

Assignment 01: Semester schedule

Due date and time: March 12, 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
Assignm 1	24	Text description, Java program, Test output	Java program	<input type="checkbox"/>	

Assignment 01: Semester schedule (24 points)

a) Semester schedule

In this assignment you should implement a program consisting of several classes to plan and administer your lectures during a semester. A semester consists of several weeks (e.g. summer semester 2009 has 18 weeks) and each week has 7 days. On each day you may have several lectures with the following properties

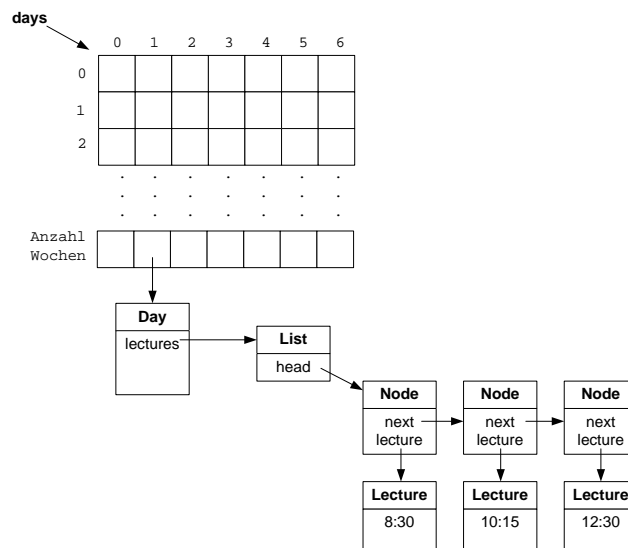
- week
- day of week
- name
- start time
- duration

The Java class `SemesterSchedule` shall be used for maintaining the lectures of a semester:

- a semester has a number of weeks.
- each week has 7 days
- for each day a list of lectures is maintained which is sorted according to the start time of the lecture

It should be possible to add new lectures, remove lectures on a particular day, and list the lectures of a day. You should implement a linear list for maintaining the sorted list of lectures on a day. You can use a 2-dimensional array for the days (dimension = number of weeks x 7 days per week). The following diagram shows this data structure.

Important in this assignment is a **clean design of the interfaces of the different classes**.



Instructions:

- For times use the class `Time` in the provided package `datetime` (see homepage). The class allows comparing times with methods `equals` and `compareTo`. Use `equals` to determine equality of two `Time` objects. Use `compareTo` to determine order of times, i.e., a negative value means smaller, zero means equal and a positive value means greater. (For date values use class `Date` which has similar semantics).
- Available is also a class `DateTimeUtil`, which contains useful methods for dealing with date and time values.

Simplifications:

- You can assume that per day there is only one lecture with a given name. That means, names of lectures are unique within a day.

4 Extra points:

- When you assign date values to the days of a semester you can earn 4 extra points.

b) Usage dialog

Write a text-based dialog to use the semester schedule implementation. The following interactive operations should be supported:

- Define a lecture for a particular day of week, a time, and duration for every week in the semester.
- Select a particular week in the semester
- Select a particular day of the current week
- List all the lectures on the selected day in the selected week
- Print out the next lecture after a given time at the current day
- Insert a new lecture for the current day
- Remove a lecture at the current day
- List all lectures with a given name in the semester

In the following a example dialogue is shown:

Sample dialogue:

```

Semester schedule
=====
Operations:
  D - Define a lecture for the semester
  W - Select a week
  T - Select a day in the week
  A - List the lectures of the current day
  N - Print out the next lecture
  E - Insert a new lecture for the current day
  L - Remove a lecture at the current day
  S - List all lectures with a given name in the semester
  X - Exit the program

Choose an operation: D
Input the lecture
  Name: Algebra
  Day of week (1 - 7): 2
  Start time (hh:mm): 10:15
  Duration: 90

Choose an operation: W
  Week: 3

Choose an operation: T
  Day of week (1 - 7): 2

Choose an operation: A
Day: TUESDAY, week 3
  Algebra: week 3, on TUESDAY, from 10:15 to 11:45

Choose an operation: T
  Day of week (1 - 7): 4

Choose an operation: 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

```

Choose an operation: **N**

Time (hh:mm): **10:00**

SWE2UE: week 3, on THURSDAY, from 10:15 to 11:45

Choose an operation: **L**

Name: **SWE2UE**

Choose an operation: **A**

Day: THURSDAY, week 3

SWE2: week 3, on THURSDAY, from 08:30 to 10:00

Choose an operation: **S**

Lectures during the semester

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

Choose an operation: **X**

Instructions:

Use methods of provided classes In and Out (see homepage) for input and output:

- `int In.readInt()` Read in an integer value
- `char In.readChar()` Read in a character
- `String In.readWord()` Read in one word
- `boolean In.done()` Test if last input operation was successful
- `Out.print(...)` Print out a value
- `Out.println(...)` Print out a value in a new line