ALBERT-LUDWIGS-
UNIVERSITÄT FREIBURG

Dr. Jochen Hoenicke
Alexander Nutz

17.6.2014
Abgabe: Montag 23.6.2014
via Daphne/Subversion

7. Übungsblatt zur Vorlesung
Einführung in die Informatik

Aufgabe 1: Verkettete Listen

Schreiben Sie Klassen für eine einfach verkettete *selbstsortierende* Liste (`SortedSinglyLinkedList.java`) und für eine doppelt verkettete selbstsortierende Liste (`SortedDoublyLinkedList.java`). Jedes Element der Liste soll einen `Comparable`-Objekt speichern. Überschreiben Sie für jede Liste die `toString()`-Methode aus `Object`. Selbstsortierend heißt hier, dass die Methode `insert(..)` ein Element so in die Liste einfügt, dass diese sortiert ist. Beide Klassen sollen das Interface `SortedLinkedList` implementieren, das auf der Vorlesungswebsite online steht.

`Comparable` ist ein Interface der Java Standardbibliothek. Es schreibt vor, das jede Implementierung eine Methode `int compareTo(Object o)` zur Verfügung stellen muss. `x.compareTo(y)` liefert einen positiven Wert, falls `x` (in einem von der Implementierung vorgegebenen Sinn) größer als `y` ist, einen negativen Wert, falls es kleiner ist, und 0, falls beide gleich sind.

Benutzen Sie folgende Klasse, um Ihre Implementierung zu testen. (Datei steht online.)

```
public class SortedLists {
  public static void main(String[] args) {
    SortedLinkedList sll = null;
    for (int i = 0; i < 2; i++) {
      if (i == 0)
        sll = new SortedSinglyLinkedList();
      else
        sll = new SortedDoublyLinkedList();
      sll.insert("Alle");
      sll.insert("meine");
      sll.insert("Entchen");
      sll.insert("schwimmen");
      sll.insert("auf");
      sll.insert("dem");
      sll.insert("See");
      System.out.println(sll);
      System.out.println("---");
    }
  }
}
```

Aufgabe 2: Alapo (2) – Spielzüge

Im `doc/public`-Ordner im SVN von Daphne finden Sie eine Implementierung von Alapo für zwei menschliche Spieler. Allerdings fehlt in der Klasse `Board.java` noch die Methode `public Board makeMove(Move move, Player player)`. Diese soll den übergebenen Zug auf Gültigkeit prüfen und, falls er gültig ist, ein neues `Board`-Objekt mit der neuen Stellung zurückgeben. Falls der Zug nicht gültig ist, soll `null` zurückgegeben werden.

Implementieren Sie die fehlende Gültigkeitsprüfung für Züge. Anschließend sollte Alapo spielbar sein, wobei noch die Gewinnbedingung fehlt (und die Benutzeroberfläche eher rudimentär ist).

Sie können gerne Ihre eigene Implementierung des Spielbretts aus dem letzten Übungsblatt verwenden und auch sonst die vorgegebene Implementierung nach Ihren Vorlieben anpassen. Für die zukünftige Kompatibilität wichtig sind hauptsächlich die Klassen `Player`, `Move` und das Interface von `Board`.