ÜBUNG 2

24WS-UE02-A01 Eigenschaften von Zahlen

Erstellen Sie ein Java-Programm, welches eine ganze Zahl (Datentyp int) vom Benutzer einliest und auf der Konsole ausgibt, ob die eingegebene Zahl gerade oder ungerade ist, sowie ob die Zahl negativ oder positiv ist.
Hinweis: Die Zahl 0 ist gerade aber weder positiv noch negativ.

Beispieldialoge:

```
Enter number: 0
0: even
Enter number: 1
1: odd positive
Enter number: 2
2: even positive
Enter number: 3
3: odd positive
Enter number: 99
99: odd positive
Enter number: 1000000
1000000: even positive
Enter number: -1
-1: odd negative
Enter number: -2
-2: even negative
Enter number: -99
-99: odd negative
Enter number: -99999999
-99999999: odd negative
```

24WS-UE02-A02 Gleichheit von Zahlen

Schreiben Sie ein Java-Programm, das vier ganze Zahlen (Datentyp `int`) a, b, c und d vom Benutzer einliest und prüft und das Ergebnis auf der Konsole ausgibt:

- ob a, b, c und d vier gleiche Werte enthalten
- ob a, b, c und d drei gleiche Werte enthalten
- ob a, b, c und d zwei gleiche Werte enthalten
- ob a, b, c und d zwei Wertepaare enthalten
- ob a, b, c und d verschiedene Werte enthalten

Beispieldialog:

```
a = 3
b = 3
c = 3
d = 3
 -> Alle Werte sind gleich.
a = 3
b = 3
c = 3
d = 4
 -> Drei Werte sind gleich.
a = 3
b = 3
c = 4
d = 5
-> Zwei Wertepaare sind gleich.
a = 3
b = 4
c = 5
d = 6
 -> Alle Werte sind verschieden.
```

24WS-UE02-A03 Summe Minimum Maximum Mittelwert mehrerer Zahlen

Erstellen Sie ein Java-Programm, welches positive Zahlen (Datentyp `int`) vom Benutzer einliest.
Verwenden Sie zum Einlesen der Zahlen eine `while`-Schleife.
Die Eingabe einer negativen Zahl bricht den Vorgang ab.
Abschließend werden die Anzahl, die Summe, das Minimum, das Maximum und der Mittelwert der eingelesenen Zahlen berechnet und ausgegeben. Beachten Sie, dass der Mittelwert eine Gleitkommazahl ist.

Beispieldialog:

```
1. Zahl eingeben: 3
2. Zahl eingeben: 2
3. Zahl eingeben: 7
4. Zahl eingeben: -1
#Zahlen: 3
Summe: 12
Minimum: 2
Maximum: 7
Mittelwert: 4.0
1. Enter number: -1
#Zahlen: 0
1. Zahl eingeben: 4
2. Zahl eingeben: 7
3. Zahl eingeben: 6
4. Zahl eingeben: -1
#Zahlen: 3
Summe: 17
Minimum: 4
Maximum: 7
Mittelwert: 5.666666666666667
```

24WS-UE02-A04 Taschenrechner

Ein Taschenrechner soll einer Benutzerin erlauben, mehrere Berechnungen durchzuführen. Dazu werden jeweils eine (Gleitkomma-)Zahl, ein Operator (+ - * /) sowie eine zweite (Gleitkomma-)Zahl eingegeben.
Das Ergebnis der Berechnung wird dann ausgegeben und die Benutzerin gefragt, ob sie eine weitere Berechnung durchführen möchte, siehe Beispieldialog.

Stellen Sie in Ihrem Programm sicher, dass von der Benutzerin jeweils nur gültige Werte für die Operatoren eingegeben werden. Gibt die Benutzerin einen ungültigen Wert ein, muss sie ihre Eingabe solange wiederholen, bis ein gültiger Wert eingegeben wird.

Beispieldialog:

```
Taschenrechner

1. Zahl: 17
Operator: +
2. Zahl: 4
17.0 + 4.0 = 21.0
Fortsetzen (j/n)? n

bye bye!
Taschenrechner

1. Zahl: 2
Operator: +
2. Zahl: 3
2.0 + 3.0 = 5.0
Fortsetzen (j/n)? j

1. Zahl: 33.4
Operator: /
2. Zahl: 22.5
33.4 / 22.5 = 1.4844444444444445
Fortsetzen (j/n)? j

1. Zahl: 33
Operator: *
2. Zahl: 2.5
33.0 * 2.5 = 82.5
Fortsetzen (j/n)? n

bye bye!
Taschenrechner

1. Zahl: 17
Operator: (
Ungültiger Operator! Gültige Operatoren sind + - * / : -
2. Zahl: 4
17.0 - 4.0 = 13.0
Fortsetzen (j/n)? n

bye bye!
```

24WS-UE02-A05 Wiederholte einfache Rechnungen

Erstellen Sie ein einfaches Java-Programm, welches diverse Berechnungen durchführt.
Lesen Sie dazu zunächst eine ganze Zahl (Datentyp int) vom Benutzer ein.
Dann lesen Sie einen Operator ein. Die Operatoren + - * / und % erfordern das Einlesen einer weiteren Zahl.
Dann wird das Zwischenergebnis ausgegeben.
Der Operator = soll die Berechnungen beenden. Es wird noch das Endergebnis ausgegeben.

Auchten Sie wiederum darauf, dass nur gültige Operatoren eingegeben werden können.

Beispieldialog:

```
Ganze Zahl: 17
Zwischenergebnis: 17
Operator (+ - * / % =): +
Ganze Zahl: 4
Zwischenergebnis: 21
Operator (+ - * / % =): =
Endergebnis: 21
Ganze Zahl: 33
Zwischenergebnis: 33
Operator (+ - * / % =): &
Ungültiger Operator! Gültige Operatoren sind + - * / % = : =
Endergebnis: 33
Ganze Zahl: 4
Zwischenergebnis: 4
Operator (+ - * / % =): +
Ganze Zahl: 3
Zwischenergebnis: 7
Operator (+ - * / % =): -
Ganze Zahl: 2
Zwischenergebnis: 5
Operator (+ - * / % =): *
Ganze Zahl: 5
Zwischenergebnis: 25
Operator (+ - * / % =): %
Ganze Zahl: 3
Zwischenergebnis: 1
Operator (+ - * / % =): =
Endergebnis: 1
```

24WS-UE02-A06 TicTacToe Benutzereingabe

Lesen Sie den gesamten nachfolgenden Text inklusive der Bearbeitungshinweise bevor Sie mit Ihrer Lösung starten!

Benutzereingabe und Ausgabe des Spielfeldes

Schreiben sie ein Java-Programm, welches ein vereinfachtes TicTacToe-Spiel repräsentiert.

Es soll ausgegeben werden, welcher Spieler*in an der Reihe ist ("Player X" oder "Player O"). Anschließend sollen die Koordinaten der Spieler eingegeben werden. Diese geben an, wo der Kreis oder das Kreuz am Spielfeld platziert werden sollen.

Die erste Ziffer gibt die Reihe an und die zweite Ziffer die Spalte. Beispielsweise fürt die Konsoleneingabe `32` dazu, dass das Spielfeld in der dritten Reihe in der zweiten Spalte befüllt wird.

Weiterhin soll die Konsoleneingabe der Koordinaten einer Validierung unterzogen werden. Dabei sollen die folgenden Fälle überprüft werden:

- Die angegebenen Koordinaten dürfen nicht außerhalb des Spielfeldes liegen.
- Bereits belegte Koordinaten dürfen nicht erneut belegt werden.

Eine Abbruchbedingung durch einen Sieg ist nicht notwendig für diese Aufgabe. Das Spiel soll beendet werden, wenn das Spielfeld komplett ausgefüllt ist.

Ein typischer Spielverlauf könnte entsprechend wie folgt aussehen:

```
Player X: 32
+---+---+---+
|   |   |   |
+---+---+---+
|   |   |   |
+---+---+---+
|   | X |   |
+---+---+---+

Player O: 32
invalid move (32 is occupied)
Player O: 14
invalid move (14 is outside the board)
Player O: 31
+---+---+---+
|   |   |   |
+---+---+---+
|   |   |   |
+---+---+---+
| O | X |   |
+---+---+---+

...


Player X: 22
+---+---+---+
| X | O | O |
```

```
+---+---+---+
| O | X | X |
+---+---+---+
| O | X | X |
+---+---+---+
```

Bearbeitungshinweise

1. Starten Sie ähnlich wie in der letzten TicTacToe-Aufgabe aus UE01 mit dem Anlegen von den Variablen für die Spielfeldkoordinaten und der Ausgabe des Spielfeldes.
2. Verwenden Sie eine Schleife, die dann abbricht, wenn alle Spielfeldkoordinaten belegt sind. Lesen Sie bei jedem Durchlauf eine Koordinate ein und geben Sie das Spielfeld wiederum aus.
3. Sie benötigen eine Variable die angibt, wer in dieser Runde spielt. Legen Sie hierfür eine Variable an und verändern Sie den Wert nach jeder Runde.
4. Führen Sie nun die Inputvalidierung ein. Wechseln Sie jetzt nur dann zwischen Player X / Player O, wenn der Benutzer einen gültigen Wert (11, 12, 13, 21, 22, 23, 31, 32, 33) eingegeben hat und der angegebene Platz noch nicht belegt ist, andernfalls, geben Sie den Fehler aus und fragen Sie erneut nach dem Eingabewert.