CS240

https://github.com/aboman17/CS240HW.git

/\*\*

A class implementing an ADT QUEUE using a doubly linked node.

@author David Roura

@version 1.0

\*/

public class DLDeque<T> implements DequeInterface<T>

{

private DLNode head; //pointer to beginning of line

private DLNode tail; //pointer to end of line

public DLDeque()

{

head = null;

tail = null;

}

public DLDeque(first person/thing in line)

{

//Fill in Later

}

public void addToFront(T newEntry)

{

DLNode newNode = newDLNode(null, newEntry, firstNode)//create a new node

if(isEmpty())

tail = newNode;

else

head.setPreviousNode(newNode);//set it to the front of the line

//fix all pointers

}

public void addToBack(T newEntry)

{

DLNode newNode = newDLNode(prevNode, newEntry, null);//create a new node

if(isEmpty())

head = newNode;

else

lastNode.setNextNode(newNode);

lastNode = newNode;//set it to the back of the line

//fix all pointers

}

public T removeFront()

{

T front = getFront();

assert head != null;

head = head.getNextNode();

if(head == null)

tail = null;

else

head.setPreviousNode(null);

return head;

//check if data is there; if NOT throw exception

//hold data in temp variable

//delete data in DLNode (for secuirity)

//move head to next

//Only IF there is a DLNode move its prev to null

}

public T removeBack()

{

T back = getBack()

assert tail != null;

tail = tail.getPreviousNode();

if(tail == null)

head = null;

else

tail.setNextNode(null);

return tail//check if there is data there if NOT throw exception

//hold data in temp variable

//delete data in DLNode (for secuirity)

//move TAIL to prev

//ONLY IF there is a DLNode that TAIL is pointing

}

public T getFront()

{

if(isEmpty())

throw new EmptyQueueException();

else

return head.getData;

}

public T getBack()

{

if(isEmpty())

throw new EmptyQueueException();

else

return tail.getData;

}

public boolean isEmpty()

{

return (head == null) && (tail == null);

}

public void clear()

{

head = null;

tail = null;

}

}

\*\*

A class implementing an ADT QUEUE using a single linked node.

@author David Roura

@version 1.0

\*/

public class SLDeque<T> implements DequeInterface<T>

{

private SLNode head;

private SLNode tail; //pointer to beginning of line

//pointer to end of line

public SLDeque()

{

head = null;

tail = null;

}

public DLDeque(first person/thing in line)

{

//Fill in Later

}

public void addToFront(T newEntry)

{

SLNode newNode = new SLNode(null, newEntry, firstNode)

if(isEmpty())

tail = newNode;head.setPreviousNode(newNode);

else

tail = newNode;

}

public void addToBack(T newEntry)

{

SLNode newNode = newDLNode(prevNode, newEntry, null);

if(isEmpty())

head = newNode;

else

lastNode.setNextNode(newNode);

tail = newNode;

}

public T removeFront()

{

T front = getFront();

assert head != null;

head = head.getNextNode();

if(head == null)

tail = null;

else

head.setPreviousNode(null);

return front;

}

public T removeBack()

{

T back = getBack()

assert tail != null;

tail = tail.getNextNode();

if(tail == null)

head = null;

else

tail.setNextNode(null);

return back

}

public T getFront()

{

if(isEmpty())

throw new EmptyQueueException();

else

return head.getData;

}

public T getBack()

{

if(isEmpty())

throw new EmptyQueueException();

else

return tail.getData;

}

public boolean isEmpty()

{

return (head == null) && (tail == null);

}

public void clear()

{

head = null;

tail = null;

}

}