Übungsbeispiele – PR1

1 Event-Kalender

Erstellen Sie eine Klasse *Event* sowie eine Klasse *EventKalender,* zur Verwaltung von Events.

Ein *Event* hat die folgenden Eigenschaften: Title (String), Ort (String), Eintrittspreis (double). Erstellen Sie dafür Getter und Setter. Erstellen Sie auch eine Methode *public String toString*, welche diese Attribute als String retourniert.

Ein *EventKalender* kann über die Methode *add* Events aufnehmen, die in einer privaten *ArrayList<Event> events* abgelegt werden. Spendieren Sie dem EventKalender die folgenden Methoden und testen Sie diese mit einer ausführbaren Klasse.

```
public Event getByTitle(String title)
```

Liefert das erste Event, das den übergebenen Titel aufweist retour

```
public ArrayList<Event> getByOrt(String ort)
```

Liefert alle Events, die im übergebenen Ort stattfinden, retour

```
public ArrayList<Event> getByEintrittsPreis(double min, double max)
```

Liefert alle Events, deren Eintrittspreise zwischen den übergebenen Werten min und max liegen bzw. genau einem dieser beiden Werte entsprechen.

```
public Event getMostExpensiveByOrt(String ort)
```

Liefert das Event mit dem höchsten Eintrittspreis im übergebenen Ort. Falls mehrere Events den selben Preis im übergebenen Ort aufweisen können soll eines (z.B. das erste gefunden) zurückgeliefert werden.

```
public double getAvgPreisByOrt(String ort)
```

Liefert den durchschnittlichen Eintrittspreis der Events im übergebenen Ort

2 Pegel-Verwaltung

Erstellen Sie eine Klasse *Wasserstand* sowie eine Klasse *WasserstandManager*.

Ein Wasserstand hat die folgenden Attribute: Id (int), GewaesserName (String), messWert (double), messWertFuerAlarmierung (double), Zeitpunkt (int; Sekunden seit 1. 1. 1970). Erstellen Sie für diese Attribute Getter und Setter. Erstellen Sie auch eine Methode *public String toString*, welche diese Attribute im Rahmen eines Strings retourniert.

Der *WasserstandManager* soll mit einer Methode *add* beliebig viele Wasserstand-Objekte aufnehmen können. Diese soll er in einer *privaten ArrayList<Wasserstand>* ablegen. Darüber hinaus soll er die folgenden Methoden anbieten:

```
public Wasserstand findById(int id)
```

Liefert den Wasserstand mit der übergebenen Id.

```
public ArrayList<Wasserstand> findAllByGewaesser(String gewaesserName)
```

Liefert alle Wasserstände des Gewässers mit dem übergebenen Namen.

```
public Wasserstand findNewestWasserstandForGewaesser(String gewaesserName)
```

Liefert den neuesten Wasserstand (= jener mit dem höchsten Integer Wert für Zeitpunkt) für das Gewässer mit dem übergebenen Namen.

```
public Wasserstand findOldestWasserstandForGewaesser(String gewaesserName)
```

Liefert den ältesten Wasserstand (= jener mit dem niedrigsten Integer Wert für Zeitpunkt) für das Gewässer mit dem übergebenen Namen.

```
public ArrayList<Wasserstand> findForAlarmierung()
```

Liefert alle Wasserstände, deren aktueller Wasserstand höher oder gleich wie dessen Wert messWertFuerAlarmierung ist

```
public ArrayList<Wasserstand> findByZeitspanne(int von, int bis, String
gewaesserName)
```

Liefert die Wasserstände für das Gewässer mit dem übergebenen Namen, die zwischen oder zu den Zeitpunkten von und bis gemessen wurden.

3 Schleifen und Arrays

Erstellen Sie eine Klasse `TicTacToeHelper` inkl. `main-Methode`. Ihre Aufgabe ist es, die folgenden Hilfsmethoden zu implementieren, welche für das bekannte Spiel Tic Tac Toe verwendet werden können.

a) `public static char[][] generateEmptyBoard(int size)`

Diese Methode soll ein 2d-Feld vom typ char[][] mit den Dimensionen size x size erzeugen, mit dem Space-Character (= Leerzeichen ‘ ‘) initialisieren und schließlich zurückliefern.

b) `public static boolean validateBoard(char[][] board, int size)`

Diese Methode soll ein 2d-Feld vom typ char[][] entgegennehmen und überprüfen, ob es sich um ein Feld mit den Dimensionen size x size handelt. Falls die Dimensionen OK sind, liefert sie *true* zurück ansonsten *false*.

c) Verwenden Sie die Methode von a) um in der main-Methode damit eine Variable char[][] myBoard mit einem leeren Spielfeld der Größe 3 x 3 zu initialisieren. Stellen Sie durch geeignete Zuweisungsoperationen auf die jeweiligen Indizes des leeren 2d-Feldes die folgende Spielsituation mit den chars ‘X’ bzw. ‘O’ nach.

X	X	O
X		O
O		X

d) `public static void printBoard(char[][] board)`

Diese Methode soll ein 2d-Feld beliebiger Dimensionen vom typ char[][] entgegennehmen und auf der Konsole wie folgt anzeigen können. Verwenden Sie in der Zeile “\t” (=Tabulator) als Trennzeichen. Um das Ergebnis zu überprüfen, rufen Sie aus der main-Methode *printBoard(myBoard)* mit der unter c) erstellen Spielsituation auf. Sie sollten den folgenden Output auf der Konsole sehen.

```
X	X	O
X		O
O		X
```