FACHHOCHSCHULE
MANNHEIM
Hochschule für Technik und Gestaltung

Fachbereich Informatik Prof. Dr. Peter Knauber

Bachelor-Studiengang
Übungen zur Vorlesung Grundlagen der Informatik, WS 2002/2003

Übungsblatt 11: Abstrakte Datentypen in Java

Ausgabe am: 16.1.2004
Abgabe am: Freitag, 23.1.2004

Betrachtet wird ein abstrakter Datentyp *Date*:

Der ADT *Date* repräsentiert ein bestimmtes Datum im Gregorianischen Kalender und bietet verschiedene Operationen auf Kalenderdaten an.

Die jeweiligen Operationen sind wie folgt definiert:

- equals: Date × Date → boolean
  Liefert genau dann *true*, wenn die beiden Kalenderdaten gleich sind
- laterThan: Date × Date → boolean
  Liefert genau dann *true*, wenn das 1. Datum später als das 2. Datum ist
- earlierThan: Date × Date → boolean
  Liefert genau dann *true*, wenn das 1. Datum früher als das 2. Datum ist
- getWeekday: Date → int
  Liefert den Wochentag zu einem Datum;
  der Montag entspricht der 1, der Dienstag der 2 usw.
- subtract: Date × Date → int
  Subtrahiert das 2. Datum vom 1. Datum; liefert die Differenz der beiden Datumsangaben in Tagen als ganze Zahl
  Voraussetzung: das 1. Datum ist größer als das 2. Datum, sonst ist das Ergebnis -1
- isValid: int × int × int → boolean
  Liefert genau dann *true*, wenn die angegebenen Zahlen in der Reihenfolge Tag, Monat, Jahr ein gültiges Datum repräsentieren
- getDay: Date → int
  getMonth: Date → int
  getYear: Date → int
  Liefern den Tag, den Monat bzw. das Jahr des angegebenen Datums als *int*-Wert
- Date: int × int × int → Date
  Wenn es sich um ein gültiges Datum handelt, liefert der Konstruktor ein entsprechendes *Date*-Objekt als Ergebnis; ansonsten wird ein *Date*-Objekt geliefert, für das getDay(), getMonth() und getYear() jeweils -1 als Ergebnis liefern

Die verschiedenen Methoden (nicht der Konstruktor und die get-Methoden!) dürfen sich darauf verlassen, dass sie nur mit gültigen Datumsangaben aufgerufen werden.

Aufgabe 1 **10 Punkte**

Informieren Sie sich über den Gregorianischen Kalender. Beschreiben Sie kurz, wie dieser definiert ist.

Aufgabe 2 **105 Punkte**

Implementieren Sie eine Klasse *Date* in Java, welche alle oben angegebenen Methoden außer *isValid* als **Instanzmethoden** enthält. Sie dürfen beliebige Attribute und zusätzliche eigene Methoden, aber keinerlei Hilfsmethoden aus der Java-Bibliothek verwenden.
Die Methode *isValid* ist als **statische Methode** zu implementieren. Punkteverteilung:

a) equals: 7 Punkte

b) laterThan: 7 Punkte

c) earlierThan: 7 Punkte

d) getWeekday: 16 Punkte

e) subtract: 12 Punkte

f) isValid: 44 Punkte

g) getDay, getMonth, getYear: 6 Punkte

h) Date (Konstruktor): 6 Punkte

Es wird erwartet, dass Namen, Parametertypen und Ergebnistyp Ihrer Implementierung der Methoden **genau** stimmen und dass die Klasse im Package *uebung11* liegt.

Aufgabe 3 **Test und Testdokumentation: 15+15 Punkte**

Entwerfen und implementieren Sie eine Klasse *TestMyDate* mit einer *main*-Methode, die den ADT *Date* benutzt und in der er gründlich (!) getestet wird.
Welche Fälle sind für welche Operationen zu berücksichtigen? Dokumentieren Sie in Form von Kommentaren in der Testklasse Ihre Überlegungen, wie Sie sicherstellen wollen, dass Ihre Implementierung korrekt ist und welche Sonderfälle Sie wie getestet haben.

Hinweis

Überlegen Sie sich vor der Bearbeitung dieser Übungsaufgaben, wie viele Punkte Sie noch benötigen, um die 75%-Klausurzulassungsquote zu erreichen und wie viele/welche (Teil-)Aufgaben Sie bearbeiten sollten, um dieses Ziel **sicher** zu erreichen!

Es ist vermutlich nicht notwendig, alle (Teil-)Aufgaben zu bearbeiten; für das Bestehen der Klausur ist nicht relevant, ob Sie 75% oder 98% der Übungspunkte erreicht haben.