Bereich: Aufzählungstypen

Monate

Package: `de.dhbwka.java.exercise.enums`	**Klasse:** `Months`

Aufgabenstellung:

Entwerfen Sie einen Aufzählungstyp `Months`, der für jeden Monat dessen Namen, die Anzahl der Tage (der Februar hat in dieser Aufgabe immer 28 Tage) und die alten deutschen Monatsnamen kennt!

Monat	***Tage***	***Altdeutsche Namen***
Januar	31	Hartung, Eismond
Februar	28	Hornung, Schmelzmond, Taumond, Narrenmond, Rebmond, Hintester
März	31	Lenzing, Lenzmond
April	30	Launing, Ostermond
Mai	31	Winnemond*, Blumenmond
Juni	30	Brachet, Brachmond
Juli	31	Heuert, Heumond
August	31	Ernting, Erntemond, Bisemond
September	30	Scheiding, Herbstmond
Oktober	31	Gilbhart, Gilbhard, Weinmond
November	30	Nebelung, Windmond, Wintermond
Dezember	31	Julmond, Heilmond, Christmond, Dustermond

**Weidemonat: heute zu Wonnemonat umgedeutet*

Geben Sie im Hauptprogramm die Daten zum aktuellen Monat aus, z.B.

```
Der Juni hat 30 Tage und hieß früher 'Brachet, Brachmond'
```

Hinweis:
Den aktuellen Monat bekommen Sie mit

```
new Date().getMonth()
```

oder seit JDK 1.1 besser:

```
Calendar.getInstance().get(Calendar.MONTH)
```

als `int` (Zählung beginnt bei 0).

Bereich: Aufzählungstypen

Kartenspiel

Package: `de.dhbwka.java.exercise.enums.cards`	**Klasse:** `CardGame`

Aufgabenstellung:

Implementieren Sie einige Grundlagen (Basisklassen) für ein Kartenspiel.

Schreiben Sie eine Klasse `PlayingCard`. Eine Spielkarte besteht dabei aus einer Farbe (Karo, Herz, Pik, Kreuz) sowie einem Kartenwert (7, 8, 9, Bube, Dame, König, 10, Ass).
Für die Farbe (`Suit`) und den Kartenwert (`CardValue`) können Sie Aufzählungstypen (`enum`) verwenden.

Hinweis:
Farben in Englisch: Diamonds (Karo), Heart (Herz), Spade (Pik), Clubs (Kreuz)
Werte in Englisch: Ace (Ass), King (König), Queen (Dame), Jack (Bube)

Die Klasse `PlayingCard` sollte folgende Methoden haben:

- `String toString()` – gibt die Spielkarte als String aus (Farbe und Kartenwert)
- `int compareTo(PlayingCard c)` – vergleicht eine Spielkarte mit einer anderen (positiver/ negativer Rückgabewert entsprechend der Reihenfolge der Farben und Kartenwerte oben).
  Hinweis: Implementieren Sie dazu `Comparable<PlayingCard>`

Schreiben Sie eine Klasse `CardGame`, deren Objekte ein Kartenspiel (ein Stapel Karten) enthalten.
Die Klasse `CardGame` soll folgende Methoden haben:

- Konstruktor `CardGame()` – erzeugt ein neues komplettes (ungemischtes) Kartenspiel (Stapel mit 32 Karten)
- `void shuffle()` – mischt die Karten im Kartenspiel (Stapel)
- `void sort()` – sortiert die Karten im Stapel aufsteigend (auch wenn schon welche entfernt wurden!) nach Farben und innerhalb der Farben nach Kartenwert
- `PlayingCard get()` – gibt die „oberste“ Karte des Spiels (Stapels) zurück und entfernt sie aus dem Stapel
- `List<PlayingCard> all()` – liefert eine Liste der (verbliebenen) Karten im Stapel

Schreiben Sie außerdem eine Klasse `TestGame`, die

- ein neues Kartenspiel (Stapel) erzeugt,
- es mischt,
- nacheinander 10 Karten vom Stapel entnimmt, ausgibt (auf Bildschirm) und sie dabei mit der „Herz 7“ vergleicht (auch Vergleichsergebnis ausgeben),
- die restlichen Karten sortiert
- und auflistet (am Bildschirm)!

Bereich: Aufzählungstypen

Sortierkriterien

Package: `de.dhbwka.java.exercise.enums.library`	**Klasse:** `Attributes`

Aufgabenstellung:

Bauen Sie die Musterlösung der Aufgabe „Bücherei“ (s. Datenstrukturen) derart um, dass die folgenden Abschnitte durch einen geeigneten Aufzählungstyp `Attributes` ersetzt werden! Natürlich muss auch der übrige Code an einigen Stellen angepasst werden, z.B. der Typ des `order`-Feldes in `BookComparator`.

Diesen Teil aus der Klasse `Book` ersetzen:

```
public static final int TITLE = 0;
public static final int AUTHOR = 1;
public static final int YEAR = 2;
public static final int PUBLISHER = 3;
public static final int[] CRITERIA = {TITLE, AUTHOR, YEAR, PUBLISHER};
```

Diesen Teil aus der Klasse `Library` ersetzen:

```
private static final String[] orderCriteria = {"Titel","Autor","Jahr","Verlag"};
```

Zusatzaufgabe:
Ersetzen Sie auch das Array `inputFields` durch eine geeignete Collection-Klasse und adressieren Sie die `JTextField`-Komponenten mit Instanzen der Aufzählung `Attributes`.