Team: 04 – Human Soleimannejad, Ulf Lasner 24.03.2016

Aufgabenaufteilung: Wir haben die Skizze zusammen erstellt

Quellenangaben: Aufgabenstellung.

Bearbeitungszeitraum: 22.03.2016 ca. 4 Std. | 24.03.2016 ca. 4 Stunden | 24.03.2016 ca. 2 Stunden

Aktueller Stand: Fertige Skizze.

Änderungen in der Skizze: Am 24.03.2016 nach der Vorlesung die Details für die interne Umsetzung entfernt und gekürzt, da zu viel Architektur drinstand.
Am 24.03.2016 nach der Email von Herr Klauck.
Schnittstellen für die Austauschbarkeiten wurden sichergestellt. Package-Hierarchie und Sichtbarkeiten wurden hinzugefügt.
Parameter wurde der Spezifikation hinzugefügt.

Skizze:
Bei uns befindet sich alles in einem Package namens ADT.
ADTListe, ADTStack, ADTQueue, ADTArray1, ADTArray2 und ADTBTree sind alle als einzelne Java Klassen in diesem Package enthalten. Jeder dieser Klassen ist durch gleichnamige Klassen anderer Teams austauschbar, was die Schnittstelle unseres ADT's darstellt.
Die einzelnen Methoden sind auch austauschbar, da der Methodenname, die Parameterliste und die Rückgabewerte vorgegeben sind.
Alle unsere Methoden sind public, da sie in jeder Klasse unterschiedliche Namen haben und es so nicht zu irgendwelchen Überschneidungen kommen kann.
Alle unsere Methoden, die mit create zu tun haben werden von uns als Klassenmethode bzw. static Methode erstellt, da wir so kein Objekt benötigen um ein Objekt bspw. eine Liste, zu erstellen.
Leere Klammern bei den Parameterlisten der Methoden bedeutet, dass sie keine Parameter erwarten. Sind Parameter angegeben so ist ihr Datentyp angegeben bzw. die Form, die reinkommt und nicht der Variablenname.

ADT Liste:

Die Liste beginnt nach außen hin bei 1 beginnen. Sie muss nicht destruktiv bearbeitet werden können. Also wenn wir ein Element hinzufügen darf kein Element verloren gehen. Alle Elemente der Liste sind ganze Zahlen. Alle hier beschriebenen Operationen müssen in der Implementierung enthalten sein, es dürfen keine weiteren Operationen hinzugefügt werden bzw. welche entfernt werden.

Operationen/Methoden für die ADT Liste:

- Create(): Erstellt eine Liste für ganze Zahlen.
- isEmpty(ADTListe): Prüft, ob die Liste leer ist oder nicht. Gibt einen Wahrheitswert zurück. True=leer. False = nicht leer.
- Equal(ADTListe, ADTListe): Vergleicht zwei Listen ob sie die gleichen Zahlen beinhalten und sich diese in derselben Reihenfolge befinden. Und gibt true für gleich aus und false für nicht gleich.
- laenge(ADTListe): Gibt die Länge der Liste als Zahl zurück.

- Insert(ADTListe,Position,GanzeZahl): Setzt einen übergebenen Wert an übergebenen Position in übergebene Liste ein.
- Delete(ADTListe,Position): Löscht an einer übergebenen Position der übergebenen Liste das Element welches an dieser Position ist.
- Find(ADTListe,GanzeZahl): Sucht aus einer übergebenen Liste ein übergebenes Element, zurückgegeben wird die Position des Elements in der Liste.
- Retrieve(ADTListe,Position): Sucht aus einer übergebenen Liste an einer übergebenen Position. Und übergibt das an dieser Stelle gefundene Zahl zurück.
- Concat(ADTListe, ADTListe): Verbindet 2 übergebene Listen zu einer und gibt die neue Liste zurück.

ADT Stack:

Operationen/Methoden des ADT Stacks:

- createS(): Erstellt einen neuen Stapel(Stack) für ganze Zahlen.
- Push(ADTStack, GanzeZahl): Es wird eine übergebene Zahl auf den Stapel gelegt.
- Pop(ADTStack): Entfernt die oberste Zahl vom Stapel und gibt diesen aus.
- Top(ADTStack): Es wird die erste Zahl des Stapels zurückgegeben.
- isEmptyS(ADTStack): Prüft ob der übergebene Stack leer ist und gibt Wahrheitswert zurück.
- equalS(ADTStack, ADTStack): Prüft ob 2 übergebene Stapel die gleichen Zahlen beinhalten und sich diese im Stapel an derselben Reihenfolge befinden.
- reverseS(ADTStack): Der übergebene Stapel wird umgedreht zurückgegeben. Intern wird kein Element verschoben, damit die Performanz gegeben ist.

ADT Queue:

- createQ(): Erstellt eine neue Schlange für ganze Zahlen. (First in – First Out.)
- enqueue(ADTQueue,GanzeZahl): Fügt eine Zahl an den Schwanz der Schlange und gibt die neue Schlange zurück.
- Dequeue(ADTQueue): Entfernt die Zahl an der Spitze und gibt die neue Schlange aus.
- front(ADTQueue): Gibt die vorderste Zahl der Schlange aus.
- isEmptyQ(ADTQueue): Prüft, ob die übergebene Schlange leer ist und gibt einen Wahrheitswert zurück.
- equalQ(ADTQueue, ADTQueue): Prüft ob 2 übergebene Schlangen die gleichen Zahlen in der gleichen Reihenfolge besitzen.

ADT Array1:

Das Array beginnt bei der Stelle 0. Es arbeitet destruktiv, also werden Zahlen gelöscht, sobald eine Zahl auf einer Position eingefügt wird, wo bereits eine Zahl stand. Das Array hat eine beliebige Größe.

Operationen/Methoden für das ADT Array1:

- initA(GanzeZahl): Gibt ein leeres Array1 zurück, die so lang ist, wie die Länge der ganzen Zahl.
- SetA(ADTArray1, Position,GanzeZahl): in einem übergebenen Array1 wird an übergebener Position eine übergebene Zahl eingefügt.

- getA(ADTArray1, Position): Zurückgegeben wird eine zahl welche an der übergebenen Position in dem übergebenen Array1 steht.
- lengthA(ADTArray1): Gibt die Länge des Arrays aus.
- EqualA(ADTArray1, ADTArray1): prüft ob 2 übergebene Arrays die gleichen Zahlen beinhalten und die gleiche Reihenfolge besitzen.

ADT Array2:

Ist das gleiche wie oben nur die innere Realisierung unterscheidet sich. In diesem Fall nur durch direkte Anwendung des Java Int Arrays anstelle des Zugriffs auf die ADT Liste.
Nach außen hin ist es das gleiche. Analog zu ADT Array1 sind auch die Methoden und Parameterlisten zu implementieren, nur das statt ADT Array1, ADT Array2 verwendet wird.

ADT BTree:

Jeder Baum besitzt einen ganzzahligen Wert eine Höhe als ganze Zahl und einen linken Baum und rechten Baum.

- initBT(): Erstellt aus einem leeren Baum.
- setLeftSuc(ADTBTree,ADTBTree): Setzt einen übergebenen Baum in einen anderen übergebenen Baum in der linken Seite ein.
- setRightSuc(ADTBTree,ADTBTree): Setzt einen übergebenen Baum in einen anderen übergebenen Baum in der rechten Seite ein.
- setHigh(ADTBTree,GanzeZahl): Setzt die Höhe des Baums und gibt den Baum zurück.
- setVal(ADTBTree,GanzeZahl): Setzt den Wert des Baums und gibt den Baut zurück.
- getLeftSuc(ADTBTree): Gibt den linken Teilbaum des übergebenen Baums zurück.
- getRightSuc(ADTBTree): Gibt den rechten Teilbaum des übergebenen Baums zurück.
- getHigh(ADTBTree): Gibt die Höhe des übergebenen Baums zurück.
- getVal(ADTBTree): Gibt den Wert des übergebenen Baums zurück.
- isEmptyBT(ADTBTree): Prüft ob der übergeben Baum leer ist und gibt einen Wahrheitswert zurück.
- Print(ADTBTree,File): erstellt eine .dot Datei, die zu Veranschaulichung in eine Bilddatei umgewandelt werden kann.