IFN564 assignment

Key information

Release date: Thursday, October 6th
Submission date: Friday, November 4th, 11.59pm

Submissions will be through Blackboard, using two links:
- A Turnitin link to submit your report
- A standard Blackboard link to submit an archive containing your code

Only documents submitted through the appropriate links will be considered for marking.

As in all units, this assignment is subject to QUT's stringent late submission policy. Extensions are processed centrally, without any input from the teaching team. Please refer to the Faculty student services pages for details.

room capacity: fix => array
queue for waiting customer
customer list: linked list (not fix)
10 fav movies: stack (latest)
receiving movies: store in collection (latest by stack)

1 screen, 1 film per day

Problem description

Should based on automated testing

Imagine you are running a small independent cinema. You have a single screening room, and show a single movie each day, for which customers can buy tickets. There is no reserved seating: customers just buy a ticket (within room capacity) [*10*] and can seat anywhere. There is no advance booking system. If there is no ticket available for the next session, customers just have to come back another day. Customers like the informal character of your cinema, and do not seem to mind the lack of a booking option.

When customers arrive to the cinema, they wait in line to be served. Once their turn has arrived, you check that they exist in the system. If not, a new profile is created. Then, a ticket is sold (if seats are still available). If the movie is sold out, all customers are told to leave.

how many seat available?

fix room capacity   *fav movie is never burn out*

You receive new movies at random times, and store them in your collection. Because the schedule for new arrivals is quite irregular, you like to always screen the most recent movie. After the movie has been shown once, it is discarded. In case no new movie has arrived in a while, you always keep a copy of your favourite movie somewhere safe, as a backup.

You need your system to allow you to add or remove customers and their information (name, phone number, payment method). You also store how many screenings they have attended, because after 10 screenings they get a free ticket.

Task 1

1. Identify which data structures are appropriate to develop this system, and clearly justify your choice.

2. Identify the algorithms you need to implement to manipulate these structures and to develop the overall system.

   focus on here

3. Analyse these algorithms in terms of their efficiency. If their best-, worst- and average-case behaviours are different, focus your analysis on the best-case and worst-case efficiency.

Task 2

1. Implement the data structures and algorithms identified in Task 1. As discussed in class, **you need to implement your own structures** (apart from the array of course). For instance, if you use a linked list, you <u>cannot</u> use the existing linked list structure in your programming language. You need to implement your own.

2. Experimentally test the correctness of your implementation. Make sure to justify why your tests are appropriate to establish correctness.

3. Experimentally test the efficiency of the algorithms you implemented.

   describe eficiency for algorithm once time

Assessment criteria

pseudo code in report

Task 1 (20 marks)

Criteria		Marks	Excellent	Satisfactory	Unsatisfactory
Data structures	Choice	2	All the data structures are appropriate for this system	Some data structures are sub-optimal	The data structures are not suitable for this system
	Justification *can replace by binary tree*	2	Clear and valid justification	Some justification	No (or incorrect) justification
Algorithms	Adding a customer	2	This is the right algorithm to use for this task, and it is presented without errors	The algorithm has some minor errors, or may not be the best option for this task	The algorithm is incorrect, or is grossly inefficient
	Removing a customer	2			
	Receiving a new movie	2			
	Scheduling the next movie	2			
check how many movies they seen	Serving customers	2			
Efficiency	Basic operation and input size	1	Basic operation and input size clearly identified, well justified, and suitable	Basic operation and input size appropriate but poorly justified	Basic operation and input size inappropriate
	Reflection on best, worst, and average cases	1.5	Appropriate and well justified reflection	Correct reflection	Incorrect reflection
Lecturer 1.3	Exact efficiency function(s)	2	Correct results, with all steps explained	Correct results, lacking justification	Incorrect results
	Efficiency class(es)	1.5	Correct results	Incorrect results (but correct with respect to an incorrect result for the efficiency function)	Incorrect results

Task 2 (20 marks)

Criteria		Marks	Excellent	Satisfactory	Unsatisfactory
Implementation	Code	4	The programs implement the algorithms faithfully	There are unexplained differences between the algorithms and their implementations that could cast doubt on the validity of the experiments	The implementations are incomplete, or differ from the given algorithms in a way which invalidates the experiments
	Documentation	2	The implementations are either self- evident or are explained clearly, succinctly, and accurately	Some parts of the implementation are not explained clearly	Lack of clear documentation or adequate comments
Correctness	Testing strategy	3	The functional correctness of the programs was tested or verified in a clear and appropriate way, including all relevant cases	The way in which the programs were shown to work correctly lacks some minor detail or fails to consider some important input cases	The programs' functional correctness is not demonstrated or verified adequately
	Test results	3	Results from the tests are presently clearly and easily reproducible without manual input	The presentation of the test results is somewhat unclear or incomplete	Inadequate presentation of the test results
Efficiency	Testing strategy	3	The efficiency of the programs is measured in an accurate way, and the input size range is appropriate	Some minor imprecisions in the measurements or in the choice of the input size range	The programs' efficiency is not tested adequately
	Test results	5	Excellent presentation of the results. It is clear how many data points contributed to the graphs of results and how many tests contributed to each data point. Experiments produced clear trends and the results are explained clearly	Some issues with the presentation of the results. For instance, graphs of results do not clearly show individual data points or it is not clear how many tests contributed to the results, or the discussion of the results lacks details	It is impossible to tell how many tests or experiments contributed to the final results. The results are insufficient or too inaccurate to allow convincing conclusions to be drawn from the experiments

(handwritten annotations:)

For example:
correctness: checking if add customer into array, is customer add correct
efficiency: check how long to add that customer into array

put this on in report

test for correctness
test for efficiency

testing not depend on input
=> automated testing

testing: wide range value, avg, small
testing in special case, worst, best, avg

automation testing

Academic honesty

This is an individual assignment.

If there is any concern about the originality of your work, you may be asked to give a practical demonstration of your program and/or an oral explanation of your analysis. If this does not allow us to authenticate your learning, you may be deferred to the University academic misconduct committee.

The purpose of this assignment is to give on-hands experience that will help you in your future studies and in the workplace. You have nothing to gain, and a lot to lose, by not approaching it honestly.

If you have any doubts, please ask questions.