Aufgabenblatt 1- ADP3

Cao,Thi Huyen; Rothenburg, Daniel

April 11, 2016

1 Documentation

1.1 UML Diagramm

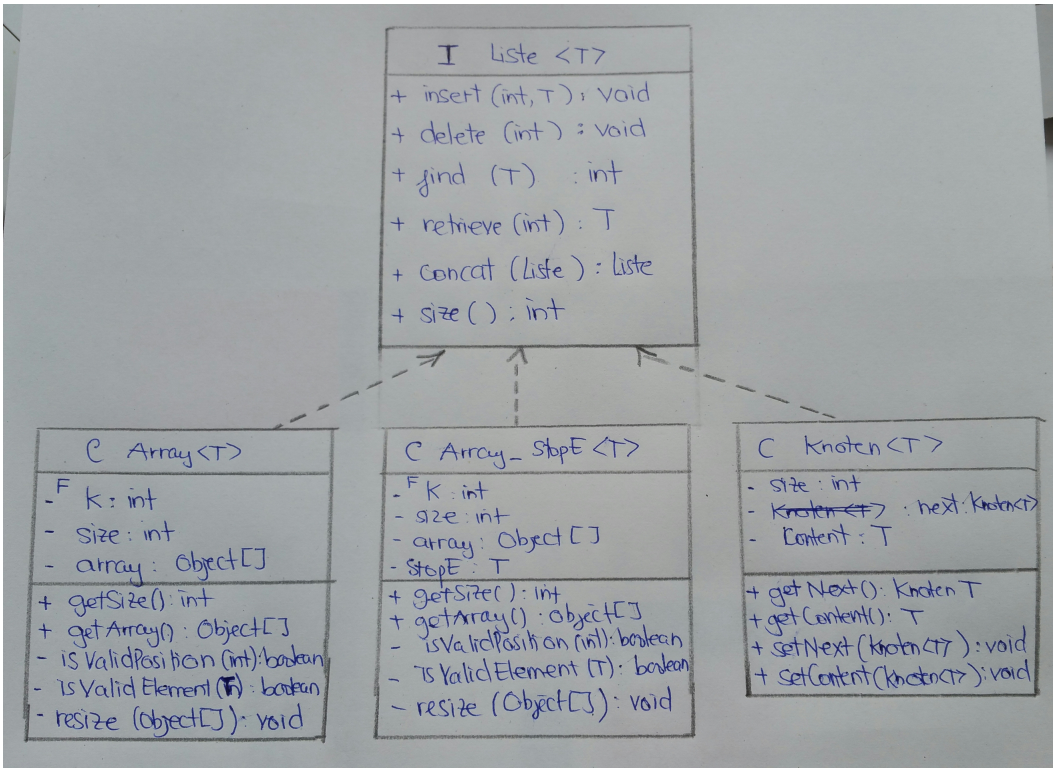

Figure 1: 3 verschiedene Implementation für die Liste

1.2 Pre und Postcondition

Operation Insert **Pre**: Die Position und Das Element ist gütig. Also kein Null Element und Position ist entweder am Anfang, zwischen 2 Elemente oder am Ende. **Post** :hinzugefügtes Element in der Liste

Operation Delete **Pre** : gültige Position (sehen Insert). **Post** :neue Liste mit entferntem Element

Operation Find **Pre**: Element von Typ T. **Post** : Position des gesuchter Element oder -1 (Wenn das Element nicht in der Liste ist)

Operation Retrieve **Pre**: gültige Position (sehen Insert). **Post** :null oder das gefundene Element

Operation Concat **Pre**: eine Liste von einem Datentyp, der kompatibel zu List ist. **Post** : alte Liste falls eingegebene Liste leer ist/ neue Liste: alle Element von eingegebener Liste werden am Ende der alten Liste angehängt

Operation Insert **Pre**:– **Post** :–

2 Aufwandsanalyse

2.1 Array

→ Zugriff auf spezielles Element über die Position in der Liste sehr schnell, da jedes Element in dem Array ein Index hat.
→ Insert und Delete ein Element kostet viel Arbeit, da manchmal das Array verlängert werden, viele Elemente nach vorne, hinter verschoben oder kopiert werden müssen.
→ Suchen ein Element am Anfang des Array ist schneller als am Ende. Da man der Key mit allen Elementen in dem Array vergleichen muss.

2.2 sortierter Array mit Stop Element

→ Das Array ist sortiert nach next und previous index.
→ Stop Element, vermeiden auf Exception

2.3 Linkedlist

→ Zugriff auf Elemente am Anfang ist viel schneller als Element am Ende, da es eine Verkettung zwischen Elementen gibt. → Beim insert und delete Element in der Mitte, concat 2 Liste muss einfach nur die Verkettung an der Stelle verändert werden. Kopier oder Verschiebung muss nicht gemacht werden.

END