UNIVERSITY OF BOLTON
BSc COMPUTING
COURSEWORK SUBMISSION FORM

Student/Centre to complete:

SURNAME/FAMILY NAME: …………………………… FORENAMES: ……………………………….

BOLTON STUDENT ID: ……………………….. EMAIL: …..……………………………………

DATE OF SUBMISSION: …………………………………..……………

MODULE NO./TITLE:…CPU5008 Data Structures and Algorithms…………………………

TUTOR'S NAME:………Abdul Razak…………………..... ……………………

COURSEWORK TITLE: Assignment 1 (part 2): Queues and Linked lists.

Please state if this is your FIRST submission OR REFERRED/DEFERRED submission
OR a REPEAT submission?

………………..……………………………………………………………………………

Declaration
I hereby declare that this work is my own work. I understand that if I am suspected of plagiarism or another form of cheating, my work be referred to Academic Registrar and/or the Board of Examiners, which may result in me being expelled from the programme. I understand once I submit this work, it will automatically belong to the University of Bolton.

Academic staff to complete:

Feedback: ………………………………………………………………………………………

…………………………………………………………………………………….......................

…………………………………………………………………………………………………

Date Issued:…..……*week 7*……….. Hand-In Date:….(**15 November 2016 @ 12.00 noon**)

Other Relevant Date e.g. Demonstration …**15 November 2016 during the practical session.**

Received: On Time ☐ Late ☐

Mark awarded: ………..% Do not apply mark penalty unless the work was submitted late.

Assessors Name: …A. Razak…….………… Signature:...

Date:………………………….

Degree Conversions	A: 70-100%	B: 60-69%	C: 50-59%	D: 40-49%	F: 0-39%
HND Conversions	Pass: 40-49%	Merit: 50-66%	Distinction: 67-100%		

Late submission
For late submission see student handbook:
http://www.bolton.ac.uk/Quality/SE/Student-Handbook/Home.aspx

Creative Technologies	
Course / Programme:	**Computing**
Module name and code:	**Data Structures and Algorithms CPU5008**
Tutor:	**Abdul Razak**
Assignment Number:	**One (part 2)**
Assignment Title:	**Queues and Linked Lists**
Issue Date:	**1st November 2016**
Submission Deadline:	**15th November 2016 @ 12.00 noon**

Learning Outcomes:

LO 2. Establish and apply queues and priority queues.

Assignment:

Using Linked lists and Queues data structures.

HE5 – Assessment is set appropriate to level HE5.

Specific Assessment Criteria

- Have solved several problem tasks using the Java language.
- Have shown the ability to decompose a problem and have designed suitable class structures to effect a solution to a given problem.
- Have demonstrated the ability to complete tasks from blank solutions in order to achieve a goal.

Grading

A percentage mark will be provided as feedback. Grading is as follows:

A:	70-100%
B:	60-69%
C:	50-59%
D:	40-49%

Marks below 40% will be classed as fail.

Assignment 1 part 2 (Queues and Linked lists)

Part 1 (30 marks)

Create a new Java project called **Queue part1**. In the project create the following 2 classes called FlightQueue and Flight which could be part of a flight controller type application

Class name: FlightQueue (Attributes)	
private LinkedList flights	
Class name: FlightQueue (Public methods)	
void joinQueue(Flight f)	Inserts a flight into the queue with no priority order.
void landFlight()	lands flight (i.e. removes flight from the queue)
int size()	returns number of aircraft in the queue
void clear()	lands all flights leaving the queue empty. You must use an Iterator in a loop.
void display()	displays a list of flights in the queue. You must use a for-each loop.

Class name: Flight (Attributes)	
private String flightID	e.g. BA378
private int priority	1 – 9 incl. (1 = lowest & 9 highest priority)

Both classes must have

1. a default constructor that initialises its attributes to sensible values.

The Flight class must also have

2. a parameter constructor that sets all the attribute values based on the parameter values.
3. a setter method for each attribute.
4. a getter method for each attribute.
5. a toString method to return a suitably formatted string of the attribute values.

Create a class called **FlightTest001** that simply creates a number of flights and joins them to the queue *without* maintaining priority order.

Provide suitable code to fully test your classes. You must use type-safe code with parameterised types.

Q. What would be the advantages and disadvantages if FlightQueue were to implement the Queue interface?

Part 2 Implementing a Priority Queue. (50 marks)

Create a new Java project called **Queue Part2**

We now require a priority queue (as well as the 'normal' queue in part 1) that will always land flights in priority order.

Create an abstract class called AbstractFlightQueue. Then create three concrete subclasses – PriorityFlightQueue1, PriorityFlightQueue2, and NormalFlightQueue.

Q. Which method(s) should be abstract in AbstractFlightQueue and which should remain concrete? Explain your choice.

In PriorityFlightQueue1 you should simply add the new flight to the back of the queue, then sort the whole queue with the Collections sort method.

In PriorityFlightQueue2 you should add by iterating through the sorted loop to find the correct point at which to insert the flight into the linked list.

Create a class called **FlightTest002** that re-runs the previous test, but this time with all three types of FlightQueue.

Q. Explain the algorithm that you implemented for PriorityFlightQueue2, In particular evaluate the performance and the order of the algorithm using Big O notation.

Part 3 Implementing the java.util.PriorityQueue. (20 marks)

Create a new class in the project called **JavaPriorityFlightQueue**. Instead of using a linked list as the container use the concrete java.util.PriorityQueue.

Create a class called **FlightTest003** that re-runs the previous test but utilises the JavaPriorityFlightQueue class instead.

Q. Does the flight queue appear sorted when you print it? If not why not, see the JavaDocs for a hint? Is it evident that queue was sorted when the flights are landed?

IMPORTANT (For all parts)
You should add JavaDoc comments to explain the purpose of the new methods added to the each class. In addition to the JavaDoc comments you should add additional normal style comments where appropriate.

What you must submit

Your must submit the assignment in class or to the student centre by 4pm on the Tuesday of the week shown page 1 of this assignment brief.

Your submission must include:

1) This assignment brief completed and signed on the front page, also complete the table below.
2) Your answers (and discussion if appropriate) to the questions highlighted in yellow
3) complete UML diagram(s)
4) Printouts of the code for the classes:
    - AbstractFlightQueue
    - PriorityFlightQueue1

- PriorityFlightQueue2
- NormalFlightQueue
- JavaPriorityFlightQueue
- Flight
- FlightQueue

5) The entire Eclipse project on disk (copy the entire project folder from the Eclipse workspace but **do not** change the folder name afterwards)
6) The disc should be clearly identified with your name, student ID, module's name and assignment title and secured with the above documentation.

In this assignment I have achieved the following objectives			
Tick appropriate box NA – not attempted : Part – part completed : Full – fully completed	NA	Part	Full
Part 1			
Part 2			
Part 3			

Assessment Criteria

The mark you get is based on

- the quality of the code provided
  - well structured
  - good use of identifier names
  - good use of comments (both JavaDoc and normal comments)
- clear concise algorithms that uses appropriate Java programming construct

NOTE: You need to demonstrate your programs and explain how they work during the practical session. No demonstration means zero marks.