Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Einführung in die Informatik
Sommersemester 2012

Prof. Dr. Wolfram Burgard
PD Dr. Cyrill Stachniss
Dipl. Ing. Jörg Röwekämper
Dipl. Inf. Markus Kuderer

Übungsblatt 7

Abgabe bis Freitag, 22.06.2012, 12:00 Uhr

Hinweis:
Aufgaben immer per E-Mail (eine E-Mail pro Blatt und Gruppe) an den zuständigen Tutor schicken (Bei Programmieraufgaben Java Quellcode und eventuell benötigte Datendateien).

Aufgabe 7.1

Schreiben Sie eine Klasse `Calendar`, die eine Methode enthält, die den Wochentag für ein beliebiges Datum berechnet. Z.B. soll `computeWeekday(21, 7, 1969)` den String `Monday` zurückgeben. Für diese Aufgabe sind folgende Informationen wichtig:

- Für die Berechnung gehen wir von dem fixen Tag 1.1.1900 aus, der ein Montag war. Um es einfach zu halten, wollen wir nur Wochentage nach dem 1.1.1900 berechnen.
- Nun können die Tage seit dem 1.1.1900 gezählt und modulo 7 gerechnet werden. Das Ergebnis ergibt dann den Wochentag (0 für Montag, ..., 6 für Sonntag). Die Anzahl der Tage ergibt sich aus:
  1. Anzahl der vollen Jahre multipliziert mit den Tagen im Jahr
  2. Anzahl aller Tage in den Monaten vor dem aktuellen Monat
  3. Die Tage im gegebenen Monat minus 1
- Schwierigkeiten bereiten jetzt noch die Schaltjahre, da sie 366 Tage haben. Ein Jahr ist ein Schaltjahr, wenn die Jahreszahl durch 4, nicht aber durch 100 teilbar ist. Jahre, deren Jahreszahl durch 400 teilbar ist, sind hingegen wieder Schaltjahre.

Aufgabe 7.2

1. Laden Sie das Beispielprojekt `MyProject`[1] von der Vorlesungsseite herunter. Kompilieren Sie die in `MyProject` enthaltenen Beispielprogramme mit Hilfe des `ant`-Buildsystemes (Siehe Foliensatz Tools).
2. Erstellen Sie Ihr eigenes Projekt `Info1Exercises`. Übernehmen Sie dafür die Struktur des Beispielprojektes. Fügen Sie die Klasse `Calendar` in das `src`- Verzeichnis ein und kompilieren Sie Ihr Projekt. Betrachten Sie die Ausgabe des `ant`-Buildsystemes und überprüfen Sie, ob alle Codekonventionen eingehalten wurden.

[1] http://ais/teaching/ss12/info/material/MyProject.zip

3. Schreiben Sie eine Klasse `CalendarTest`, die `JUnit`-Tests für die Methode `computeWeekday` enthält. Testen Sie die Methode für mindestens drei Verschiedene Tage.

Aufgabe 7.3

Betrachten Sie den Auszug der Klasse `GenericTuple`, die ein Tupel aus drei Objekten darstellt.

```
public class GenericTuple<A,B,C> {
  public void setA( ... ) { ... }
  public void setB( ... ) { ... }
  public void setC( ... ) { ... }

  public ... getA(){ ... }
  public ... getB(){ ... }
  public ... getC(){ ... }

  public String toString() {
    ...
  }

  A a;
  B b;
  C c;
```

1. Vervollständigen Sie die `set` - und `get`-Methoden der Klasse `GenericTuple`.
2. Vervollständigen Sie die `toString` Methoden der Klasse `GenericTuple`, so dass sie einen String der Form "`(···,···,···)`" ausgibt.
   Hinweis: Verwenden Sie die `toString`-Methoden der drei Member-Variablen.
3. Implementieren Sie eine Klasse `GenericTupleVector<A,B,C>`, die einen Vektor aus `GenericTuple<A,B,C>`-Elementen enthält.
4. Implementieren Sie für die Klasse `GenericTupleVector<A,B,C>` eine Methode `addTuple(GenericTuple<A,B,C> tuple)`, die ein Tupel in den Vektor einfügt.
5. Implementieren Sie für die Klasse `GenericTupleVector<A,B,C>` eine Methode `GenericTupleVector<A,B,C> compareB(B b)`, die einen Vektor zurückgibt, der alle Tupel `t` enthält, für die gilt `b.equals(t.getB())`
6. Implementieren Sie für die Klasse `GenericTupleVector<A,B,C>` eine Methode `int getSize()`, die die Anzahl der Elemente im Vektor zurückgibt.
7. Auf der Vorlesungshomepage finden Sie eine `JUnit`−Klasse `GenericTupleVectorTest.java`. Nutzen Sie diese, um ihre Klassen zu testen.