Algorithmen und Datenstrukturen

Informationen zum Praktikum

Sommersemester 2019

Prof. Dr. Thomas Fuhr

Praktikumsaufgabe – Überblick

Entwerfen Sie eine Klassenbibliothek/Framework für Dictionaries (Wörterbücher), welche(s) Integer-Elemente verwalten kann.

a) Klären Sie welche ***abstrakten Datentypen*** zu berücksichtigen sind.
   - wird gemeinsam in der Übung festgelegt.

b) Als ***konkrete Datentypen*** sind zu implementieren:
   - siehe weitere Folien

c) Organisieren Sie die Datentypen in geeigneter Weise in einer ***Klassenhierarchie***
   - Trennen Sie in geeigneter Weise in Schnittstellenklassen (keine Methoden implementiert), abstrakte Klassen (teilimplementiert)) und konkrete Klassen.

d) Implementieren Sie die Hierarchie in einer ***objektorientierten Sprache Ihrer Wahl***.

Berücksichtigen Sie hierbei die wesentlichen Konzepte der ***Objektorientierung***:
   - Wiederverwendung von Code
   - Schnittstellenklassen / abstrakte Klassen / konkrete Klassen

WICHTIG: Implementieren Sie die Verwaltungsfunktionen ***effizient***!

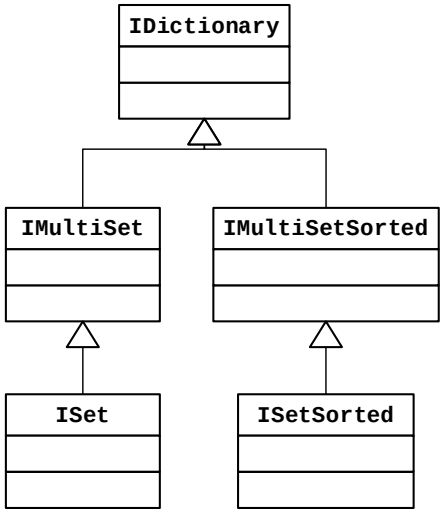
Praktikumsaufgabe – Schnittstellenhierarchie

- Ein Dictionary (Wörterbuch) ist i. Allg. Fall eine unsortierte Sammlung von Objekten, dabei können Mehrfacheinträge gestattet oder verboten sein.
  - Multiset (Familie)
  - Set (Menge)

- Für bestimmte Anwendungen sind aber auch sortierte Dictionaries sinnvoll, d.h. solche, in denen die verwalteten Objekte nach irgendeinem Kriterium sortiert sind.
  - Sorted Multiset
  - Sorted Set

- Schnittstellenhierarchie ohne Mehrfachvererbung!

  !!Benamung übernehmen!!



Praktikumsaufgabe – konkrete Datentypen (1)

Als konkrete Dictionary-Datentypen sind zu implementieren
(!! ***und auch so zu benamen*** !!):

- Array: sortiert/unsortiert *sowie* Mehrfachvorkommen unerlaubt/erlaubt:
  - **`SetSortedArray`**, **`SetUnsortedArray`**,
  - **`MultiSetSortedArray`**, **`MultiSetUnsortedArray`**
  - ➢ 4 konkrete Klassen
  - ➢ Hinweis: Sie dürfen – zur Vereinfachung -- eine feste Maximalgröße des Arrays festlegen.
    D.h. sie müssen keine Reallokationsstrategie implementieren!
- Verkettete Liste: sortiert/unsortiert *sowie* Mehrfachvorkommen unerlaubt/erlaubt:
  - **`SetSortedLinkedList`**, **`SetUnsortedLinkedList`**,
  - **`MultiSetSortedLinkedList`**, **`MultiSetUnsortedLinkedList`**
  - 4 konkrete Klassen
- Binärer Suchbaum:
  - Welcher Art ist dieses Wörterbuch? Einbindung in Hierarchie!
  - **`BinSearchTree`**
  - ➢ 1 konkrete Klasse
- AVL-Baum:
  - Welcher Art ist dieses Wörterbuch? Einbindung in Hierarchie!
  - **`AVLTree`**
  - ➢ 1 konkrete Klasse

Praktikumsaufgabe – konkrete Datentypen (2)

Als konkrete Wörterbuch-Datentypen sind ferner berücksichtigen:

- Treap:
  - Welcher Art ist dieses Wörterbuch? Einbindung in Hierarchie!
  - **`Treap`**
  - ➢ 1 konkrete Klasse

- Hashverfahren
  - Hashfunktion: Division
  - zwei Verfahren: separate Verkettung, quadratische Sondierung
  - Welcher Art ist dieses Wörterbuch? Einbindung in Hierarchie!
  - **`HashTabSepChain`**
  - **`HashTabQuadProb`**
  - ➢ 2 konkrete Klassen
  - ➢ Hinweis: Sie dürfen – zur Vereinfachung -- eine feste Maximalgröße der Tabelle festlegen.
  - ➢ Hinweis: Rehashing muss nicht implementiert werden.

➢ plus ggf. Hilfsklassen (abhängig von Ihrer Realisierung)

Praktikumsaufgabe – Alle Datentypen

Spezifizieren bzw. implementieren Sie für alle zu berücksichtigenden Wörterbücher die öffentlichen Methoden:

- `bool search(int elem)` `// true = gefunden`
- `bool insert(int elem)` `// true = hinzugefügt`
- `bool delete(int elem)` `// true = gelöscht`
- `void print()`

```
// Einfache Ausgabe der Elemente des
// Wörterbuchs auf die Console,
// so dass Inhalt & Struktur
// daraus eindeutig erkennbar
// Verwenden Sie für die Ausgabe von
// Bäumen, das in der Übung behandelte
// Verfahren
```

Praktikumsaufgabe – Alle Datentypen

Hinweise zur Umsetzung:

- Implementieren Sie diese Methoden unter Verwendung der in der Vorlesung behandelten Algorithmen!
- Stellen Sie sich immer die Frage bzgl. Wiederverwendung (≠ Kopieren des Codes!):
  - Wird in meiner Bibliothek an anderer Stelle eine gleichartige Struktur verwendet? Wie kann ich diese ggf. wiederverwenden?
  - Wird in meiner Bibliothek an anderer Stelle ein gleichartiger Algorithmus verwendet? Wie kann ich diesen ggf. wiederverwenden?
- Denken Sie daran, dass Suchen sehr oft Teil des Einfügens und des Löschens ist! D.h. für effiziente Implementierungen des Einfügens und Löschens darf nicht mehrfach durchsucht werden!

  Implementieren Sie in diesen Fällen z. B. eine zusätzliche nicht-öffentlichen Such-Methode _**`search`**_, auf die die öffentliche **`search`**-Methode zugreift.

  Diese Suchmethode, muss als Ergebnis den „Fundort“ des gesuchten Elements hinterlassen oder aber, falls das Element nicht vorhanden, den „Zielort“, an dem das Element in die Datenstruktur integriert werden müsste.

  Die Signatur für _**`search`**_ hängt von Ihrer Realisierung ab. Ggf. können Sie auch zusätzliche Klassenattribute einführen und nutzen.

Praktikumsaufgabe – Rahmenprogramm

Realisieren Sie ein Rahmenprogramm **`main()`**, welches dem Benutzer erlaubt

1. einen *abstrakten* Datentyp auszuwählen.
2. abhängig vom gewählten Datentyp eine vorhandene Implementierung auszuwählen (*konkreter Datentyp*)
3. *flexibel* nicht negative ganze Zahlen hinzuzufügen, zu suchen und zu löschen.
4. Nutzen Sie *unbedingt* die Möglichkeit des späten Bindens!

Wichtige Hinweise:

- Es reicht eine einfache Konsolenein/ausgabe aus!
- Andere Daten als nicht negative ganze Zahlen müssen nicht durch die zu realisierenden Dictionary-Klassen verwaltet werden können!

Wesentliche Konzepte der Objektorientierung

- Trennung von Spezifikation und Implementierung
  - Schnittstellenklassen
  - ggf. teilimplementierte Klassen (abstrakte Klassen)
  - implementierte Klassen
- Information Hiding
  - eine Klasse sollte nur jene Methoden veröffentlichen, welche für die beabsichtigte Nutzung der Klasse notwendig sind.
- Wiederverwendung / Vermeidung von Doppelimplementierungen
  - wo immer sinnvoll möglich (d.h. insbesondere bei Erhalt der Klarheit der Gesamtstruktur der Klassengerüsts) sollten bereits vorhandene Funktionalitäten oder Klassen genutzt werden.
- Vererbung
  - eine abgeleitete Klasse erbt die Attribute und Methoden ihrer Oberklasse

    Bem.: eine abgeleitete Klasse sollte i. A. zusätzliche Funktionalität besitzen
- Polymorphismus
  - Objekte eines Typs können durch Objekte abgeleiteter Typen ersetzt werden
  - Analog für Referenzen

Umsetzung Praktikumsaufgabe

1. **Entwurf der Klassenhierarchie**
2. **Erste Umsetzungsschritte**

- **Gruppengespräch**

  [Konkrete Termine siehe Moodlekurs]

  Besprechung Ihres aktuellen Stands:

  - Hierzu gehört Ihre Klassenhierarchie in grafischer Form (*obligatorisch*)
  - Ggf. bereits von Ihnen implementierte Klassen (*nicht obligatorisch*)

3. **Weitere Umsetzung**

- **Zwischengespräch; nur bei Bedarf**
  - Bei Bedarf können Sie nochmals ihre Zwischenergebnisse mit mir besprechen
    - Ggf. modifizierte Klassenhierarchie
    - Ggf. Implementierungsstand

4. **Weitere Umsetzung**

4. Abnahme

Ca. eine/zwei Wochen VOR Beginn der schriftlichen studienbegleitenden Leistungsnachweise

[Konkrete Termine siehe Moodlekurs]

- Gegenstand der Abnahme:
  - Endgültige Klassenhierarchie in grafischer Form
  - Vorführung Ihres Frameworks
    - Zu jeder konkreten Klasse wird es mindestens einen Testfall geben
    - Testfälle werden von Prof. Fuhr während der Abnahme zufällig ausgewählt.

WICHTIGE HINWEISE:

*Grundlage für die Abnahme ist die erfolgreiche Umsetzung der Praktikumsaufgabe **bis zum gesetzten Termin**.*

*Insbesondere muss **jedes (!)** Gruppenmitglied in der Lage sein*

- *die Struktur der Klassenbibliothek zu begründen.*

sowie

- *seine eigenen Realisierungsanteile darzustellen und*
- *diese korrekt lauffähig zu präsentieren.*

Termine Praktikumsaufgabe

- *Alle Termine werden im Moodlekurs zur Lehrveranstaltung ausgewiesen!*

- *Sofern möglich stehe ich gerne auch zur Beratung in den Pausen vor/nach der Lehrveranstaltung bzw. in meiner Sprechstunde (nach Voranmeldung per E-Mail) zur Verfügung.*