Die Aufgaben müssen von jedem Teilnehmer einzeln bearbeitet und abgegeben werden. Die Abgabe muss **bis spätestens Montag 02.06.2014 um 23:59 Uhr** über unser Hausaufgabenverwaltungssystem **https://se.cs.uni-kassel.de/hms/** erfolgen. Die Abgabe ist nur als einzelne *.zip oder *.jar-Datei möglich. Daher müssen alle für eine Abgabe relevanten Daten zu **einem** solchen Archiv kombiniert werden.

Hinweise zur Abgabe:

- Die Hausaufgabe ist in Form von **zwei** exportierten Eclipse Projekten abzugeben. Mit Hilfe der Eclipse Export Funktion ist es möglich mehrere Projekte in eine zip-Datei zu exportieren. Ist ein Projekt nicht korrekt exportiert, kann es bei der Korrektur nicht berücksichtigt werden (es bietet sich also an, den Import des exportierten Projektes auszuprobieren).

  WICHTIG: Benennen Sie ihre Projekte für diese Abgabe nach folgendem Schema:

  DPSS14_HA<a>_<b>_<Matrikelnummer>,

  wobei <a> für die aktuelle Hausaufgabe und <b> für die Aufgabennummer steht.
  Beispiel für Aufgabe 1:
  DPSS14_HA6_1_12345678.

Allgemeines
Orientieren Sie sich für die Lösung der Aufgaben an den zugehörigen Übungen und Vorlesungen:
http://seblog.cs.uni-kassel.de/category/currentterm/design-patterns2014/

Die Benotung ergibt sich aus den Hausaufgaben, wobei eine Abgabe ausgelassen werden darf. Für die Note wird die nicht abgegebene Abgabe, beziehungsweise die Abgabe mit der geringsten Prozentzahl, nicht beachtet. Die Endnote ergibt sich aus dem Mittelwert der erreichten Prozentpunkte der übrigen Abgaben.

Die Veranstaltung gilt als nicht bestanden, wenn mehr als **eine Hausaufgabe nicht abgegeben** wurde oder der Mittelwert der zur Benotung herangezogenen Abgaben **weniger als 50%** beträgt.

Fragen bitte an Stefan.Lindel@cs.uni-kassel.de oder george@uni-kassel.de

Aufgabe 1 Flyweight Pattern

Implementieren Sie ein Datenmodell für ein Grafikprogramm (entweder JavaFX **oder** SWT) welches in der Lage ist Kreis, Linien und Rechtecke zu zeichnen. Hierfür ist das Flyweight Pattern zu verwenden, um gemeinsame Attribute der Standardform zu speichern.

- Bilden Sie dabei folgende Teile des Systems ab:
  - Circle
  - Rectangle
  - Style
  - Line
  - Pane
- Schreiben Sie einen Junit-Test in welchem
  - eine sinnvolle Objektstruktur erzeugt wird
  - das Grafikprogramm einen Kreis, zwei Linien und drei Rechtecke zeichnet und anschließend die Füllung der Rechtecke vom Standard (ohne Füllung) auf schwarz ändert
  - sinnvolle Assertions verwendet werden

Fragen bitte an Stefan.Lindel@cs.uni-kassel.de oder george@uni-kassel.de

Aufgabe 2 Command Pattern

Implementieren Sie ein Fussballfeld unter Verwendung des Command Pattern.

- Bilden Sie dabei folgende Teile des Systems ab:
  - Field
  - Player
  - Ball
  - Team
  - Goal
- Implementieren Sie die Command Klassen:
  - Undo und Redo
- Es reicht aus, das Spiel komplett ohne Oberfläche zu programmieren werden und über die Konsole, wie unten zu sehen, den momentanen Zustand ausgeben zu lassen
- Die Abgabe ist so aufzubauen, dass das Field
  - den aktuellen Spieler mit `getCurrentPlayer():Player` zurückgibt auf diesem die Methoden `left()`, `right()`, `top()`, `down()` aufrufbar sind. Weiterhin muss der Player mit `getX():int` und `getY():int` seine aktuelle Position zurückgeben
  - die Ballposition mit `getBallX():int` und `getBallY():int` zurückgeben kann
  - die Methoden `undo()` und `redo()` sollen von außen aufrufbar sein
- Schreiben Sie einen Junit-Test in welchem
  - eine sinnvolle Objektstruktur erzeugt wird
  - die aktuelle unten angegebene Situation unten erstellt wird
  - sinnvolle Assertions verwendet werden

```
##########################
#                        #
#                        #
G                        G
G          P*            G
#                        #
#                        #
##########################
```

Fragen bitte an Stefan.Lindel@cs.uni-kassel.de oder george@uni-kassel.de