Aufgabe 8:

Für die Implementierung einer Feldstruktur (Array List) der Form sei die Klasse ListeF<T> aus der Vorlesung gegeben.

Ergänzen Sie die Klasse um eine Methode
public void loescheErst()
die das erste Element der Liste löscht, sodass alle Elemente am Ende wieder aneinandergereiht sind.

Es sollen hier die nachfolgenden Elemente eins nach „vorne“ rutschen.

Aufgabe 9 (wer früher gehen will – 1 von 2):

Für die Implementierung einer Feldstruktur (Array List) der Form sei die Klasse ListeF<T> aus der Vorlesung gegeben.

Ergänzen Sie die Klasse um eine Methode
public void loescheMin()
die das niedrigste Element löscht, sodass alle Elemente am Ende wieder aneinandergereiht sind.

Damit Vergleiche in generischen Datenstrukturen möglich sind erweitern sie vorab bitte den Kopf der Klassendefinition auf
public class ListeF <T extends Comparable<T>> {
und ändern im Konstruktor den Array-Typ Object auf Comparable.

Der Vergleich zwischen zwei Werten ist dann wie folgt möglich:
a.compareTo(b) > 0, wenn gilt a > b und
a.compareTo(b) < 0, wenn gilt a < b

Zusätzlich:
a.compareTo(b) = 0, wenn gilt a = b oder
a.equals(b) = true, wenn gilt a = b (dies ist immer möglich)

Aufgabe 10:

Ergänzen Sie die Klasse **ListeL** um eine Methode
public int find1 (I o)
die mit einer passenden Schleife eine Suche auf der „Liste“ durchführt und die Position/den Index (≥ 0) zurückliefert, an der o zum ersten Mal in der Liste vorkommt, oder −1, falls o nicht in der Liste vorkommt.

Aufgabe 11:

Überlegen Sie sich eine oder mehrere alternative Instanzmethoden
ElementL<T> insert (T o, ElementL<T> pos)
für die in der Vorlesung behandelte Klasse **ListeL** zur Darstellung einer einfach verketten Liste, die das Objekt **o** an der von **pos** referenzierten Stelle einfügen können. **Pos** wird so zum Nachfolger des neuen Elementes aus **o**. Es liegt also keine Referenz auf den Vorgänger in der Liste vor. Mit Aufwand O(1) bis O(n) – also max. 1 Schleife – kann man sich diese jedoch besorgen.

Aufgabe 12 (freiwillig, da erhöhte Schwierigkeit):

Ergänzen Sie die Klasse **ListeL** um eine Methode
public void sort1()
die die Liste mittels Selectionsort sortiert. Dabei sollen die zu tauschenden Elemente nicht aus der verketteten Liste ausgehängt werden, sondern lediglich ihre Inhalte vertauscht werden.

Aufgabe 13 (wer früher gehen will – 2 von 2):

Ergänzen Sie die Klasse **ListeL** um eine Methode, die das erste Elemente der Liste aushängt und es vor einem Element einhängt, das beginnend am neuen Kopf, das erste Element ist, das inhaltlich größer als das umzuhängende Element selbst ist.
Beispiel: Die Liste 5 – 3 – 7 – 2 – 8 – 6 wird zu 3 – 5 – 7 – 2 – 8 – 6.