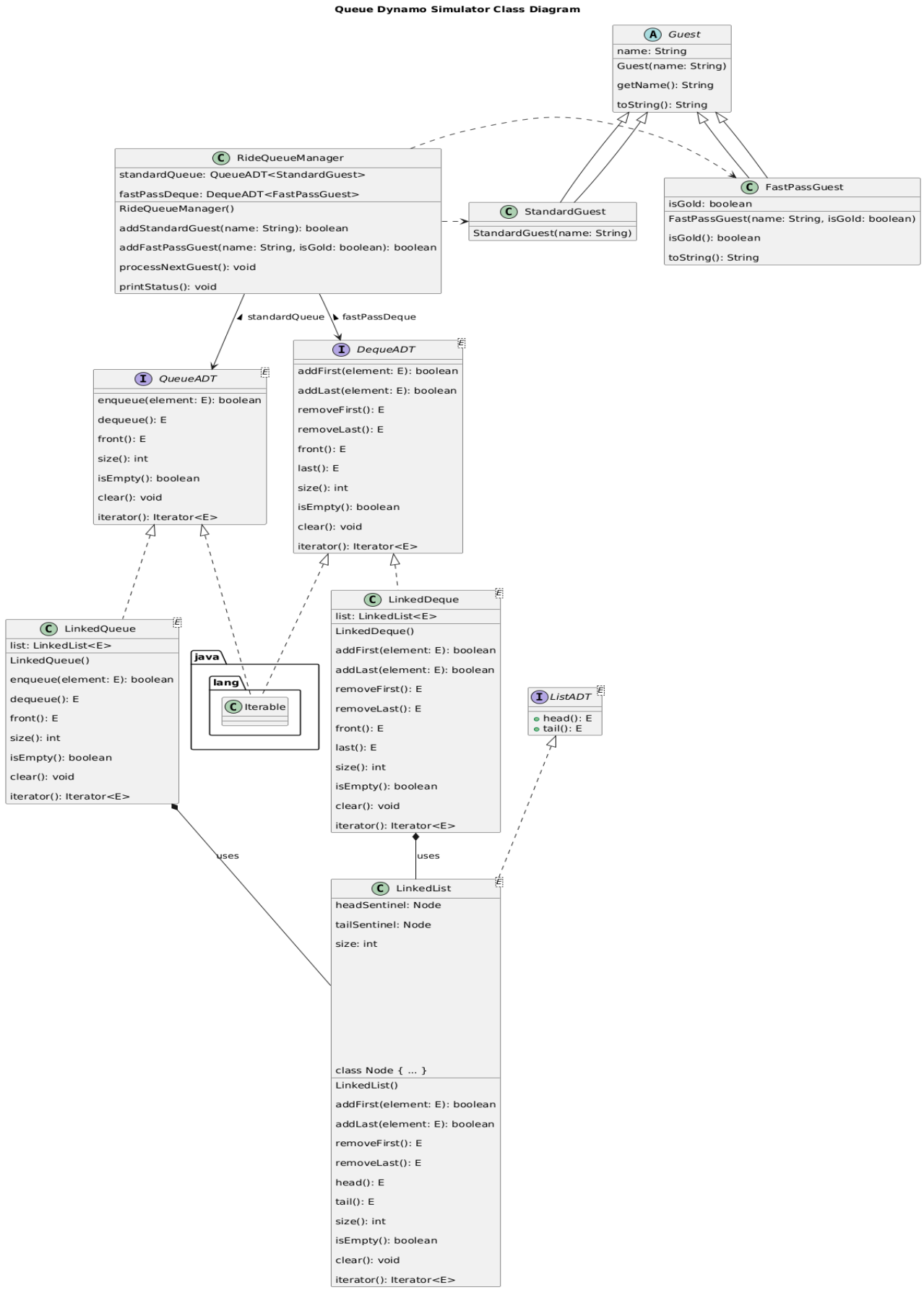

Taru Nigam C3509116

Testing

Testing was conducted for the data structures (LinkedList, LinkedQueue, LinkedDequeue) using Junit in the DataStructureTest.java file. The test covers initialization, add/remove operatiosn, edge cases, iterators, and toString methods. Below is the summary of the test cases.

Test ID	Description	Expected	Actual	Notes / Actions
1	LinkedList: Initialize empty list	isEmpty() true, size 0	Passed	Initial test to confirm setup.
2	LinkedList: addFirst and check head/tail	Head "A", tail "A" after addFirst("A")	Passed	Verified single element.
3	LinkedList: addLast and check order	[A, B] after addLast("A"), addLast("B")	Passed	Confirmed FIFO addition.
4	LinkedList: add at index 1 in [A, C]	[A, B, C]	Passed	Checked index insertion.
5	LinkedList: add at invalid index	IndexOutOfBoundsException	Passed	Edge case handled.
6	LinkedList: removeFirst from [A, B]	Returns "A", size 1	Passed	Verified removal.
7	LinkedList: removeLast from [A, B]	Returns "B", size 1	Passed	Confirmed last removal.
8	LinkedList: remove at index 1 in [A, B, C]	Returns "B", [A, C]	Passed	Index removal tested.
9	LinkedList: remove from empty list	IllegalStateException	Passed	Exception thrown correctly.
10	LinkedList: get invalid index	IndexOutOfBoundsException	Passed	Bounds checking.
11	LinkedList: head/tail on empty	IllegalStateException	Passed	Edge case.
12	LinkedList: contains existing/non-existing	True for "Apple", false for "Orange"	Passed	Search functionality.
13	LinkedList: clear list	isEmpty true, size 0	Passed	Reset operation.
14	LinkedList: Iterator traversal	Next elements: One, Two, Three	Passed	Full iteration.
15	LinkedList: Iterator on empty	NoSuchElementException on next()	Passed	Empty iterator.
16	LinkedQueue: Initialize empty	isEmpty true, size 0	Passed	Basic setup.
17	LinkedQueue: enqueue and front	Front "First" after enqueue	Passed	FIFO order.

18	LinkedQueue: dequeue	Returns "A", size 1	Passed	Removal test.
19	LinkedQueue: dequeue empty	IllegalStateException	Passed	Exception handling.
20	LinkedQueue: front without remove	Returns "X", size unchanged	Passed	Peek operation.
21	LinkedQueue: clear	isEmpty true	Passed	Reset.
22	LinkedQueue: Iterator	Traverses Alpha, Beta, Gamma	Passed	Iteration.
23	LinkedDeque: Initialize empty	isEmpty true	Passed	Setup.
24	LinkedDeque: addFirst/addLast	Front "B", last "A"	Passed	Order check.
25	LinkedDeque: removeFirst/removeLast	Correct removals	Passed	Operations.
26	LinkedDeque: remove empty	IllegalStateException	Passed	Exceptions.
27	LinkedDeque: front/last	Returns elements without removal	Passed	Access.
28	LinkedDeque: clear	isEmpty true	Passed	Reset.
29	LinkedDeque: Iterator	Traverses Y, Z, A	Passed	Full traversal.
30	Integration: Run App.java with sample inputs	Outputs match Figure 1	Passed	Manual verification of full system.
31	Edge Case: Add null name to RideQueueManager	Returns false, prints message	Passed	Input validation.
32	Failure Example: LinkedList add at index (initial bug)	Incorrect insertion order	Failed initially	Debugged by checking predecessor/successor links in addBetween; fixed node linking.
33	Repeat Test 32 after fix	Correct order	Passed	Confirmed resolution.

The overall testing by Junit achieved 100% pass rate after fixes. Manual testing of the code worked as per given guidelines.

Reflection

The initial setup and programming part was difficult but with the help of the course materials, I understood what I needed to do and had some assistance from research. I faced challenges while debugging the iterator traversal and ensuring exception handling for empty structures,

which I fixed using the print statements and then referring back to the notes on node linking. Following the SDLC, I started with linkedlist testing incrementally, then built the dependant classes around it. I faced difficulties in understanding the linkedlist and using the node linking. While compiling the code, I received many errors and crashes. I went back to the notes and online information to help me better understand where I went wrong and learnt about the mistakes, using the information I went back and solved the errors and rebuilt the code. The code ran successfully and is able to run through completely without errors. The assignment has taught me about the key concepts in data structures, polymorphism, and software design, specifically through implementation of doubly linked list with sentinel nodes and modifying it for queue/dequeue ADTs. This assignment has improved my debugging skills and understanding of unit testing, as it helped catch bugs within the program earlier.