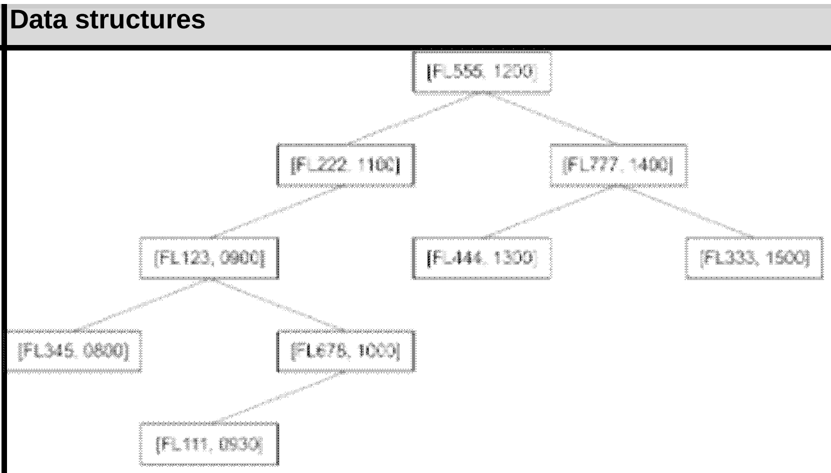
Qn	Task
1	**Decision table**
a	YYY: drop packet YYN: drop packet
	YNY: no reassembly YNN: drop packet
	NYY: first packet NYN: subsequent packet
	NNY: drop packet NNN: subsequent (last) packet
b	YY- or Y-N: drop packet
	N-N: subsequent packet
	Penalties
2	**Sorting algorithms, recursion**
a	[9, 11, 17, 21, 23, 15]
b	Insertion sort
c	If array is largely sorted such that elements do not need to be shifted much, the time complexity can be more efficient / closer to O(n)
	because the inner loop (WHILE) will complete in a small number of iterations (<< n)
d	PROCEDURE P(arr_num: ARRAY, index_of_item: INTEGER) // Correct procedure interface ... ENDPROCEDURE

	// Base case
	IF index_of_item < number_of_items THEN
	// Follow given pattern for iterating index
	...
	ENDIF

	...
	// logically, index_of_item can only represent
	// one pointer
	// makes more sense for it to represent
	// pointer from original code
	// iterate through unsorted items
	key ← arr_num[index_of_item]
	...

	...
	// determine insertion point, carry out insertion
	// Done by shifting elements
	current_item ← index_of_item − 1
	WHILE (arr_num[current_item] > key) AND (current_item > -1) DO
	arr_num[current_item + 1] ← arr_num[current_item]
	current_item ← current_item − 1
	ENDWHILE
	arr_num[current_item + 1] ← key
	...

	...
	// recursive call
	P(arr_num, index_of_item + 1)
	...

3	**Data structures**
ai	 [FL555, 1200] [FL222, 1100] [FL777, 1400] [FL123, 0900] [FL444, 1300] [FL333, 1500] [FL345, 0800] [FL678, 1000] [FL111, 0930]
	[-1 per mistake]
aii	FL345, FL123, FL111, FL678, FL222, FL555, FL444, FL777, FL333
aiii	O(log n)
bi	Starting from the head node, **walk the linkedlist**
	compare each node to [FL253, 0900]. if the **arrival time of the node is later than 0900** (at FL888), FL253 is inserted ...
	by **linking the pointer of the previous node** to FL253, and **the pointer of FL253 to the target/next node**
bii	O(n)
ci	Time complexity determines **how execution time increases with the input size**/data structure size
	Hence, the speed of insertion cannot be determined from time complexity alone
cii	**Insertion into a BST has lower time complexity than insertion into a linked list**, thus the **execution time grows more slowly** as the data structure grows (*Time complexity of BST & linkdlist insertoion need not be stated again in student answer since it is already stated in aiii and bii*)
	If the airport **expects many more international flight arrivals compared to domestic flights**, the BST would be a more appropriate choice of data structure [context]
4	**Data representation, data validation & verification**
a	Unicode is able to represent characters **beyond the ASCII range** ...
	... which allows **international characters/other symbols** to be used **for expense description** [context]
bi	Format check / range check / presence check; any 2
bii	Double entry / proofreading

ci	4 bytes
cii	7 bytes / 8 bytes
d	advantage: human-readable
	disadvantage: more bytes/storage required / not easily used in calculations, requiring conversion
ei	BFF1: (11 * 16^3) + (15 * 16^2) + (15 * 16^1) + + (1 * 16^0) = **49137**
eii	1 --> 0001 A --> 1010 F --> 1111 0 --> 0000 1AF0 = **0001 1010 1111 0000**
5	**Object-oriented programming**
a	Encapsulation: Project - project_ID - project_start_date - project_leader
	appropriate getters and setters (all classes)
	Inheritance: SoftwareProject inherits Project
	- programming_language - base_development_budget - testing_state
	Inheritance: BespokeSoftwareProject inherits SoftwareProject
	- client_name - delivery_date - maintenance_fee (*may be implemented as method*)
	method to determine development budget method to determine annual maintenance fee
	Inheritance: OfftheshelfSoftwareProject inherits Project
	- project title - anticipated_retail_price (*may be implemented as method*) - sales_forecast (*may be implemented as method*)
	method to determine anticipated retail price
	Inheritance: InstallationProject inherits Project
	- client_company - installation_cost - manpower_cost (*may be implemented as method*)
b	A class is a blueprint/template for an object
	Objects are instantiated from classes / A class is used to create an object
ci	Encapsulation involves the **bundling of data and methods that act on the data into an object** (*not accepted: information hiding; already given in cii*)
cii	Encapsulation involves **restricting access to private attributes** using public methods

	thus **separating interface from implementation** / hiding private information by **exposing only allowed public methods**
di	Inheritance enables child classes to access public methods from parent classes, **promoting code reuse**
dii	Inheritance enables classes to share methods/code, **reducing code duplication** [point]
	... code only needs to be updated in one place / there is lower chance of duplicated code being edited out of sync ... [elaboration]
	... making code maintenance easier / reducing likelihood of bugs [context]
ei	Polymorphism **promotes code generalisation**
eii	Polymorphism **reduces the complexity of code** / the need for conditional handling of objects [advantage]
	Software objects/instances are expected to have a <name of getter/setter>, which can be invoked **regardless of type of software object** [context/example]
6	**SQL database, data normalisation, network security, data privacy**
a	**Quantity column depends only on ItemID and CartID** but not on other key columns (partial dependence)
	partial dependency **violates 2NF**
	3NF requires 2NF hence Cart table is not in 2NF
b	data in both tables come from Cart table, all columns in Cart table represented [1]
	both tables are 3NF [1]
	appropriate identification of PK [1]
c	Entities: Customer, Item, Cart
	Cart <--n--n--> Item
	Customer <--1--n--> Cart
d	**Voucher entity/table** is required (with an id PK)
	Cart requires a **VoucherID column (FK)** (*Qn implies Cart <--1--1--> Voucher relationship*)
e	SELECT SUM(Price * Quantity) (*Correct columns* [1], *correct function use* [1])

	`FROM Cart`
	`INNER JOIN Item`
	`ON Cart.ItemID = Item.ItemID`

	`WHERE Cart.CustID = 6`
	Correct syntax use (single-quoted literals, correct operators, etc)
fi	Malware / unauthorised database queries / leaked database credentials / (*any suitable vulnerability within syllabus*)

	Appropriate measure to address above vulnerability
fii	Digital certificates are used to **authenticate the Toppee server** to customers [purpose]
	This enables clients/customers to **trust the public key** (for Toppee) ... [elaboration]
	which can then be used to improve security through **encryption** / **identifying phishing sites** / <other suitable points involving correct use of public key> [context]
g	Relevant PDPA obligation
	Appropriate measure to implement obligation
	Relevant PDPA obligation
	Appropriate measure to implement obligation
7	**Application design and useability, ethics**
a	Native applications are installed on the target device, while web applications are accessed through (sending requests to a) web server / native applications require updating through an update server / app store, while web applications are updated on the server so clients always get the latest version
	Native applications have a wider set of capabilities (if permission is granted by the user), while native applications have a narrower set of capabilities (*with appropriate examples/elaboration: access to file storage, other hardware devices, etc*)
b	Justified: company has **better control over the software** and services provided ... [supporting justification]
	because customers/clients will only be able to access the version that is used on the server / if a native app is used, the company cannot force users to stay updated to the latest version / the company cannot fix critical vulnerabilities/bugs and ensure all users receive fixed version / <other relevant elaboration>
c	Principle correctly identified/described
	Appropriate application of principle to improve user experience **for banking**
d	Not aligned with code of ethics ...
	Integrity: Company did not communicate consequences of tight deadline to customers/stakeholders
	Responsibility: Company did not take required actions responsibly
	Professionalism: Company did not carry out testing for vulnerabilities where important financial data is involved, where this is reasonably expected behaviour for released software