Student Number:	**Seat Number:**
Student Name:	**Module Group:**

NGEE ANN POLYTECHNIC

Data Structures & Algorithms (DSA)

Year 2/3 (2020/21), Semester 4/6

SCHOOL OF INFOCOMM TECHNOLOGY
Diploma in Cybersecurity & Digital Forensics
Diploma in Information Technology

TEST 1 Sample paper

Weightage: 20%

INSTRUCTIONS TO CANDIDATES:

1. This paper consists of 5 pages including this cover page. Check carefully to make sure your set is complete.

2. There are **3** questions in this paper. Answer ALL questions.

3. Write your answers in the SOLUTION document provided online.

4. Download the files from MeL as instructed by your tutor.

5. Computer laptops/notebooks (with no internet access) are ALLOWED. Language translators are NOT allowed.

There are 3 questions. Answer ALL questions (100 marks).

Question 1 (40 marks)

A pointer-based List ADT (Linked list) is given in the folder Q1.

(a) You are tasked to implement a function `void List::sortedInsert(Itemtype& item)` for the List ADT, that inserts a given new element, `item`, into the correct position in the list sorted in ascending order.

For example, if the linked list has the following values:

0 2 3 4 6 8 9

using the `sortedInsert()` function to insert 5 would result in the following linked list:

0 2 3 4 5 6 8 9

Duplicate values are allowed in the list.

(15 marks)

(b) Given two lists sorted in increasing order, write a `Node* sortedMerge(Node*& list1, Node*& list2)` function that returns a new list representing the merged result of the two lists pointed by `list1` and `list2`, while still maintaining the sorted order. The new list should have its own memory and the original lists should not be changed.

For example, when the function is applied to following two lists:

List1: 24 25 70 80 90
List2: 20 24 70 85 90

Resultant List: 20 24 24 70 70 80 85 90

(15 marks)

(c) Explain the time performance of `List::sortedInsert()` and `List::sortedMerge()` implemented in 1(a) and 1(b) respectively.

(10 marks)

Question 2 (35 marks)

The customer service centre of an IT company has decided to implement a queue system to process the service requests of its customers. The specification of the queue is given below. A queue ADT is given in folder Q2.

(a) Implement a function, `void registerCustomer(Queue& serviceQueue, int& queueNumber)`, to allow the staff to register a customer. The function should prompt the customer to enter his/her name and add a customer to the queue, `serviceQueue` with the `queueNumber`, followed by incrementing the `queueNumber` for next potential customer.

(10 marks)

(b) Implement a function, `void nextCustomer(Queue& serviceQueue),` to allow the staff to get the next customer in the queue to serve. The function should remove the customer object from the queue and display the queue number of that customer.

(5 marks)

(c) Implement a function, `void displayCount(Queue& serviceQueue)`,to allow the staff to compute and display the total number of customers left in the queue

(15 marks)

(d) In some cases, when the customer service queues become too long, the service centre will encourage some later service requests to be removed and processed on another separate appointment. Another possible scenario is due to special conditions, some incoming customer requests can be processed first.

Explain what possible changes can be made to the queue ADT to cater for such a process.

(5 marks)

Question 3 (25 marks)

In mathematics, the Greatest Common Divisor (GCD) of two non-zero integers is the largest positive integer that divides the numbers without a remainder. For example, `GCD(8, 12)` is integer 4.

Explanation,
8 is divisible by 1, 2, **4**, 8
12 is divisible by 1, 2, **4**, 6, 12

Hence the GCD of 8 and 12 is 4 (largest positive integer divisible by 8 or 12).

To achieve this, we can make use of % or modulo operator, iterate it till the remainder is 0. Below shows the possible GCD Iterations for two sample runs of the function.

Example 1 iteration of `GCD (8, 12)` is 4

1st iteration	*x =8*	*y = 12*	*x % y = 8*		
2nd iteration		*x = 12*	*y = 8*	*x % y = 4*	
3rd iteration			*x = 8*	***y = 4 (found)***	*x % y = 0*

After 3 iterations, if x % y = 0, the GCD value is y.

Example 2 iteration of `GCD (60, 24)` is 12

1st iteration	*x = 60*	*y = 24*	*x % y = 12*	
2nd iteration		*x = 24*	***y = 12 (found)***	*x % y = 0*

After 2 iterations, if x % y = 0, the GCD value is y.

Some sample data of GCD value is shown as follows:

x	***y***	***GCD(x, y)***
24	54	6
8	12	4
24	60	12
41	27	1

(c) With reference to Figure 3-1, show with the aid of a similar diagram, the iteration(s) of `GCD (24, 54)`. *(You need not write any code)*

(5 marks)

(d) Write a recursive `GCD(x, y)` function. State clearly the base case.

(10 marks)

(e) Write an iterative `GCD(x, y)` function.

(10 marks)

– End of Paper –