

# Üben 2 – Verzweigungen, Schleifen

Einführung in die Programmierung 1  
Wintersemester 23



# Vorbereitung

- TUWEL öffnen, IntelliJ-Projekt „Ueben2.zip“ herunterladen
- Projekt in IntelliJ öffnen
- Laden Sie das bearbeitete Projekt am Ende wieder in TUWEL hoch

# CharacterCalc

```
public class CharacterCalc {  
    public static void main(String[] args) {  
        char a = '2', b = '5', c = '7';  
        int diffAB = -1, diffBC = -1;  
        if (a < b) {  
            if (b < c && a >= '0' && c <= '9') {  
                // hier gilt für a, b, c: ...  
                diffAB = b - a;  
                diffBC = c - b;  
            }  
        } else {  
            if (c <= b && c >= '0' && a <= '9') {  
                // hier gilt für a, b, c: ...  
                diffAB = a - b;  
                diffBC = b - c;  
            }  
        }  
        if (diffAB < 0) {  
            // hier gilt für diffBC: ...  
            System.out.println("Ungültige Werte");  
        } else if (diffAB > diffBC) {  
            System.out.println("Ja, " + diffAB + " größer " + diffBC);  
        } else {  
            System.out.println("Nein, " + diffAB + " nicht größer " + diffBC);  
        }  
    }  
}
```

1) Welche Werte haben diffAB und diffBC am Ende des Programms?

2) Welche Aussagen lassen sich über a, b und c an den markierten Stellen machen (Beziehung zueinander, mögliche Werte, ...), auch wenn ihre genauen Werte nicht bekannt sind?

3) Wann lautet die Ausgabe "Ungültige Werte"? Was passiert z.B., wenn man a und b vertauscht?

4) Warum ergibt

`int r='5'-'3';` das Ergebnis 2, aber  
`int r='5'+'3';` nicht das Ergebnis 8?  
Was ergibt  
`int r='5'+'3';` und was  
`char r='5'+'3';` ?

# CharacterCalc – Lösung

```
public class CharacterCalc {
    public static void main(String[] args) {
        char a = '2', b = '5', c = '7';
        int diffAB = -1, diffBC = -1;
        if (a < b) {
            if (b < c && a >= '0' && c <= '9') {
                // hier gilt: a < b < c und a, b, c sind Ziffern
                diffAB = b - a;
                diffBC = c - b;
            }
        } else {
            if (c <= b && c >= '0' && a <= '9') {
                // hier gilt: a >= b >= c und a, b, c sind Ziffern
                diffAB = a - b;
                diffBC = b - c;
            }
        }
        if (diffAB < 0) {
            // hier gilt: diffBC == -1
            System.out.println("Ungültige Werte");
        } else if (diffAB > diffBC) {
            System.out.println("Ja, " + diffAB + " größer " + diffBC);
        } else {
            System.out.println("Nein, " + diffAB + " nicht größer " + diffBC);
        }
    }
}
```

1) diffAB == 3 und diffBC == 2

2) Siehe Kommentare im Code

3) Werte müssen streng aufsteigend oder absteigend sein, um „gültig“ zu sein

4) '5' - '3' gibt an, nach wie vielen Codezeichen das Zeichen '5' auf das Zeichen '3' folgt.  
'5' + '3' addiert die Codewerte von '5' und '3', ergibt die Zahl 104, welche das Zeichen 'h' codiert.

# CharacterCalc – Mögliche Vereinfachung

```
public class CharacterCalcSimplified {  
  
    public static void main(String[] args) {  
        char a = '1', b = '3', c = '4';  
        if ((a < b && b < c && a >= '0' && c <= '9') || (a >= b && b >= c && a <= '9' && c >= '0')) {  
            int diffAB = Math.abs(a - b);  
            int diffBC = Math.abs(b - c);  
            if (diffAB > diffBC) {  
                System.out.println("Ja, " + diffAB + " größer " + diffBC);  
            } else {  
                System.out.println("Nein, " + diffAB + " nicht größer " + diffBC);  
            }  
        } else {  
            System.out.println("Ungültige Werte");  
        }  
    }  
}
```

# Lines

```
public class Lines {  
  
    public static void main(String[] args) {  
        System.out.println("*****");  
        System.out.println("*****");  
        System.out.println("*****");  
        System.out.println("*****");  
        System.out.println("*****");  
        System.out.println("*****");  
        System.out.println("*****");  
        System.out.println("*****");  
        System.out.println("*****");  
    }  
}
```

## 1. Aufgabe:

Ändern Sie den Code so, dass alle Linien mit Hilfe einer Schleife gezeichnet werden!

## 2. Aufgabe:

Ändern Sie den Code so, dass mit Hilfe der Änderung einer Variable eine beliebige Anzahl von Zeilen gezeichnet werden kann.

```
*****  
*****  
*****  
*****  
*****  
*****  
*****  
*****  
*****  
*****
```

2

```
*****  
*****
```

5

```
*****  
*****  
*****  
*****  
*****
```

9

```
*****  
*****  
*****  
*****  
*****  
*****  
*****  
*****  
*****  
*****
```

# Lines – 1. Erweiterung

- Erweitern Sie Ihr Programm wie folgt:
  - Die Zeilen sollen anwachsen und zwar jede Zeile ist um zwei '\*' länger als die davor

1  
\*\*

4  
\*\*  
\*\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*

8  
\*\*  
\*\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*

11  
\*\*  
\*\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*

# Lines – 2. Erweiterung

- Erweitern Sie Ihr Programm wie folgt:
  - Die Zeilen sollen abwechselnd aus '\*' und '-' bestehen

1

```
**
```

4

```
**  
---  
*****  
-----
```

8

```
**  
---  
*****  
-----  
*****  
-----  
*****  
-----
```

11

```
**  
---  
*****  
-----  
*****  
-----  
*****  
-----  
*****  
-----  
*****  
-----
```



# Lines – 3. Erweiterung (optional)

- Erweitern Sie Ihr Programm wie folgt:
  - Die Zeilen sollen bis zur Mitte anwachsen, danach wieder kürzer werden!
  - Hinweis: Bei geraden Zahlen sind in der Mitte 2 Linien gleich lang

1  
\*\*

4  
\*\*  
--  
\*\*\*\*  
--

8  
\*\*  
--  
\*\*\*\*\*  
-----  
\*\*\*\*\*  
-----  
\*\*\*\*  
--

11  
\*\*  
--  
\*\*\*\*\*  
-----  
\*\*\*\*\*  
-----  
\*\*\*\*\*  
-----  
\*\*\*\*\*  
-----  
\*\*\*\*\*  
--  
\*\*

# Lines – Mögliche Lösung

```
public class Lines {  
  
    public static void main(String[] args) {  
        for (int i = 0; i < 9; i++) {  
            System.out.println("*****");  
        }  
    }  
}
```

```
public class Lines {  
  
    public static void main(String[] args) {  
        int lines = 9;  
        for (int i = 0; i < lines; i++) {  
            System.out.println("*****");  
        }  
    }  
}
```

# Lines 1. Erweiterung – Mögliche Lösung

```
public class Lines {  
  
    public static void main(String[] args) {  
        String star = "***";  
        int lines = 9;  
        for (int i = 0; i < lines; i++) {  
            for (int j = 0; j <= i; j++) {  
                System.out.print(star);  
            }  
            System.out.println();  
        }  
    }  
}
```

# Lines 2. & 3. Erweiterung – Mögliche Lösung

```
// 2. Erweiterung
public class Lines {

    public static void main(String[] args) {
        String star = "***";
        String line = "--";
        String act;
        int lines = 9;
        for (int i = 0; i < lines; i++) {
            if (i % 2 == 0) {
                act = star;
            } else {
                act = line;
            }
            for (int j = 0; j <= i; j++) {
                System.out.print(act);
            }
            System.out.println();
        }
    }
}
```

```
// 3. Erweiterung
public class Lines {

    public static void main(String[] args) {
        int lines = 9;
        for (int i = 0; i < lines; i++) {
            String star = "***";
            String line = "--";
            String act;
            if (i % 2 == 0) {
                act = star;
            } else {
                act = line;
            }
            int max = i < lines / 2 ? i : lines - i - 1;
            for (int j = 0; j <= max; j++) {
                System.out.print(act);
            }
            System.out.println();
        }
    }
}
```

# Triangles

- Implementieren Sie das folgende Programm:
  - Deklarieren Sie eine Variable `lines` für ganze Zahlen und initialisieren Sie sie mit verschiedenen Werten zwischen 0 und 9
  - Geben Sie die entsprechende Anzahl an Zeilen aus
    - Die erste Zeile enthält eine `1`, die zweite Zeile zweimal die `2`, die dritte dreimal die `3`, usw.

```
lines = 0
```

```
lines = 1
```

```
1
```

```
lines = 2
```

```
1  
22
```

```
lines = 4
```

```
1  
22  
333  
4444
```

# Triangles – 1. Erweiterung

- Erweitern Sie Ihr Programm um folgende Funktionalitäten:
  - Zwischen zwei Zahlen soll jeweils das Zeichen '|' ausgegeben werden
  - Die Anzahl an Zahlen pro Zeile bleibt unverändert

```
lines = 0
```

```
lines = 1
```

```
1
```

```
lines = 2
```

```
1  
2|2
```

```
lines = 4
```

```
1  
2|2  
3|3|3  
4|4|4|4
```

# Triangles – 2. Erweiterung (optional)

- Erweitern Sie Ihr Programm so, dass zwischen zwei Zeilen mit Zahlen jeweils eine Zeile bestehend aus so vielen '-', wie die Zeile darüber Zeichen enthält, ausgegeben wird
  - Die Anzahl der Zeilen mit Zahlen bleibt unverändert
  - Das Programm soll für alle einstelligen Zahlen richtig funktionieren

```
lines = 4
```

```
1
-
2|2
---
3|3|3
-----
4|4|4|4
```

```
lines = 9
```

```
...
-----
7|7|7|7|7|7|7
-----
8|8|8|8|8|8|8
-----
9|9|9|9|9|9|9|9
```

# Triangles – Mögliche Lösung

```
public class Triangles {  
  
    public static void main(String[] args) {  
        int lines = 2;        // Werte zwischen 0 und 9 möglich  
        for (int i = 1; i <= lines; i++) {  
            for (int j = 0; j < i; j++) {  
                System.out.print(i);  
            }  
            System.out.println();  
        }  
    }  
}
```



# Triangles 1. Erweiterung – Mögliche Lösung

```
public class Triangles {  
  
    public static void main(String[] args) {  
        int lines = 5;        // Werte zwischen 0 und 9 möglich  
        for (int i = 1; i <= lines; i++) {  
            for (int j = 0; j < i; j++) {  
                System.out.print(i);  
                if (j != i - 1) {  
                    System.out.print('|');  
                }  
            }  
            System.out.println();  
        }  
    }  
}
```

# Triangles 2. Erweiterung – Mögliche Lösung

```
public class Triangles {  
  
    public static void main(String[] args) {  
        int lines = 9;        // Werte zwischen 0 und 9 möglich  
        for (int i = 1; i <= lines; i++) {  
            for (int j = 0; j < i; j++) {  
                System.out.print(i);  
                if (j != i - 1) System.out.print('|');  
            }  
            System.out.println();  
            if (i != lines) {  
                for (int j = 0; j < 2 * i - 1; j++) {  
                    System.out.print('-');  
                }  
                System.out.println();  
            }  
        }  
    }  
}
```

# ConsolePattern

- Implementieren Sie das folgende Programm:
  - Deklarieren und initialisieren Sie eine Variable n für die Größe des Quadrats
  - Geben Sie n Zeilen aus, wobei jede Zeile genau n Punkte '.' enthält
  - Der Lerneffekt ist größer, wenn Sie das Programm selber schreiben und nicht aus den Vorlesungsfolien kopieren

Größe des Quadrats:

1

.

Größe des Quadrats:

2

..

..

Größe des Quadrats:

5

.....

.....

.....

.....

.....

# ConsolePattern – 1. Erweiterung

- Erweitern Sie das Programm wie folgt:
  - Geben Sie in den Ecken das Zeichen '+' aus
  - Geben Sie in der ersten und letzten Zeile das Zeichen '-' aus
  - Geben Sie in der ersten und letzten Spalte das Zeichen '|' aus
  - Geben Sie an den anderen Stellen des Quadrats das Zeichen '.' aus

Größe des Quadrats:

1

+

Größe des Quadrats:

2

++

++

Größe des Quadrats:

5

+---+

|...|

|...|

|...|

+---+

# ConsolePattern – 2. Erweiterung (optional)

- Erweitern Sie das Programm wie folgt:
  - Geben Sie in der Hauptdiagonale das Zeichen `'\'` aus
  - Geben Sie in der Gegendiagonale das Zeichen `'/'` aus
  - Geben Sie das Zeichen `'x'` dort aus, wo sich die Diagonalen treffen
- Nehmen Sie an, dass die Größe des Quadrats ungerade ist

Größe des Quadrats:

3

```
+--+  
|x|  
+--+
```

Größe des Quadrats:

5

```
+---+  
|\./|  
|.x.|  
|/.\|  
+---+
```

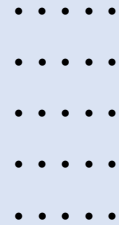
Größe des Quadrats:

7

```
+-----+  
|\.../|  
|.\/.|  
|..x..|  
|./.\.|  
|/...\|  
+-----+
```

# ConsolePattern – Mögliche Lösung

```
public class ConsolePattern {  
  
    public static void main(String[] args) {  
        int n = 5;  
        for (int i = 0; i < n; i++) {  
            for (int j = 0; j < n; j++) {  
                System.out.print('.');  
            }  
            System.out.println();  
        }  
    }  
}
```



.....  
.....  
.....  
.....  
.....

# ConsolePattern 1. Erweiterung – Mögliche Lösung

```
public class ConsolePattern {  
  
    public static void main(String[] args) {  
        int n = 5;  
        for (int i = 0; i < n; i++) {  
            for (int j = 0; j < n; j++) {  
                if ((i == 0 || i == n - 1) && (j == 0 || j == n - 1)) System.out.print('+');  
                else if (i == 0 || i == n - 1) System.out.print('-');  
                else if (j == 0 || j == n - 1) System.out.print('|');  
                else System.out.print(' ');  
            }  
            System.out.println();  
        }  
    }  
}
```

```
+----+  
|...|  
|...|  
|...|  
+----+
```

# ConsolePattern 2. Erweiterung – Mögliche Lösung

```
public class ConsolePattern {  
  
    public static void main(String[] args) {  
        int n = 5;  
        for (int i = 0; i < n; i++) {  
            for (int j = 0; j < n; j++) {  
                if ((i == 0 || i == n - 1) && (j == 0 || j == n - 1)) System.out.print('+');  
                else if (i == 0 || i == n - 1) System.out.print('-');  
                else if (j == 0 || j == n - 1) System.out.print('|');  
                else if (i == n / 2 && j == n / 2) System.out.print('x');  
                else if (j == i) System.out.print('\\');  
                else if (i + j == n - 1) System.out.print('/');  
                else System.out.print(' ');  
            }  
            System.out.println();  
        }  
    }  
}
```

```
+----+  
|\.\/|  
|.x.|  
|\/.\|  
+----+
```