COMP 203 LAB 4 SOLUTION:

1. public class ArrQueue<T> { //(5pt)

private T[] array;

private int front;

private int rear;

private int size;

private int capacity;

public ArrQueue(int capacity) {(5pt)

this.capacity = capacity;

this.array = (T[]) new Object[capacity];

this.front = 0;

this.rear = -1;

this.size = 0;

}

public void Enqueue(T item) {//(10pt)

if (IsFull()) {

System.out.println("Queue is full. Cannot enqueue.");

return;

}

rear = (rear + 1) % capacity;

array[rear] = item;

size++;

}

public T Dequeue() {//(10pt)

if (IsEmpty()) {

System.out.println("Queue is empty. Cannot dequeue.");

return null; // Return an error value or handle it differently for generic types

}

T item = array[front];

array[front] = null; // Clear the reference to allow for garbage collection

front = (front + 1) % capacity;

size--;

return item;

}

public int Size() {//(5pt)

return size;

}

public boolean IsEmpty() { //(5pt)

return size == 0;

}

public boolean IsFull() {

return size == capacity;

}

public static void main(String[] args) { //(10pt)

ArrQueue<Integer> queue = new ArrQueue<>(5);

queue.Enqueue(1);

queue.Enqueue(2);

queue.Enqueue(3);

System.out.println("Queue Size: " + queue.Size());

while (!queue.IsEmpty()) {

System.out.println("Dequeued: " + queue.Dequeue());

}

}

}

2. public class SLLQueue<T> {//(10pt)

private static class Node<T> {

T data;

Node<T> next;

Node(T data) {

this.data = data;

}

}

private Node<T> head;

private int size;

public SLLQueue() {

head = null;

size = 0;

}

public void Enqueue(T item) { //(10pt)

Node<T> newNode = new Node<T>(item);

if (head == null) {

head = newNode;

} else {

Node<T> current = head;

while (current.next != null) {

current = current.next;

}

current.next = newNode;

}

size++;

}

public T Dequeue() {//(10pt)

if (IsEmpty()) {

System.out.println("Queue is empty. Cannot dequeue.");

return null; // Return an error value or handle it differently for generic types

}

T item = head.data;

head = head.next;

size--;

return item;

}

public int Size() {//(5pt)

return size;

}

public boolean IsEmpty() { //(5pt)

return size == 0;

}

public static void main(String[] args) {//(10pt)

SLLQueue<Integer> queue = new SLLQueue<>();

queue.Enqueue(1);

queue.Enqueue(2);

queue.Enqueue(3);

queue.Enqueue(4);

System.out.println("Queue Size: " + queue.Size());

while (!queue.IsEmpty()) {

System.out.println("Dequeued: " + queue.Dequeue());

}

}

}