THIS PAPER IS NOT TO BE REMOVED FROM THE EXAMINATION HALLS

UNIVERSITY OF LONDON **CO2226 ZA**

BSc Examination

COMPUTING AND INFORMATION SYSTEMS, CREATIVE COMPUTING and COMBINED DEGREE SCHEME

Software engineering, algorithm design and analysis

Friday 11 May 2018: 14.30 – 17.30

Time allowed: 3 hours

This paper is in two parts: Part A and Part B. There are a total of **THREE** questions in each part. You should answer **TWO** questions from Part A and **TWO** questions from Part B.

Full marks will be awarded for complete answers to a total of **FOUR** questions, **TWO** from Part A and **TWO** from Part B. The mark for each part of a question is indicated at the end of the part in [.] brackets.

Only your first **TWO** answers from Part A and first **TWO** answers from Part B, in the order they appear in your answer book, will be marked.

There are 100 marks available on this paper.

A handheld calculator may be used when answering questions on the paper but it must not be pre-programmed or able to display graphics, text or algebraic equations. The make and type of machine must be stated clearly on the front cover of the answer book.

Part A

Question 1

a) You are responsible for the verification and validation of a new design that was proposed to your company, and your manager has given you the task to verify class diagrams against state machine diagrams. Briefly explain how you would go about this and what you would be looking out for and comparing.

[5]

b) LogItNow.com is a new online company providing software for running helpdesks. The user will fill in a registration form where their contact details (*e.g.* first and last name, email address, mobile phone number) are captured and they are invited to select a username and a password.

- Once the user is logged in, they can create a new ticket. The ticket will have an owner (by default the user who logs it), and a secondary contact address so that if it is an issue affecting other users as well, they can also be notified when a resolution is reached.
- Once the ticket is logged, depending on its nature, it will be assigned to one of the teams responsible for looking after the various technical areas (*e.g.* Windows, Unix, Networking, Desktop build, Apple, Servers, *etc.*).
- Once assigned to a team, it will be further assigned to an individual member of the team. Every staff member will have an account with the system – the same details are kept as for a regular user, but on top of this we record staff expertise (*e.g.* E-Learning, Unix) and its level (*e.g.* intermediate, advanced).
- The ticket may be passed from one team to another, as the first staff member who has contact with the user will ask further questions to determine if it needs to be passed on to a more specialised team. Every interaction between a staff member and the customer is logged on the system, and any staff member as well as the customer should be able to access it.
- In some cases, external parts will be required for the repair of a device, and they will have to be purchased by the user before the specialist teams can carry out any further work. If this is the case, the customer will be presented with the parts that need to be purchased, and the bill for these. Payment of at least half the amount must be made before more work can be done. The rest of it will be payable on collection; the customer cannot pick up their device until payment has been made in full.
- The company does not process the payment, but passes the information to a payment gateway that does it for them. They do, however, keep information about the payment, such as means of payment, end result, authorisation code if they need to trace back any details, *etc.*
- The software allows administrators to mark certain users as VIPs (usually senior management and their executive assistants, as well as those responsible for running critical software for the company).

- The customer can update their ticket at any time by logging on to the portal. This update is visible to the team working on the ticket when they check the list as the status of the ticket would have changed. They can select the individual ticket to view the details. In the case of VIP customers, the update will be emailed to the person assigned to the ticket or, if no assignment has been made yet, to all engineers in the team.
- Sometimes, if a major system goes down (*e.g.* internet connection), a number of tickets will be generated with the same subject and, once the connection issues are resolved, will be closed with the same resolution. The LogItNow.com software provides the option to link those incidents together into a "broadcast", where the same resolution is sent out to all users. There is also the option of using information from the ticket, *e.g.* the first name of the customer can be used, so that the closing ticket will be addressed to Dear Alison).
- The company makes use of a third-party mass mailing system to notify users of any issues with the LogItNow.com software, *e.g.* downtime due to scheduled maintenance or unscheduled interruptions. These are ad-hoc messages that go out whenever needed and administrators use this feature to notify users of any upcoming events that will prevent them from using the software.
- At the end of every month, the system generates a number of reports to be made available to management to allow them to make more informed decisions about processes and procedures, and identify major bottlenecks in their IT processes. These reports include the number of tickets generated during the month, as well as what categories these were logged under (*e.g.* Software Installation, Wireless Network, *etc.*). The report also offers a breakdown for each analyst, as well as the number of tickets raised, open and resolved in each category.

Develop a **class** diagram for the above scenario using the appropriate naming conventions (*e.g.* class names starting with a capital letter) and suggest class attributes with name and type, as well as methods with name and return type. Illustrate associations, aggregations, and generalisation relationships between the objects.

[20]

Question 2

a) How would an extension use case differ from an alternative flow in a use case diagram? Briefly explain your answer with the use of examples.

[5]

b) Prepare a use case diagram to accommodate the specification of the LogItNow.com system described in **Question 1**. Your answer should include all use cases, actors and any associations between different use cases, different actors or use cases and actors. Please state any assumptions you make about the system.

[20]

Question 3

a) Briefly describe the concept of polymorphism in Object-Oriented Design and explain how and why it can be useful in designing systems; use an example to illustrate your answer.

[5]

b) Create a state machine diagram describing the behaviour of a ticket object described in **Question 1**. The following rules apply:

- The ticket object becomes **New** once the customer selects the New Ticket option on the website;
- It moves to **In Preparation** while the customer is adding any relevant information that will help with troubleshooting;
- Once the customer has added the description and any other information about the incident and submits the ticket, it becomes **Submitted**;
- After submission of the ticket, an acknowledgement email is sent out to the customer and the ticket becomes **Acknowledged**;
- One of the HelpDesk team is meant to take this ticket within three days of it becoming **New** – if this does not happen for whatever reason (*e.g.* excessive workload), then the system assigns it to a HelpDesk team member;
- The team member allocated the ticket might need to ask for additional information to resolve the problem – in this case, the ticket will be **Waiting For Customer Response**;
- When the customer replies, the ticket will change to **Customer Replied** so it is easy for the technician to review;
- If the ticket needs to be moved to another queue, then the ticket becomes **Reassigned**;
- In the case that external parts need to be ordered, the ticket will go into **Awaiting Payment** until payment is made, in which case it will become **Payment Accepted**;
- Once the issue is resolved, the ticket becomes **Completed** and an invitation, open for one week, is sent out to the customer for feedback – during this time, the ticket is **Awaiting Feedback**;
- If feedback is received within a week, then the ticket moves to **Feedback Received** and then is immediately closed, otherwise it goes into **Closed** after a week of inactivity;
- One month after the ticket is closed, it becomes **Archived**.

Design the state machine diagram using state transitions and labels with three parts.

[20]

Part B

Question 4

This question focuses on computability and intractability.

a) Give the **TWO** main reasons why an abstract computational model is used in the study of computability.

[4]

b) Explain the concept of an undecidable problem with an example. Contrast this with the concept of intractability.

[5]

c) The Travelling Salesman Problem (TSP) is a combinatorial optimisation problem. There are N cities and each city has to be visited **once only** before returning to the origin (City A in this case).

All cities are connected to each other but not themselves. The intercity distances are symmetric, in other words the distance from city B to C will **always be the same** as from C to B.

The aim of the TSP is to find the cyclic path (*e.g.* A-C-D-B-A) that minimises the distance travelled.

Given the above description of the optimisation problem:

i) Give the corresponding decision problem for the TSP.
ii) Explain what it means for the TSP decision problem to be NP-complete.

[6]

d) The adjacency matrix with intercity distances for a TSP instance is given below.

		(To) A	B	C	D
	A	0	1	4	2
(From)	B	1	0	2	3
	C	4	2	0	3
	D	2	3	3	0

i) Draw a search tree using depth first search, clearly marking the optimal path(s).
ii) Propose and apply a greedy algorithm to this problem.

[6]

e) Briefly compare the outcomes of the two algorithms above. Can this result be expected in all cases? State your reason(s).

[4]

Question 5

This question focuses on stack and queue data structures.

a) Explain briefly what a queue is. Use an example to illustrate your answer.

[5]

b) Describe how a queue can be implemented if we use the data structure of an array for this purpose. Provide the implementation of the **enqueue** method for this type of implementation.

[7]

c) What is a stack? Provide **ONE** way in which a stack can be implemented.

[6]

d) Illustrate how a queue with the elements a, c, b, e, d, f will be constructed and explain the process for removing b from the queue.

[7]

Question 6

This question focuses on recursion, divide and conquer and dynamic programming.

a) Describe **TWO** benefits of using recursive algorithms over their iterative equivalents.

[4]

b) Provide a **recursive** algorithm which takes two arrays as arguments (A and B) and appends array B to array A by forming array C; so the function will look like **append(A, B, C)** where C would be the output of the algorithm and A and B the input arrays; you do not need to worry about implementation details, only the pseudocode is needed.

[6]

c) Briefly explain how the Merge Sort algorithm works in terms of the divide and conquer approach.

[3]

d) Demonstrate the divide and conquer algorithm by sorting the following lists in **alphabetical order** by using Merge Sort (please make sure that you show every step of the process):

(d, b, g, f) and (a, h, e, c).

[6]

e) Describe the main steps involved when using dynamic programming.

[3]

f) Explain using an example, how dynamic programming is different from recursion.

[3]

END OF PAPER