HTWK Leipzig, FIMN
Dipl.-Math. Dörte König

Anwendungsorientierte Programmierung II
Hinweise zur Übung 4

1. Klassen

Klasse Card: enthält den „Bauplan“ für ein Karten-Objekt mit allen Eigenschaften (Attributen) und Methoden wie im Aufgabenblatt angegeben
Getter, Setter und `toString()`-Methode können über das `Source`-Menue von Eclipse automatisch erzeugt werden

Klasse Main: enthält die `main()`-Methode sowie folgende statische Methoden für die Funktionalitäten:

`genCardDeck()`	Erzeugung des Skatblattes
`showCardDeck()`	Ausgabe des Skatblattes
`shuffleCardDeck()`	Mischen des Skatblattes
`dealCards()`	Verteilen der Karten an Skat und Spieler

2. Statische Methoden

Klassenmethoden
exisitieren unabhängig von einer konkreten Objektinstanz
Deklaration mit Schlüsselwort `static`

3. Test der Klasse Card

Bevor Sie mit der eigentlichen Aufgabe beginnen, können Sie die erstellte Klasse `Card` und ihre Methoden in der `main()`- Methode testen:

```
Card card = new Card();           //Konstruktoraufruf, Objekterzeugung
card.setFarbe("Pik");             //Aufruf der set-Methoden
card.setWert("Dame");
System.out.println(card);         //Ausgabe über die toString()-Methode
```

Entfernen Sie die Anweisungen anschließend wieder, da sie für den weiteren Programmablauf nicht gebraucht werden.

4. Erzeugung des Skatblattes

```
Card[] skatspiel = new Card[32];
```

erzeugt ein Array mit 32 `Card`-Objekten für das Skatblatt.
Farbe und Wert für jedes `Card`-Objekt werden mit der vorgegebenen Methode `genCardDeck()` gesetzt:

```
genCardDeck(skatspiel);
```

Arrays werden in Java immer als **Referenzparameter** an Methoden übergeben. D.h. Änderungen am Array innerhalb einer Funktion werden direkt auf den referenzierten Speicherbereich geschrieben und bleiben damit nach Beendigung der Methode erhalten.

5. **Kartenblätter für Spieler und Skat**
   Analog zum Kartenspiel sind die Kartenblätter der Spieler, sowie der Skat als Felder anzulegen:

```
//Spieler
Card[] spieler1 = new Card[10];
...

//alternativ:
//Spielermatrix
Card[][] spieler = new Card[3][10];

//Skat
Card[] skat = new Card[2];
```

6. **Simulation des Mischvorgangs**

   Die Anzahl der Mischvorgänge ist als ganze Zufallszahl in einem selbst zu wählenden Bereich zu erzeugen. Erzwingen Sie eine **Mindestanzahl von Mischvorgängen**!

   Bei jedem **Mischvorgang** sind zwei Kartenobjekte zu vertauschen, deren Indexe ebenfalls zufällig zu generieren sind. Beachten Sie, dass der Random-Bereich mit der Feldlänge korrelieren muss! Tauschen Sie komplette Kartenobjekte und nicht deren Eigenschaften!

7. **Verteilung der Karten an Spieler und Skat**
   Die Verteilung der Karten des Skatblattes an Spieler und Skat soll über eine statische Methode **`dealCards()`** realisiert werden. Dieser Methode sind die Arrays zu Kartenblatt, Spielern/Spieler-matrix und Skat als Referenzparameter zu übergeben.

```
public static void dealCards( Card[]deck,
                              Card[] s1,
                              Card[] s2,
                              Card[] s3,
                              Card[] sk) {}
```

   Beachten Sie, dass in jeder Verteilungsrunde Karten-"Päckchen" zu drei bzw. vier Karten (wie im Aufgabenblatt beschrieben) zu verteilen sind. Realisieren Sie jede Verteilungsrunde in möglichst kompakter Form.

   Beispiel für die Verteilung der ersten neun Karten in Runde1:

```
//Runde1
for(int i=0;i<9;i++)
{
      if(i<3)                                  //deck - Skatblatt
            s1[i]=deck[i];                     //s1   - Spieler1
      else
            if(i<6)
                  s2[i-3]=deck[i];             //s2   - Spieler2
            else
                  s3[i-6]=deck[i];             //s3   - Spieler3
}
```

   Alternative Lösungen sind möglich.