

Übung 01: Semesterplaner

Abgabetermin: 12. 3. 2009, 8:15

Name: _____

Matrikelnummer: _____

Informatik: G1 (Prähofer) G2 (Prähofer) G3 (Würthinger) G4 (Prähofer)

WIN: G1 (Khalil) G2 (Khalil) G3 (Schwinger)

Aufgabe	Punkte	abzugeben schriftlich	abzugeben elektronisch	korr.	Punkte
Übung 1	24	Prosabeschreibung, Java-Programm, Ausgabe eines Testlaufs	Java-Programm	<input type="checkbox"/>	

Übung 01: Semesterplaner (24 Punkte)

a) Semesterplaner

In dieser Übung soll ein Programmsystem bestehend aus mehreren Klassen realisiert werden, mit dem man die Lehrveranstaltungen eines Semesters planen und verwalten kann.

Ein Semester besteht aus mehreren Wochen (z.B. SS 2009 18 Wochen) und jede Woche aus den 7 Wochentagen. An jedem dieser Tage hat man eine Reihe von Terminen von Lehrveranstaltungen, die folgende Daten haben:

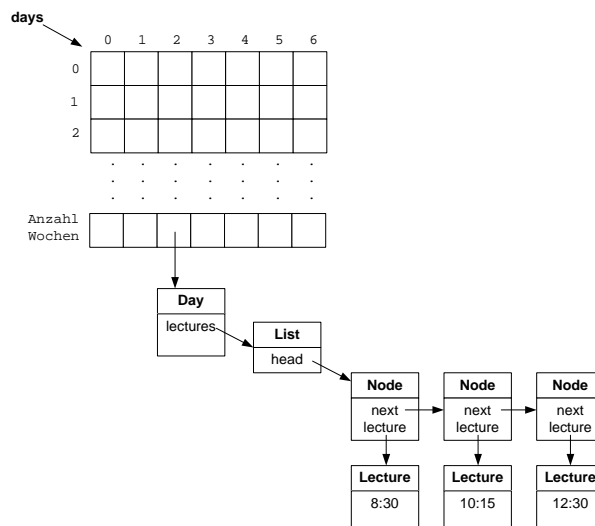
- Woche
- Wochentag
- Bezeichnung
- Beginnzeit
- Dauer

Eine Klasse `SemesterSchedule` dient zur Verwaltung der Termine eines Semesters:

- Ein Semester besteht aus einer bestimmten Anzahl von Wochen,
- jede Woche aus 7 Tagen und
- für jeden Tag soll eine Liste von Terminen von Lehrveranstaltungen nach der Zeit aufsteigend sortiert gespeichert werden können.

Es soll möglich sein, neue Termine hinzuzufügen, Termine zu löschen und auf die Termine eines Tages zuzugreifen. Implementieren Sie eine lineare Liste für die sortierte Verwaltung der LVAs eines Tages. Die LVAs sollen nach der Beginnzeit aufsteigend sortiert gespeichert werden. Verwenden Sie am besten ein zweidimensionales Array für die Tage (Dimension = Anzahl Wochen x 7 Tage pro Woche). Die folgende Abbildung zeigt schematisch diese Speicherstruktur.

Achten Sie bei dieser Aufgabe insbesondere auf **eine saubere Gestaltung der Schnittstellen der einzelnen Klassen**.



Hinweise und Anleitungen:

- Für Zeiten verwenden Sie die bereitgestellte Klasse `Time` im Package `datetime`. Diese erlaubt mit den Methoden `equals` und `compareTo` den Vergleich zweier `Time`-Objekte. `equals` bestimmt, ob zwei `Time`-Objekte die gleiche Zeit darstellen. `compareTo` liefert einen negativen, positiven oder 0-Wert, je nachdem ob das `Time`-Objekt kleiner, größer oder gleich dem als Parameter übergebenen Objekt ist. (Analoges gilt für `Date`-Objekte, siehe Zusatzaufgabe.)
- Bereitgestellt wird auch eine Klasse `DateTimeUtil`, die nützliche Funktionen für den Umgang mit Datum und Zeit enthält. So enthält die Klasse unter anderem Funktionen zur Bestimmung der Anzahl der Tage pro Jahr und Monat, zur Überprüfung der Gültigkeit von Datums- und Zeitangaben und für diverse Umrechnungen.

Vereinfachungen:

- Pro Tag soll es nur eine Lehrveranstaltung mit einem bestimmten Namen geben. Das heißt, Namen von Lehrveranstaltungen sind für einen Tag eindeutig.

Zusatzaufgabe (4 Zusatzpunkte):

- Ordnen Sie den einzelnen Tagen ein Datum zu. Dies ist eine freiwillige Zusatzaufgabe, für die es 4 Extrapunkte gibt.

b) *Benutzerdialog*

Schreiben Sie einen Dialog (nur textbasiert), mit dem ein Benutzer mit dem Semesterplaner arbeiten kann. Er kann über einen Operationscode eine Operation auswählen. Folgende Operationen sollen unterstützt werden:

- Eine LVA an einem bestimmten Wochentag, einer bestimmten Zeit und einer festgelegten Dauer für alle Wochen des Semesters definieren
- Eine bestimmte Woche auswählen
- Einen bestimmten Tag in der Woche auswählen
- Ausgeben der LVAs für den aktuellen Tag
- Ausgeben der nächsten LVA am aktuellen Tag nach einer gegebenen Zeit
- Eine neue LVA beim aktuellen Tag einfügen
- Eine LVA beim aktuellen Tag löschen
- Ausgeben der Liste alle Termine einer LVA im

Folgend ist ein Beispiel eines Benutzerdialogs angegeben.

Beispieldialog:

```

Semesterplan
=====
Folgende Operationen stehen zur Verfügung:
D - Definition einer LVA für das Semester
W - Bestimmen einer Woche
T - Bestimmen eines Tages der Woche
A - Ausgeben LVAs des aktuellen Tages
N - Ausgeben der nächsten LVA
E - Einfügen eines LVA beim aktuellen Tag
L - Löschen einer LVA des aktuellen Tages
S - Ausgeben der Semestertermine einer LVA
X - Programm beenden

Bitte Operation auswaehlen: D
Eingabe einer LVA
  LVA Name: Rechtsgrundlagen
  Wochentag (1 - 7): 2
  Bitte Zeit eingeben (hh:mm): 10:15
  Dauer: 90

Bitte Operation auswaehlen: W
  Woche: 3

Bitte Operation auswaehlen: T
  Wochentag (1 - 7): 2

Bitte Operation auswaehlen: A
Day: TUESDAY, week 3
  Rechtsgrundlagen: week 3, on TUESDAY, from 10:15 to 11:45

Bitte Operation auswaehlen: T
  Wochentag (1 - 7): 4

Bitte Operation auswaehlen: A

```

Day: THURSDAY, week 3
SWE2: week 3, on THURSDAY, from 08:30 to 10:00
SWE2UE: week 3, on THURSDAY, from 10:15 to 11:45

Bitte Operation auswaehlen: **N**
Nächste LVA für diesen Tag nach Zeit
Bitte Zeit eingeben (hh:mm): **10:00**
SWE2UE: week 3, on THURSDAY, from 10:15 to 11:45

Bitte Operation auswaehlen: **L**
Löschen einer LVA für diesen Tag
LVA Name: **SWE2UE**

Bitte Operation auswaehlen: **A**
Day: THURSDAY, week 3
SWE2: week 3, on THURSDAY, from 08:30 to 10:00

Bitte Operation auswaehlen: **S**
Termine einer LVA im Semester
LVA Name: **SWE2**
SWE2: week 1, on THURSDAY, from 08:30 to 10:00
SWE2: week 2, on THURSDAY, from 08:30 to 10:00
SWE2: week 3, on THURSDAY, from 08:30 to 10:00
SWE2: week 4, on THURSDAY, from 08:30 to 10:00
SWE2: week 5, on THURSDAY, from 08:30 to 10:00
SWE2: week 6, on THURSDAY, from 08:30 to 10:00
SWE2: week 7, on THURSDAY, from 08:30 to 10:00
SWE2: week 8, on THURSDAY, from 08:30 to 10:00
SWE2: week 9, on THURSDAY, from 08:30 to 10:00
SWE2: week 10, on THURSDAY, from 08:30 to 10:00
SWE2: week 11, on THURSDAY, from 08:30 to 10:00
SWE2: week 12, on THURSDAY, from 08:30 to 10:00
SWE2: week 13, on THURSDAY, from 08:30 to 10:00
SWE2: week 14, on THURSDAY, from 08:30 to 10:00
SWE2: week 15, on THURSDAY, from 08:30 to 10:00
SWE2: week 16, on THURSDAY, from 08:30 to 10:00
SWE2: week 17, on THURSDAY, from 08:30 to 10:00
SWE2: week 18, on THURSDAY, from 08:30 to 10:00

Bitte Operation auswaehlen: **X**

Hinweise:

Verwenden Sie die Methoden der beiden bereitgestellten Klassen In und Out für Eingabe und Ausgabe:

- `int In.readInt()` Einlesen eines Integer-Wertes
- `char In.readChar()` Einlesen eines Zeichens
- `String In.readWord()` Einlesen eines Wortes
- `boolean In.done()` Prüfen, ob letzte Einleseoperation erfolgreich war
- `Out.print(...)` Ausgabe eines Wertes
- `Out.println(...)` Ausgabe eines Wertes mit Leerzeile