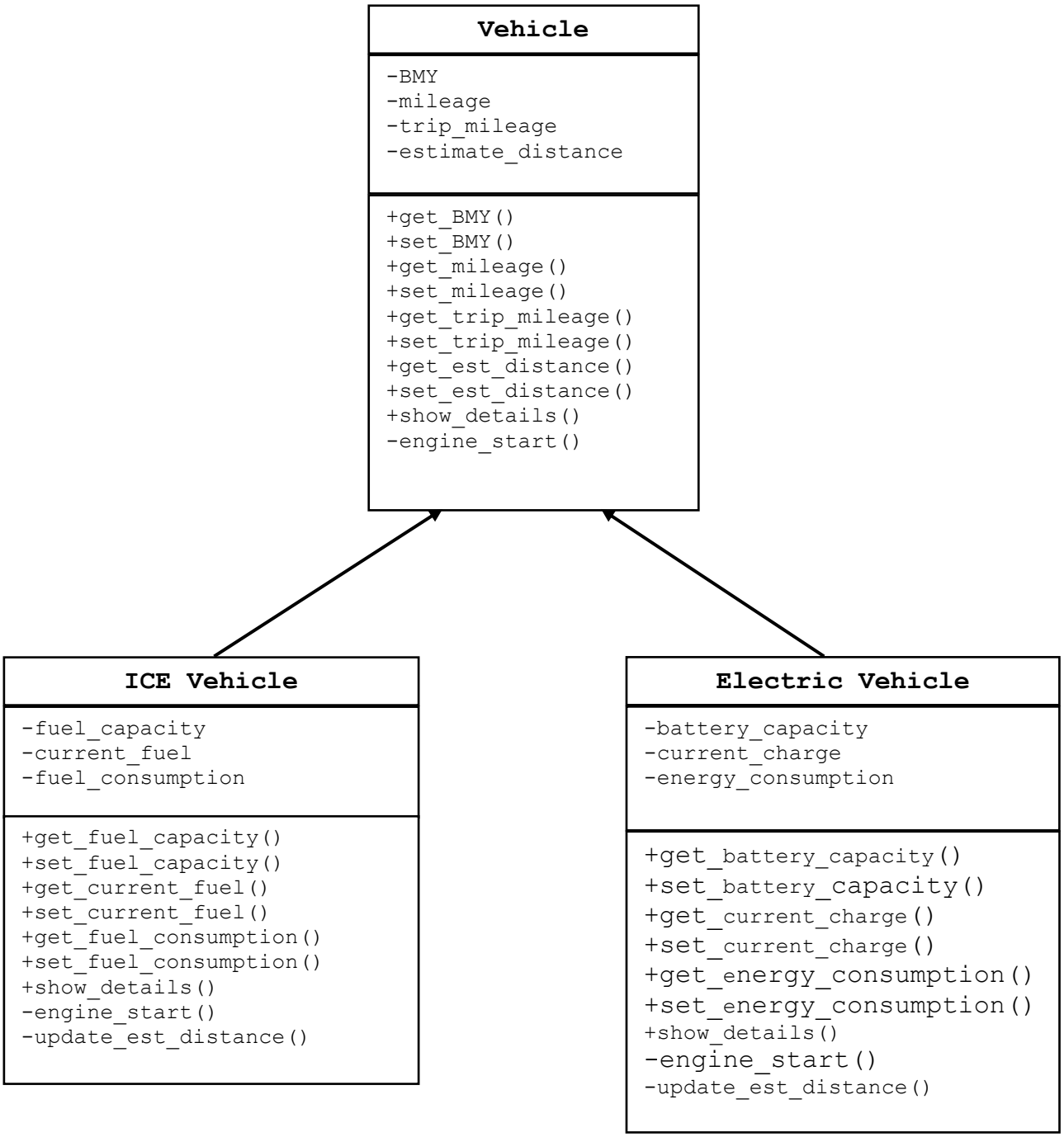
(a) The purpose of a superclass is to define common attributes and behaviors (methods) that can be **shared** by multiple related classes.
The purpose of a subclass is to **inherit** the properties and methods from the superclass and specialize or extend them by adding new attributes or new behaviors to fit more specific needs.

(b) Polymorphism is the object's ability to take different forms. Its purpose is to allow different objects to be treated independently, such that when calling the methods, whose names different objects share, the programme is able to call the correct version of the method depending on the type of object.

Example: Vehicle `engine_start()` method is overridden by ICE `engine_start()` method and EV `engine_start()` method. When `engine_start()` method is invoked, the programme will call the correct `engine_start()` method depending on whether the object is ICE or EV.

(d) Reason : Private method is only for internal use only. It is not meant to be invoked by external code. Example : `update_est_distance()` method is for internal use only. It is not meant to be invoked by external code to update `estimate_distance`.

(c) refer to diagram on the next page


Vehicle
-BMY
-mileage
-trip_mileage
-estimate_distance
+get_BMY()
+set_BMY()
+get_mileage()
+set_mileage()
+get_trip_mileage()
+set_trip_mileage()
+get_est_distance()
+set_est_distance()
+show_details()
-engine_start()
ICE Vehicle
-fuel_capacity
-current_fuel
-fuel_consumption
+get_fuel_capacity()
+set_fuel_capacity()
+get_current_fuel()
+set_current_fuel()
+get_fuel_consumption()
+set_fuel_consumption()
+show_details()
-engine_start()
-update_est_distance()
Electric Vehicle
-battery_capacity
-current_charge
-energy_consumption
+get_battery_capacity()
+set_battery_capacity()
+get_current_charge()
+set_current_charge()
+get_energy_consumption()
+set_energy_consumption()
+show_details()
-engine_start()
-update_est_distance()

6 A hash function and associated hash table are commonly used when finding storage space for a new record and searching for a specific record within a data set.

(a) State **three** features of an effective hash function. [3]

(b) Explain the meaning of a collision in the context of finding storage space for a new record. [1]

(c) A method of collision resolution is chaining. Explain what happens when a new record causes a collision and how the search operation works. [3]

(d) Explain why the chains, used in chaining, are usually not sorted. [2]

Each node in the chain of the hash table contains the following:

- record – the value of the record
- next – reference to the next node in the chain.

The following pseudocode performs the search on the chain.

```
FUNCTION find_value (head, target)
    current ← head
    WHILE current <> NULL
        IF current.record = target THEN
            RETURN current.record
        ENDIF
        current ← current.next
    ENDWHILE
    RETURN NULL
ENDFUNCTION
```

(e) Write in pseudocode the equivalent recursive function `find_value_recursive(node, target)` that performs the same task. [6]

(f) State **one** advantage and disadvantage of using recursion in this context. [2]

Soln

(a)
(i) Uniform distribution of records evenly across all available slots to reduce clustering and collisions
(ii) The same record must always produce the same hash value everytime it is hashed.
(iii) The hash function should be fast and simple to calculate to maintain quick insertions and lookups.

(b) A collision occurs in a hash table when two different records are assigned the same index/slot in the table by the hash function.

(c) Each slot in the hash table stores a reference to a linked list that holds all records that hash to that index. When a collision occurs, the new record is appended to the chain at that slot. During a search, the record is first hashed to find the appropriate index, and then the system traverses the chain at that index to locate the desired record.

(d) Any 2.
(i) Hash tables are used to find data quickly in constant time (O(1)). To keep this speed, new record is added to the start or end of a chain without sorting. Sorting the chain would slow things down, so it's avoided.
(ii)Sorting adds overhead – inserting into a sorted chain requires finding the correct location O(n) in the worst case. Sorting is not helpful unless the chain is long – which a well-designed hash table avoids via good hash function and a low load factor.
(iii)Searching a sorted chain is still O(n) as it is still a linear search unless a different structure is used. Binary is not possible in a linked list as binary search requires random access and linked list only supports sequential access.

(e)

```
FUNCTION find_value_recursive(node, target)
    IF node = NULL THEN
       RETURN NULL
    ENDIF
    IF node.record = target THEN
          RETURN node.record
    ENDIF
       RETURN find_value_recursive(node.next, target)
ENDFUNCTION
```

(f)
(i) Advantage: Simpler and more elegant code for small chains
(ii) Disadvantage: Risk of stack overflow with very long chains due to deep recursion

7 A binary search tree holds the lower-case words in a piece of text in alphabetical order, together with the number of times each word occurs in the array `bst`. Each element of the array contains four values. `leftPtr` and `rightPtr` are integers, `word` is a string and `frequency` is an integer.

leftPtr	**word**	**frequency**	**rightPtr**

The root of the binary search tree is stored in an integer variable, `rootPtr`.
The unused elements of the array are in a free space list that starts from `freePtr`.
The contents of the array `bst` is shown below. –1 represents the null pointer.

Index	**leftPtr**	**word**	**frequency**	**rightPtr**
0	1	the	6	–1
1	2	man	1	5
2	4	chased	3	3
3	–1	cow	2	–1
4	–1	cat	2	–1
5	–1	panther	1	–1
6	7			–1
7	8			–1
8	9			–1
9	–1			–1

rootPtr
0

freePtr
6

An individual value in the array can be accessed in the following format:
`bst[0].leftPtr`

(a) Draw the binary search tree represented by the array `bst`. Give the output of the words performing the pre-order traversal. [3]

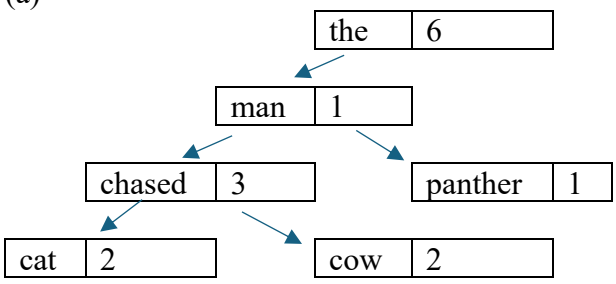
(b) The word, man, is to be removed from the binary search tree. Identify the changes that will be made to any pointers and the contents of the array `bst`. [2]

(c) Write a recursive pseudocode function `Search()` that takes the value of `index` and `target` word, and returns `True` if `target` is found in the binary search tree, and `False` otherwise. [4]

(d) If, after building a complete tree, all the words had to be printed in order of decreasing frequency. Words with the same number of occurrences are printed in reverse alphabetical order. Describe briefly how this could be accomplished. [4]

Soln

(a)



Pre-order output of words: the, man, chased, cat, cow, panther

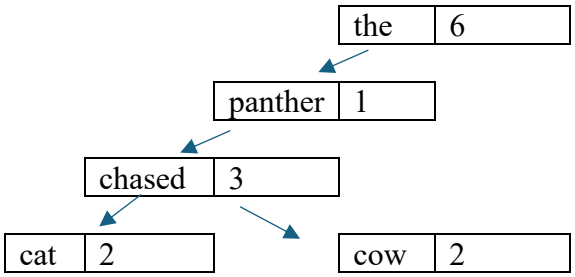
1M tree of words

1M frequency

1M pre-order output

(b) Method 1: replace man with panther

`freePtr`: 1

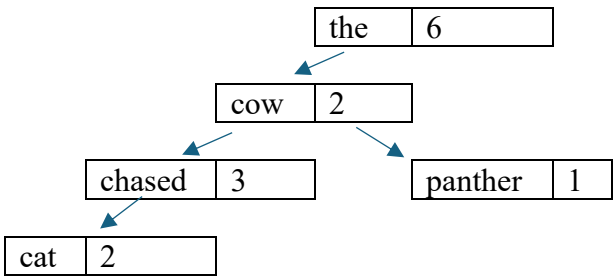


Index	leftPtr	word	frequency	rightPtr
0	5	the	6	–1
1	6	man	1	–1
2	4	chased	3	3
3	–1	cow	2	–1
4	–1	cat	2	–1
5	2	panther	1	–1

1M any two changes
1M all changes

Method 2: replace man with cow

`freePtr`: 1



Index	leftPtr	word	frequency	rightPtr
0	3	the	6	–1
1	3	man	1	–1
2	4	chased	3	–1
3	2	cow	2	5
4	–1	cat	2	–1
5	–1	panther	1	–1

(c)

```
FUNCTION Search (index: INTEGER, target: STRING) RETURNS BOOLEAN
    IF index = –1
        RETURN False
    ELSE IF target < bst[index].word
        RETURN Search(bst[index].leftPtr)
    ELSE IF target > bst[index].word
        RETURN Search(bst[index].rightPtr)
    ELSE
        RETURN True
    ENDIF
ENDFUNCTION
```

1M return False

1M recursion (either)

1M bst[index].ptr/word

1M return True

(d)

Using new BST:

1. **traverse** the existing tree (the order does not matter) to obtain all the pairs of frequency and words
2. Build a new binary search by repeating this step recursively with subtrees starting from the root.
   - If a word's frequency is **less** than current node, insert to the **right** subtree.
   - If a word's frequency is **greater** than current node, insert to the **left** subtree.
   - If a word's frequency is the **same** as the current node, insert to the right subtree if the word is alphabetically smaller than the current node, otherwise into the left subtree.
3. Use **in-order traversal** on the bst to output all the words in the correct order.

1M traverse bst and attempt to build a new data structure

2M insert in descending order of frequency, then word

1M correct traverse

Using new List/Linked List:

- 2D list with each row of frequency and word.
- linked list with each node of frequency and word.

1. traverse the existing tree (the order does not matter) to obtain all the pairs of frequency and words
2. Insert each pair into the new data structure by comparing with the current row/node starting from the first pair:

• If the frequency is less than the current row/node, compare with the next row/node • If the frequency is greater than the current row/node, insert this pair before the current row/node • If the frequency is equal to the current row/node and the word is alphabetically less than the current row/node, compare with the next cell/node • If the frequency is equal to the current row/node and the word is alphabetically greater than the current row/node, insert this pair before the current row/node 3. Print the list from index 0 or the linked list from head.	