Die Aufgaben können (müssen aber nicht) in Gruppen von bis zu drei Leuten bearbeitet werden. Falls die Aufgaben in einer Gruppe bearbeitet werden, genügt **eine** Abgabe mit den Namen aller Gruppenmitglieder. Abgabe bis **spätestens Mittwoch 26.05.2010** per Mail mit dem Betreff **PMSS2010 HA5 <Matrikelnummer>** (z. B. „PMSS2010 HA5 12345678“) an **pm@cs.uni-kassel.de**. Für diese Hausaufgabe gibt es 19 + 8 Punkte.

Hinweise zur Abgabe:

- Aufgabe 1 und die Zusatzaufgabe als exportierte Eclipse Projekte. Falls Sie mehrere Projekte anlegen, können diese alle in **eine** .zip Datei gepackt werden (erledigt die Eclipse Export Funktion automatisch!). Sind die Projekte nicht korrekt exportiert, können diese bei der Korrektur nicht berücksichtigt werden (es bietet sich also an, den Import des exportierten Projekts auszuprobieren).

  WICHTIG Bennen Sie ihre Projekte nach folgendem Schema:

  ```
  PMSS2010 HA5 A<i> <Matrikelnummer>,
  ```

  wobei <i> für die Aufgabennummer steht. Beispiel:

  ```
  PMSS2010 HA5 A1 12345678.
  ```

- Falls die Zusatzaufgabe bearbeitet wird, diese als PDF abgeben. Andere Formate führen zu Punktabzug!

Vorbereitung

Für die Bearbeitung der Hausaufgabe 5 benötigen Sie eine Implementierung des Klassendiagramms von Mancala. Sie können entweder ihre eigene Implementierung aus Hausaufgabe 4 verwenden oder eine fertige Version vom Blog zur Veranstaltung herunterladen (`http://seblog.cs.uni-kassel.de/category/currentterm/pm/`) und als Projekt in Eclipse importieren.

Darüber hinaus benötigen Sie das *eDOBS*-PlugIn für Eclipse. Installieren Sie das PlugIn wie in der Übung gezeigt über den Eclipse Update Mechanismus. Die URL für die Update-Site lautet: `http://www.se.eecs.uni-kassel.de/se/fileadmin/se/projects/eDOBS/update`.

Aufgabe 1 - Implementierung (5P)

Implementieren Sie die Methode

`initGame(firstPlayerName:String,secondPlayerName:String):void`

Sie soll das Spiel initialisieren und folgende Objekte erstellen:

- Zwei Spieler mit den als Parameter übergebenen Namen
- Alle 12 Löcher
- Vier Steine für jedes Loch
- Zwei Kalahs

Vergessen Sie nicht alle Objekte korrekt zu verlinken. Falls Sie nicht die im Blog zur Verfügung gestellte Implementierung verwenden, fügen Sie ihrer Implementierung eine Methode mit obiger Signatur hinzu.

Aufgabe 2 - eDOBS (4P)

Setzen Sie am Ende der Methode `initGame(...):void` einen Breakpoint und fertigen Sie einen Screenshot von eDOBS an, indem die aktuelle Objektstruktur nach Ausführung der `initGame(...):void` Methode zu sehen ist. Im Screenshot sollte mindestens folgendes zu sehen sein:

- Ein Mancala Objekt
- Zwei Player Objekte
- 12 Pit Objekte
- Zwei Kalah Objekte
- 12 Stone Objekte (hier genügt jeweils ein Stone pro Pit)

sowie die Links zwischen den Objekten. Hinweise zur Bedienung des eDOBS finden sich auf den Folien zur Übung.

Aufgabe 3 - Zetteltest, eDOBS

Die Klasse `Pit` soll eine Methode `moveStones():void` besitzen, welche alle Steine aus dem Pit auf dem die Methode aufgerufen wird entfernt und auf die nächsten Pits verteilt. Die Methode soll zunächst keine weiteren Regeln berücksichtigen! Eine Beispielimplementierung ist in Listing 1 abgedruckt.

```
1  public void moveStones ()
2  {
3    Pit nextPit = getNext();
4    for(Iterator<Stone> iter = getStones().iterator(); iter.
         hasNext();)
5    {
6      Stone stone = iter.next();
7      nextPit.addToStones(stone);
8      nextPit = nextPit.getNext();
9    }
10 }
```

Listing 1: Beispielimplementierung der Methode Pit::moveStones():void

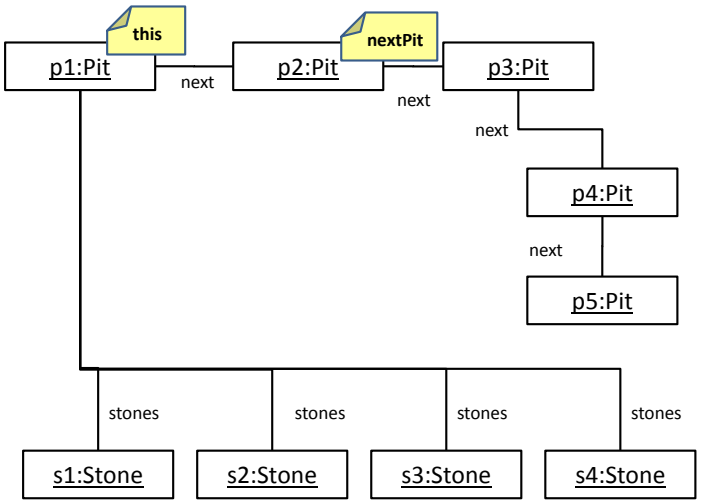
Falls Sie die im Blog zur Verfügung gestellte Implementierung verwenden, ist diese Methode in bereits implementiert.

3.1 - Zetteltest (8P)

Fertigen Sie einen Zetteltest zur Ausführung der Methode `moveStones():void` an. Nutzen Sie hierfür die Tabelle auf der nächsten Seite. Die Objektstruktur zu Beginn ist bereits vorgegeben. Zeichnen Sie für jeden Schritt die vollständige Objektstruktur **nach** einem Durchlauf der Schleife. Falls Sie die in Listing 1 verwenden, ist die Schleife in den Zeilen 4-10 gemeint.

3.2 - eDOBS (2P)

Setzen Sie einen Breakpoint an das Ende der Methode `moveStones():void` und fertigen Sie einen Screenshot der eDOBS Objektstruktur an.

Schritt	Objektstruktur
1	
2	
3	
4	
5	

Zusatzaufgabe - Implementierung erweitern (8P)

Schreiben Sie einen JUnit Test, der die Einhaltung einer weiteren Mancala Regel beim Aufruf der Methode `Pit::moveStones():void` überprüft: Falls der letzte Stein in ein leeres Pit fällt und sich im gegenüber liegenden Pit (d.g. auf Seiten des Gegners) Steine befinden, so dürfen diese dem eigenen Kalah hinzugefügt werden.

Erweitern Sie anschließend ihre Implementierung der Methode `Pit::moveStones():void` die eben diese Regel. Fertigen Sie einen Screenshot der eDOBS Objektstruktur *vor* dem Aufruf `p1.moveStones()` (Startsituation) und *nach* dem Aufruf (Endsituation) der Methode an. Erzeugen Sie in ihrem Test die in Abbildung 1 dargestellte Objektstruktur:

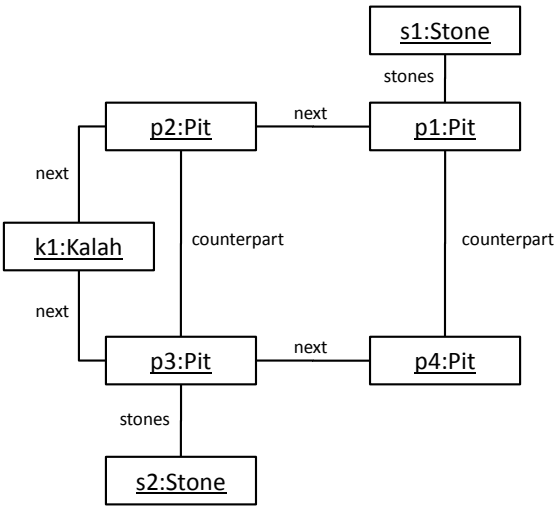


Abbildung 1: Zu erzeugende Objektstruktur

Die Abgabe für diese Aufgabe beinhaltet also:

1. Den JUnit Test
2. Die Erweiterung der Methode `moveStones():void`.
3. Zwei Screenshots vom eDOBS.