/////////// DobbeltLenketListe ////////////////////////////////////

import java.util.Comparator;

import java.util.ConcurrentModificationException;

import java.util.Iterator;

import java.util.NoSuchElementException;

import java.util.Objects;

public class DobbeltLenketListe<T> implements Liste<T>

{

private static final class Node<T> // en indre nodeklasse

{

// instansvariabler

private T verdi;

private Node<T> forrige, neste;

private Node(T verdi, Node<T> forrige, Node<T> neste) // konstruktør

{

this.verdi = verdi;

this.forrige = forrige;

this.neste = neste;

}

protected Node(T verdi) // konstruktør

{

this(verdi, null, null);

}

} // Node

// instansvariabler

private Node<T> hode; // peker til den første i listen

private Node<T> hale; // peker til den siste i listen

private int antall; // antall noder i listen

private int endringer; // antall endringer i listen

// hjelpemetode

private Node<T> finnNode(int indeks)

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

// konstruktør

public DobbeltLenketListe()

{

hode = hale = null;

antall = 0;

endringer = 0;

}

// konstruktør

public DobbeltLenketListe(T[] a)

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

// subliste

public Liste<T> subliste(int fra, int til)

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

@Override

public int antall()

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

@Override

public boolean tom()

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

@Override

public boolean leggInn(T verdi)

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

@Override

public void leggInn(int indeks, T verdi)

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

@Override

public boolean inneholder(T verdi)

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

@Override

public T hent(int indeks)

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

@Override

public int indeksTil(T verdi)

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

@Override

public T oppdater(int indeks, T nyverdi)

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

@Override

public boolean fjern(T verdi)

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

@Override

public T fjern(int indeks)

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

@Override

public void nullstill()

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

@Override

public String toString()

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

public String omvendtString()

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

public static <T> void sorter(Liste<T> liste, Comparator<? super T> c)

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

@Override

public Iterator<T> iterator()

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

public Iterator<T> iterator(int indeks)

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

private class DobbeltLenketListeIterator implements Iterator<T>

{

private Node<T> denne;

private boolean fjernOK;

private int iteratorendringer;

private DobbeltLenketListeIterator()

{

denne = hode; // denne starter på den første i listen

fjernOK = false; // blir sann når next() kalles

iteratorendringer = endringer; // teller endringer

}

private DobbeltLenketListeIterator(int indeks)

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

@Override

public boolean hasNext()

{

return denne != null; // denne koden skal ikke endres!

}

@Override

public T next()

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

@Override

public void remove()

{

throw new UnsupportedOperationException("Ikke laget ennå!");

}

} // DobbeltLenketListeIterator

} // DobbeltLenketListe