Mihir Jham
CS251
Project 2
Report

CS251 Project2 Report

1) The SimpleStack class uses a linked list as its data structure, as its advantages outweighed the advantages of using an array in this instance. The primary reason I chose to use the Linked List data structure is because it quickly adds a new element to the list and has the stack grow as the resources expands. I implemented the class by having the basic methods: push(), pop(), size() isEmpty(). A private Node class is embedded within to make use of the Linked List. The class uses generics so that it is abstract enough to be used by different data types.
2) The SimpleQueue class uses a linked list as its data structure, as its advantages outweighed the advantages of using an array in this instance. The primary reason I chose to use the Linked List data structure is because it quickly adds a new element to the list and has the stack grow as the resources expands. I implemented the class by having the basic methods: enqueue(), dequeue(), size() isEmpty(). A private Node class is embedded within to make use of the Linked List. The class uses generics so that it is abstract enough to be used by different data types. The class also implements the Iterable() class, to make use of the foreach iterator.
3) The performance of dequeue() is constant as it always removes the head. The performance of enqueue() is different. To make it constant time, we need both a head pointer and a tail pointer in the data structure. In dequeue(), once the queue is empty, the tail points to null. This ensures constant time operations.
4) Check if input is an operator
   If it is an operator, pop the stack twice and process the operation according to input operator and push it the result onto the stack.
   If it is not an operator, check if it is null.
   If it is not null, push the input variable onto the stack(as it will be a number)
   If it is null, throw an error.