

## NitscheSahoo - AD Praktikum für 17.10.13 – Aufgabe 1 - Listen

### Das Interface:

```
public interface IList<T> {
    // add elem to the front
    void cons(T elem);

    // removes first element and return first
    T head();

    // Not in interface but a simple get(index) method
    // throws outofbound when n > arrlength-1 && n < 0
    T get(int index);

    // get first element
    T first();

    // how many elements has a list?
    int length();

    // is the list empty?
    boolean isempty();

    // insert element after index n so between n and n+2 if there is a current
    // n+2
    void insert(T elem, int n);
}
```

### Akkumulation:

„Müssen Sie in Ihrer Lösung alle bisherigen Messwerte mitspeichern? Wenn Ja: Geht das auch anders? Wie?“

Unter Verwendung des Satz von Steiner brauchen wir akkumulierend nur zwei Werte zu speichern und zur Rückgabe der Varianz/des Durchschnitts nur kleine Berechnungen zu machen. Zum Vergleich der expliziten und iterativen Implementation haben wir jedoch dennoch die Speicherung aller Werte und die explizite Varianten implementiert. Wir setzten diese jedoch nicht ein

### Aufgabe 5 – Anzahl der Dereferenzierungen

Siehe Datei „ab1\_test/ListAufgaben.java“ Zeile 19-33.

The first elements need 1 access. All following elements need 2 accesses.

Therefore with NoOfElements=15 we have:  $1 + (15 - 1) * 2 = 29$  accesses.

### Aufgabe 6 - Am Anfang hinzufügen

Siehe Datei „ab1\_test/ListAufgaben.java“ Zeile 36-81.

Siehe Excel tabelle.

These observations fit to a  $O(n)$ -Algorithm. Indeed, the time needed for inserting elements at the front grows proportionally to the number of elements.

10x the elements needs approx. 10x the time.

The initial overhead can also be recognized.

### Aufgabe 7 - Am Ende inserten:

Siehe Datei „ab1\_test/ListAufgaben.java“ Zeile 85-134.

Siehe Excel tabelle.

The number of Accesses suspects a quadratic complexity.

It can be shown, that inserting at the end in a linked list is indeed of quadratic complexity.

## Aufgabe 8 - An einer zufälligen Stelle hinzufügen:

Siehe Datei „ab1\_test/ListAufgaben.java“ Zeile 137-152.

Siehe Excel tabelle. Wir stellen fest, dass der Durchschnitt langsam sinkt. Dies liegt wahrscheinlich daran, dass der Overhead nur bei kleinen n bemerkbar ist. Die hohe Varianz lässt sich dadurch erklären, dass aufgrund des Hinzufügens an einer zufälligen Position die benötigte Zeit, je nach dem, wie oft weiter hinten in der Liste hinzugefügt wird, stark schwankt.

## Aufwandsabschätzung:

Arbeitszeit: 9h gemeinsam – viel davon ist organisatorisches

Zeitverteilung: 10% AvgVarianz - 60% Listen und Experimente - 30% Tests und Javadoc

## Vorschau der Javadoc:

The screenshot displays the Javadoc API documentation for the `MLinkedList` class. The interface is divided into several sections:

- All Classes:** A sidebar on the left lists the classes `AverageVariance`, `IList`, and `MLinkedList`.
- Constructor Detail:** A sidebar on the left shows the constructor `MLinkedList` with its signature `public MLinkedList()` and the note "Standard-Constructor".
- Method Detail:** A sidebar on the left shows the method `MLinkedList` with its signature `public MLinkedList(T first)` and a description: "Takes the first element of the list and returns a linked list with this element already included." It also lists the parameter `first` as "the first element to be included".
- Class MLinkedList<T>:** The main content area shows the class declaration `ab1_adts.ListImpl`, its inheritance from `java.lang.Object`, and its implementation of the `IList<T>` interface. It includes the source code snippet:

```
public class MLinkedList<T>
    extends java.lang.Object
    implements IList<T>
{
    // ...
}
```
- Constructor Summary:** A table summarizing the constructors:

Constructor and Description
<code>MLinkedList()</code> Standard-Constructor
<code>MLinkedList(T first)</code> Takes the first element of the list and returns a linked list with this element already included.
- Method Summary:** A table summarizing the methods:

Modifier and Type	Method and Description
-------------------	------------------------