Hochschule -

Fakultät IV – Technische Informatik
Modul: Programmieren 1
Professor: -

Entwicklungsarbeit

von

Sebastian Schramm Matrikel-Nr. -

12. März 2021

Inhaltsverzeichnis

1 Kapitel 1

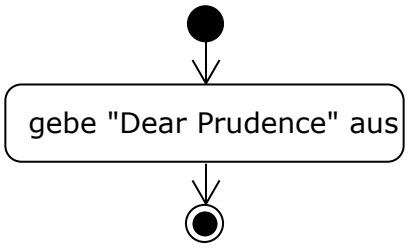
1.1 Aufgabenstellung

Um uns mit Java und der Konsole vertraut zu machen sollten wir zuerst ein Java Programm in der Konsole schreiben, welches ”Dear Prudence” ausgibt. Anschließend mit Javac Kompilieren und mit Java ausführen. Wenn wir dies geschafft haben, sollte wir als nächstes in NetBeans ein neues Projekt erstellen und dort das selbe Programm erneut Programmieren.

1.2 Anforderungsdefinition

1. Das Programm soll ”Dear Prudence” auf der Konsole ausgeben.

1.3 Entwurf



1.4 Quellcode

1.4.1 Main.java

```
1  package chapter_01;
2
3  /**
4   * Klasse mit der Main-Methode
5   * @author sebastian
6   *
7   */
8  public class Main {
9
10   /**
11    * Die Main Methode
12    * Gibt "Dear Prudence" aus
13    * @param args
14    */
15   public static void main(String[] args) {
16     System.out.println("Dear Prudence");
17   }
18 }
```

1.5 Testdokumentation

Wenn das Programm gestartet wird, sollte ”Dear Prudence” auf der Konsole ausgegeben werde. Dies war der fall.

1.6 Benutzungshinweise

Navigieren Sie in der Kommandozeile zum dem Ordner, wo sich die Java Datei befindet. Danach führen sie ”javac Main.java” auf. Jetzt können Sie das Programm mit ”java main” starten. In der Konsole sollte nun ”Dear Prudence” angezeigt werden.

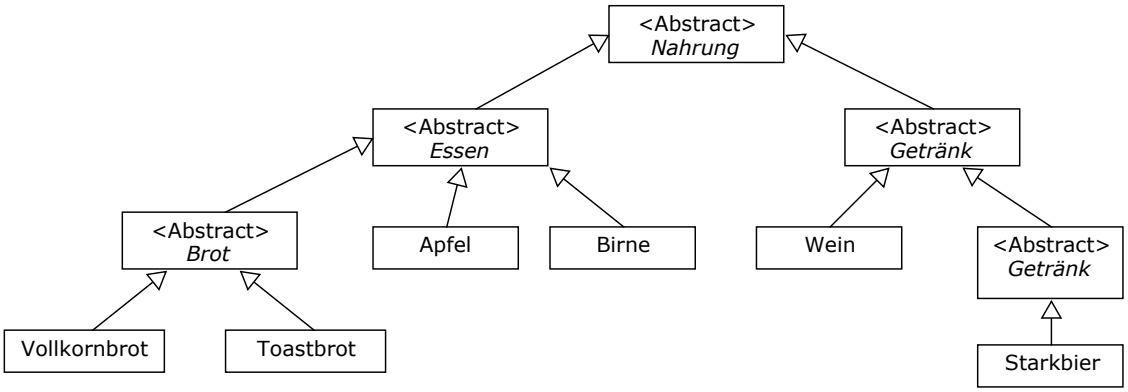
1.7 Anwendungsbeispiel

Nach dem Aufruf von java Main, sollten wir folgendes sehen:

```
[sebastian@laptop bin]$ java Main
Dear Prudence
[sebastian@laptop bin]$
```

2 Kapitel 2

2.1 Entwurf



3 Kapitel 3

3.1 Teilaufgabe 1

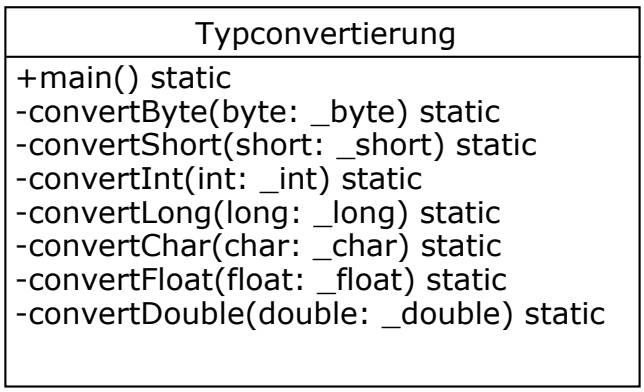
3.1.1 Aufgabenstellung

In der ersten Teilaufgabe sollten wir uns mit der Typkonvertierung befassen. Dafür schreiben wir ein kleines Programm, welches die primitiven Datentypen erweiternd und einschränkend Konvertiert.

3.1.2 Anforderungsdefinition

1. Zu jedem Primitiven Datentypen eine erweiternde und einschränkende Konvertierung durchführen.

3.1.3 Entwurf



3.1.4 Quelltext

3.1.4.1 Typkonvertierungen.java

```java
package chapter_03;

/**
 * Klasse mit der Main-Methode
 * und der einzelnen Typkonvertierungen
 * @author Sebastian
 */
public class Typkonvertierungen {

  public static void main(String[] args) {
    /*
     * Rund die einzelnen Methoden auf, mit entsprechenden Werten
     */
    convertByte((byte) -128);
    convertShort((short) 34);
    convertInt(98987);
    convertLong(987987987);

    convertChar('a');

    convertFloat(15.0f);
    convertDouble(1.79769313486231570E308);
  }

  /**
   * Eine erweiternde Konvertierung von Byte zu Double
   * @param _byte
   */
  private static void convertByte(byte _byte) {
    short newShort = _byte;
    int newInt = _byte;
    long newLong = _byte;
    float newFloat = _byte;
    double newDouble = _byte;

    System.out.println("---------------------");
    System.out.println("Byte erweiternd");
    System.out.println("Byte   " + _byte);
    System.out.println("Short  " + newShort);
    System.out.println("Int    " + newInt);
    System.out.println("Long   " + newLong);
    System.out.println("Float  " + newFloat);
    System.out.println("Double " + newDouble);
    System.out.println("\nChar   " + (char) newInt); //Char wird hier separat
        ausgegeben
    System.out.println("---------------------");
  }

  /**
   * Eine einschränkende Konvertierung von Short zu Byte
   * Eine erweiternde Konvertierung von Short zu Double
   * @param _short
   */
  private static void convertShort(short _short) {
    byte newByte = (byte) _short;
    int newInt = _short;
    long newLong = _short;
    float newFloat = _short;
    double newDouble = _short;

```

```
60      System.out.println("Short einschränkend");
61      System.out.println("Short  " + _short);
62      System.out.println("Byte   " + newByte);
63
64      System.out.println("Short erweiternd");
65      System.out.println("Short  " + _short);
66      System.out.println("Int    " + newInt);
67      System.out.println("Long   " + newLong);
68      System.out.println("Float  " + newFloat);
69      System.out.println("Double " + newDouble);
70      System.out.println("\nChar   " + (char) newInt); //Char wird hier separat
            ausgegeben
71      System.out.println("----------------------");
72    }
73
74    /**
75     * Eine einschränkende Konvertierung von Int zu Byte
76     * Eine erweiternde Konvertierung von Int zu Double
77     * @param _int
78     */
79    private static void convertInt(int _int) {
80      short newShort = (short) _int;
81      byte newByte = (byte) _int ;
82
83      long newLong = _int;
84      float newFloat = _int;
85      double newDouble = _int;
86
87      System.out.println("Int einschränkend");
88      System.out.println("Int    " + _int);
89      System.out.println("Short  " + newShort);
90      System.out.println("Byte   " + newByte);
91
92      System.out.println("Int erweiternd");
93      System.out.println("Int    " + _int);
94      System.out.println("Long   " + newLong);
95      System.out.println("Float  " + newFloat);
96      System.out.println("Double " + newDouble);
97      System.out.println("\nChar   " + (char) _int); //Char wird hier separat
            ausgegeben
98      System.out.println("----------------------");
99    }
100
101   /**
102    * Eine einschränkende Konvertierung von Long zu Byte
103    * Eine erweiternde Konvertierung von Long zu Double
104    * @param _long
105    */
106   private static void convertLong(long _long) {
107     int newInt = (int) _long;
108     short newShort = (short) _long;
109     byte newByte = (byte) _long;
110
111     float newFloat = _long;
112     double newDouble = _long;
113
114     System.out.println("Long einschränkend");
115     System.out.println("Long   " + _long);
116     System.out.println("Int    " + newInt);
117     System.out.println("Short  " + newShort);
118     System.out.println("Byte   " + newByte);
119
120     System.out.println("Long erweiternd");
```

```
121     System.out.println("Long   " + _long);
122     System.out.println("Float  " + newFloat);
123     System.out.println("Double " + newDouble);
124     System.out.println("\nChar   " + (char) newInt); //Char wird hier separat
            ausgegeben
125     System.out.println("----------------------");
126   }
127
128   /**
129    * Eine einschränkende Konvertierung von Char zu Byte
130    * Eine erweiternde Konvertierung von Char zu Double
131    * @param _char
132    */
133   private static void convertChar(char _char) {
134     int newInt = _char;
135     short newShort = (short) _char;
136     byte newByte = (byte) _char;
137
138     long newLong = _char;
139     float newFloat = _char;
140     double newDouble = _char;
141
142     System.out.println("Char einschränkend");
143     System.out.println("Char   " + _char);
144     System.out.println("Long   " + newLong);
145     System.out.println("Int    " + newInt);
146     System.out.println("Short  " + newShort);
147     System.out.println("Byte   " + newByte);
148
149     System.out.println("Char erweiternd");
150     System.out.println("Char   " + _char);
151     System.out.println("Long   " + newLong);
152     System.out.println("Float  " + newFloat);
153     System.out.println("Double " + newDouble);
154     System.out.println("----------------------");
155   }
156
157   /**
158    * Eine einschränkende Konvertierung von Float zu Byte
159    * Eine erweiternde Konvertierung von Float zu Double
160    * @param _float
161    */
162   private static void convertFloat(float _float) {
163     long newLong = (long) _float;
164     int newInt = (int) _float;
165     short newShort = (short) _float;
166     byte newByte = (byte) _float;
167
168     double newDouble = _float;
169
170     System.out.println("Float einschränkend");
171     System.out.println("Float  " + _float);
172     System.out.println("Long   " + newLong);
173     System.out.println("Int    " + newInt);
174     System.out.println("Short  " + newShort);
175     System.out.println("Byte   " + newByte);
176
177     System.out.println("Float erweiternd");
178     System.out.println("Float  " + _float);
179     System.out.println("Double " + newDouble);
180     System.out.println("\nChar   " + (char) newInt); //Char wird hier separat
            ausgegeben
181     System.out.println("----------------------");
```

```
182     }
183
184     /**
185      * Eine einschränkende Konvertierung von Double zu Byte
186      * @param _double
187      */
188     private static void convertDouble(double _double) {
189       float newFloat = (float) _double;
190       long newLong = (long) _double;
191       int newInt = (int) _double;
192       short newShort = (short) _double;
193       byte newByte = (byte) _double;
194
195       System.out.println("Double einschränkend");
196       System.out.println("Double " + _double);
197       System.out.println("Float  " + newFloat);
198       System.out.println("Long   " + newLong);
199       System.out.println("Int    " + newInt);
200       System.out.println("Short  " + newShort);
201       System.out.println("Byte   " + newByte);
202       System.out.println("\nChar   " + (char) newInt); //Char wird hier separat
              ausgegeben
203       System.out.println("---------------------");
204     }
205
206 }
```

3.1.5 Testdokumentation

Nach den Aufruf des Programms, sollten alle Typkonvertierungen auf der Konsole ausgegeben werden. Dies ist auch geschehen.

3.1.6 Benutzungshinweise

Keine Besonderen Benutzungshinweise. Man navigiere zu dem Ordner von sich die Compilierte Datei mit dem Namen ”Typkonvertierungen.class” befindet und führt anschließend java Typkonvertierungen aus.

3.1.7 Anwendungsbeispiel

Nach dem man das Programm gestartet hat (aufgrund der Formatierung, werden einige Zeichen bei Char nicht dargestellt), sollte folgende Ausgabe erscheinen:

```
1  [sebastian@laptop bin]$ java Typkonvertierungen
2  ---------------------
3  Byte erweiternd
4  Byte   -128
5  Short  -128
6  Int    -128
7  Long   -128
8  Float  -128.0
9  Double -128.0
10
11 Char
12 ---------------------
13 Short einschränkend
14 Short  34
15 Byte   34
16 Short erweiternd
17 Short  34
18 Int    34
19 Long   34
20 Float  34.0
```

```
Double 34.0

Char   "
---------------------
Int einschränkend
Int    98987
Short  -32085
Byte   -85
Int erweiternd
Int    98987
Long   98987
Float  98987.0
Double 98987.0

Char
---------------------
Long einschränkend
Long   987987987
Int    987987987
Short  -32749
Byte   19
Long erweiternd
Long   987987987
Float  9.8798797E8
Double 9.87987987E8

Char
---------------------
Char einschränkend
Char   a
Long   97
Int    97
Short  97
Byte   97
Char erweiternd
Char   a
Long   97
Float  97.0
Double 97.0
---------------------
Float einschränkend
Float  15.0
Long   15
Int    15
Short  15
Byte   15
Float erweiternd
Float  15.0
Double 15.0

Char
---------------------
Double einschränkend
Double 1.7976931348623157E308
Float  Infinity
Long   9223372036854775807
Int    2147483647
Short  -1
Byte   -1

Char
---------------------
[sebastian@laptop bin]$
```

3.2 Teilaufgabe 2

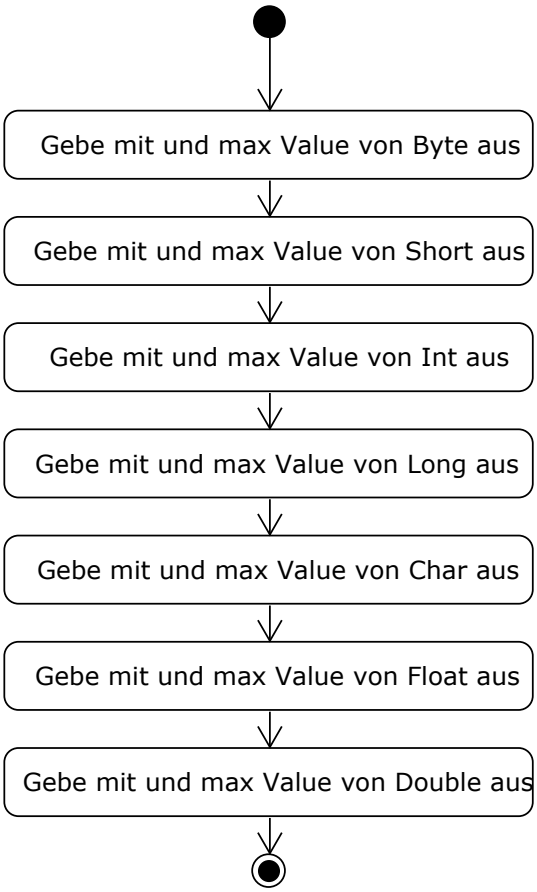
3.2.1 Aufgabenstellung

In der Zweiten Teilaufgabe sollen wir uns mit den Wertebereichen der Primitiven Datentypen befassen und deren Wertebereiche in einem Programm ausgeben.

3.2.2 Anforderungsdefinition

1. Zu jedem primitiven Datentypen den Max und Min-Wert ausgeben.

3.2.3 Entwurf



3.2.4 Quelltext

3.2.4.1 Wertebereiche.java

```
1  package chapter_03;
2
3  /**
4   * Klasse mit der Main-Methode
5   * und gibt die Wertebereiche der primitiven Datentypen aus
6   * @author Sebastian
7   */
8  public class Wertebereiche {
9
10   public static void main(String[] args) {
11     //Min und Max Value von Byte
12     System.out.println("Byte min " + Byte.MIN_VALUE + " | Byte max " + Byte.
           MAX_VALUE);
```

```
13      //Min und Max Value von Short
14      System.out.println("Short min " + Short.MIN_VALUE + " | Short max " + Short.
            MAX_VALUE);
15      //Min und Max Value von Integer
16      System.out.println("Integer min " + Integer.MIN_VALUE + " | Integer max " +
            Integer.MAX_VALUE);
17      //Min und Max Value von Long
18      System.out.println("Long min " + Long.MIN_VALUE + " | Byte Long " + Long.
            MAX_VALUE);
19
20      //Min und Max Value von Char
21      System.out.println("Char min \u0000 | Char max \uffff");
22
23      //Min und Max Value von Float
24      System.out.println("Float min " + Float.MIN_VALUE + " | Float max " + Float.
            MAX_VALUE);
25      //Min und Max Value von Double
26      System.out.println("Double min " + Double.MIN_VALUE + " | Double max " + Double
            .MAX_VALUE);
27    }
28
29  }
```

3.2.5 Testdokumentation

Nach dem Start des Programms sollten die Min und Max werte der einzelnen Datentypen ausgegeben werden. Anschließend wurden die Werten mit den in der Java Dokumentation stehenden Werten verglichen. Alles hat soweit gestimmt.

3.2.6 Benutzungshinweise

Keine Besonderen Benutzungshinweise. Man navigiere zu dem Ordner von sich die Compilierte Datei mit dem Namen ”Wertebereiche.class” befindet und führt anschließend java Wertebereiche aus.

3.2.7 Anwendungsbeispiel

Nach dem man das Programm gestartet hat (aufgrund der Formatierung, werden einige Zeichen bei Char nicht dargestellt), sollte folgende Ausgabe erscheinen:

```
30  [sebastian@laptop bin]$ java Wertebereiche
31  Byte min -128 | Byte max 127
32  Short min -32768 | Short max 32767
33  Integer min -2147483648 | Integer max 2147483647
34  Long min -9223372036854775808 | Byte Long 9223372036854775807
35  Char min  | Char max
36  Float min 1.4E-45 | Float max 3.4028235E38
37  Double min 4.9E-324 | Double max 1.7976931348623157E308
38  [sebastian@laptop bin]$
```

4 Kapitel 4

4.1 Teilaufgabe 1

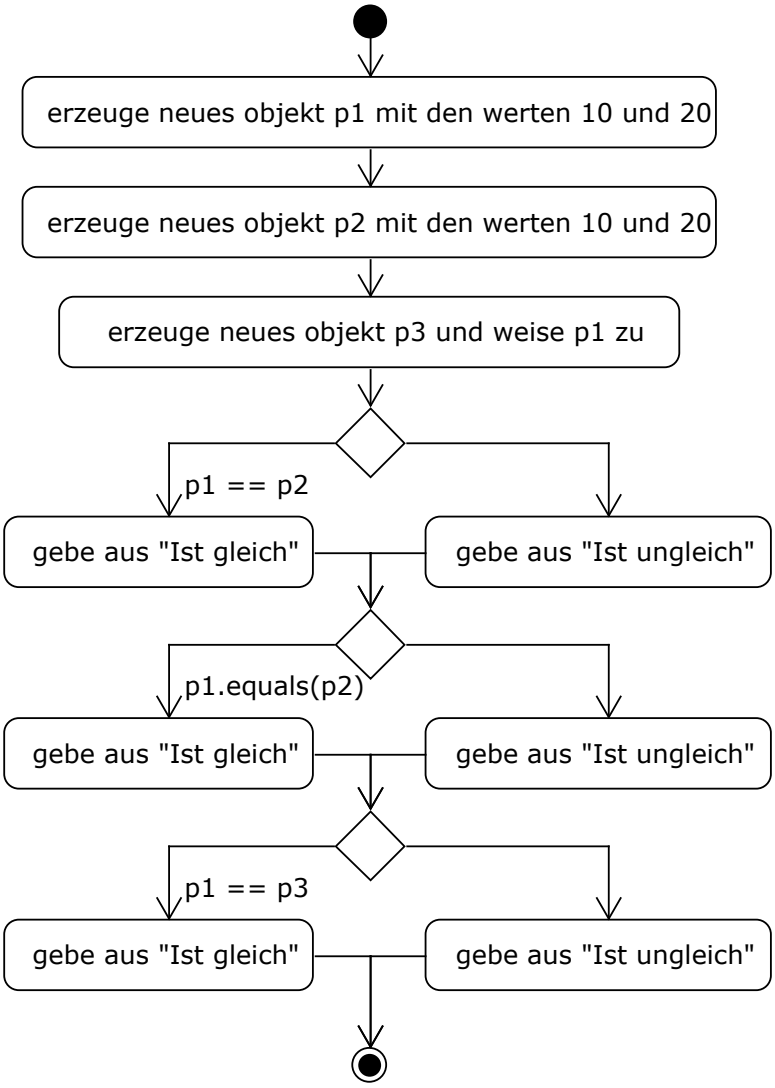
4.1.1 Aufgabenstellung

Wir sollen ein Programm schreiben welches Prüft ob zwei Referenzen gleich sind.

4.1.2 Anforderungsdefinition

1. Prüfe ob zwei Referenzen gleich sind.

4.1.3 Entwurf



4.1.4 Quellcode

4.1.4.1 Referenzen.java

```
1  package chapter_04;
2
3  /**
4   * Klasse mit der Main-Methode
5   * und prüft ob Zwei Referenzen gleich sind
6   * @author Sebastian
7   *
8   */
9  public class Referenzen {
10
11    public static void main(String[] args) {
12      /*
13       * Es werden zwei identische Objekte erzeugt
14       * mit den selben Werten.
15       * Zuletzt wird noch ein drittes erzeugt mit einer
16       * Referenz auf das erste
17       */
18      Punkt p1 = new Punkt(10, 20);
```

```
19      Punkt p2 = new Punkt(10, 20);
20      Punkt p3 = p1;
21
22      //Hier wird geprüft ob p1 und p2 die selbe Adresse hat.
23      if (p1 == p2)
24        System.out.println("Ist gleich");
25      else
26        System.out.println("Ist ungleich");
27
28      //Hier wird geprüft ob der Inhalt der selbe ist
29      if (p1.equals(p2))
30        System.out.println("Ist gleich");
31      else
32        System.out.println("Ist ungleich");
33
34      //Hier wird geprüft ob p3 und p1 gleich sind
35      if (p3 == p1)
36        System.out.println("Ist gleich");
37      else
38        System.out.println("Ist ungleich");
39    }
40
41  }
```

4.1.4.2 Punkt.java

```
1  package chapter_04;
2
3  /**
4   * Punkt Klasse
5   * Hier werden nur Zwei Punkte gespeichert
6   * @author Sebastian
7   *
8   */
9  @SuppressWarnings("unused")
10 public class Punkt {
11   private static int x = 0;
12   private static int y = 0;
13
14   public Punkt(int x, int y) {
15     this.x = x;
16     this.y = y;
17   }
18 }
```

4.1.5 Testdokumentation

Nach dem Start des Programms sollten die ersten beiden Bedingungen falsch sein und die dritte wahr, dies war auch der Fall.

4.1.6 Benutzungshinweise

Navigieren Sie in der Kommandozeile zum dem Ordner, wo sich die Java Datei befindet. Danach führen sie ”javac Referenzen.java” auf. Jetzt können Sie das Programm mit ”java Referenzen” starten.

4.1.7 Anwendungsbeispiel

Nach dem Aufruf von java Referenzen, sollten wir nun folgendes sehen:

```
1  [sebastian@laptop bin]$ java Referenzen
2  Ist ungleich
3  Ist ungleich
```

```
4 Ist gleich
5 [sebastian@laptop bin]$
```

4.2 Teilaufgabe 2

4.2.1 Aufgabenstellung

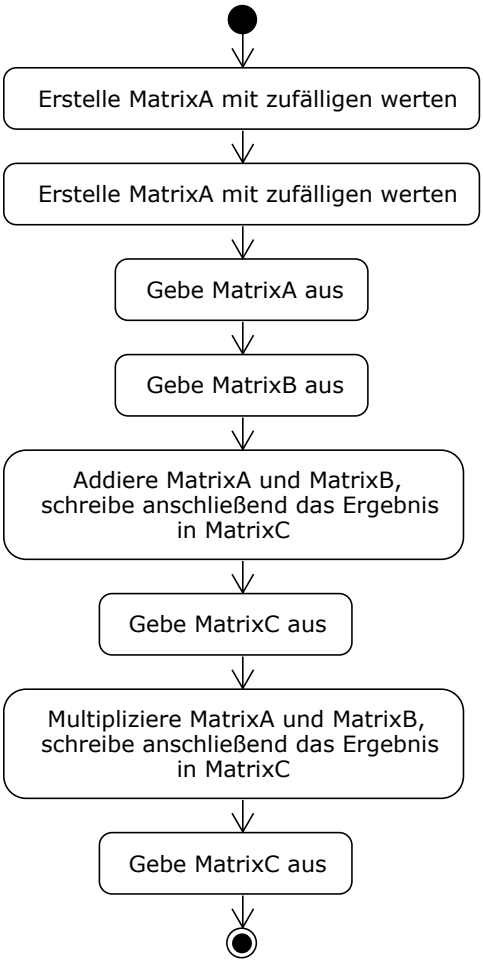
Es soll ein Programm erstellt werden welches zwei 2x2 Matrizen miteinander multiplizieren und addieren kann. Wobei die Elemente der beiden Matrizen bei der Erzeugung zufällig initialisiert werden sollen.

4.2.2 Anforderungsdefinition

1. Addiere zwei 2x2 Matrizen.
2. Multipliziere zwei 2x2 Matrizen.
3. Initialisiere die Matrizen mit zufälligen zahlen.

4.2.3 Entwurf

Es sollen zunächst 2 2x2 Matrizen erzeugt welche anschließend einer Methode übergeben werden, welche die beiden Matrizen mit zufälligen Werten initialisiert. Danach sollen beide Matrizen einmal ausgegeben werden. Nach der Ausgabe werden zwei Methoden aufgerufen, eine für die Multiplikation und eine für die Addition, diese liefern jeweils eine neue Matrix zurück mit den ausgerechneten Werten. Diese Matrix wird als MatrixC Temporär gespeichert und anschließend ausgegeben.



4.2.4 Quellcode

4.2.4.1 Matrizen.java

```java
package chapter_04;

import java.util.Random;

/**
 * Klasse mit der Main-Methode
 * Addiert und Multipliziert Matrizen
 * @author Sebastian
 *
 */
public class Matrizen {

  public static void main(String[] args) {
    int matrixA[][];
    int matrixB[][];

    /*
     * Initialisierungsmethode wird mit dem Wert n aufgerufen.
     * AnschlieSSend wird diese Matrix erzeugt und mit
     * zufällig generierten Zahlen befüllt.
     */
    matrixA = initialize(2);
    matrixB = initialize(2);

    /*
     * Zuerst werden die Beiden Matrizen A und B jeweils ausgegeben
     */
    System.out.println("Matrix A:");
    printMatrix(matrixA);
    System.out.println("Matrix B:");
    printMatrix(matrixB);
    /*
     * AnschlieSSend werden die Matrizen hier Addiert
     */
    System.out.println("Addition von A und B:");
    printMatrix(addition(matrixA, matrixB));
    /*
     * Und hier Multipliziert
     */
    System.out.println("Multiplikation von A und B:");
    printMatrix(multiplikation(matrixA, matrixB));
  }

  /**
   * Initialisierung des Arrays
   * @param n Die grÖSSe der nxn Matrix
   * @return matrix
   */
  private static int[][] initialize(int n) {
    int matrix[][] = new int[n][n];
    /*
     * Bei der Initialisierung wird einmal durch das gesamt Array dutch iteriert.
     * Dabei werden dann mit Math.random() zufällige Zahlen rein geschrieben.
     */

    Random rand = new Random();
    for (int i = 0; i < matrix.length; ++i)
      for (int l = 0; l < matrix[i].length; ++l)
        matrix[i][l] = rand.nextInt(200) - 100;

```

```
61      return matrix;
62    }
63
64    /**
65     * Addition der beiden Matrizen A und B
66     * @param matrixA
67     * @param matrixB
68     * @return Gibt ein neues Array mit den Addierten Werten zurück
69     */
70    private static int[][] addition(int matrixA[][], int matrixB[][]) {
71      int matrixAd[][] = new int[matrixA.length][matrixA[0].length]; //Es wird ein
            neues Temporäres Array angelegt
72
73      for (int i = 0; i < matrixA.length; ++i) {
74        for (int n = 0; n < matrixA[i].length; ++n) {
75          matrixAd[i][n] = matrixA[i][n] + matrixB[i][n];
76        }
77      }
78
79      return matrixAd;
80    }
81
82    /**
83     * Multiplikation der beiden Matrizen A und B
84     * @param matrixA
85     * @param matrixB
86     * @return Gibt ein neues Array mit den Multiplizierten Werten zurück
87     */
88    private static int[][] multiplikation(int matrixA[][], int matrixB[][]) {
89      int matrixMult[][] = new int[matrixB.length][matrixB[0].length];
90
91      for (int HmatrixB = 0; HmatrixB < matrixB.length; ++HmatrixB)
92        for (int WmatrixB = 0; WmatrixB < matrixB[HmatrixB].length; ++WmatrixB)
93          for (int WmatrixA = 0; WmatrixA < matrixB.length; ++WmatrixA)
94            matrixMult[HmatrixB][WmatrixB] += matrixA[HmatrixB][WmatrixA] * matrixB[
                  WmatrixA][WmatrixB];
95
96      return matrixMult;
97    }
98
99    /**
100    * Hier wird die Matrix ausgegeben
101    * @param matrix
102    */
103   private static void printMatrix(int matrix[][]) {
104     for (int y[]: matrix) {
105       for (int x: y)
106         System.out.print(x + "\t");
107       System.out.println();
108     }
109     System.out.println();
110   }
111
112 }
```

4.2.5 Testdokumentation

Das Programm hat nach dem Aufruf Zwei 2x2 Matrizen erstellt und initialisiert, anschließend miteinander Addiert und Multipliziert. Dabei wurde das Ergebnis mit einem Matrizen Rechner im Internet verglichen und die Ergebnisse haben übereingestimmt.

4.2.6 Benutzungshinweise

Navigieren Sie in der Kommandozeile zum dem Ordner, wo sich die Java Datei befindet. Danach führen sie ”javac Matrizen.java” auf. Jetzt können Sie das Programm mit ”java Matrizen” starten. Nach dem das Programm gestartet ist, können Sie die größe der Matrix angeben.

4.2.7 Anwendungsbeispiel

Nach dem Aufruf von java Matrizen, sollten wir nun folgendes sehen:

```
[sebastian@laptop bin]$ java Matrizen
Matrix A:
70  50
16  52

Matrix B:
80  75
11  33

Addition von A und B:
150 125
27  85

Multiplikation von A und B:
6150  6900
1852  2916
[sebastian@laptop bin]$
```

5 Kapitel 5

5.1 Teilaufgabe 1

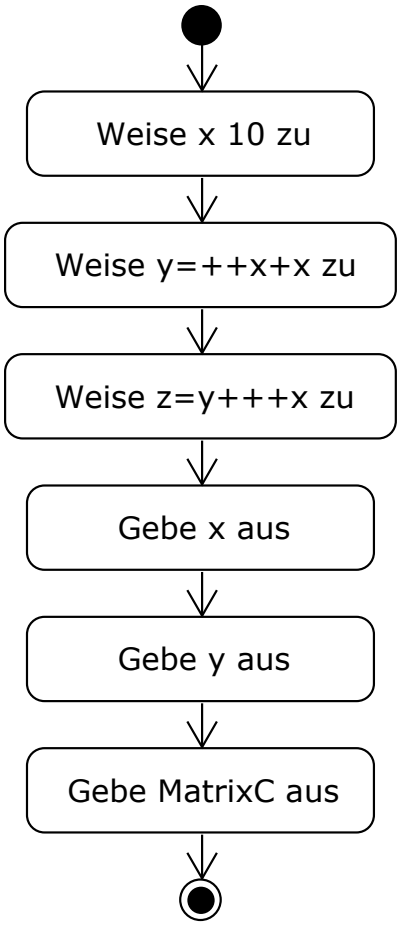
5.1.1 Aufgabenstellung

In der ersten Teilaufgabe sollen wir ein Kleines simples Programm schreiben, welches die Nebeneffekte in Java verdeutlicht.

5.1.2 Anforderungsdefinition

1. Nebeneffekte verdeutlichen.

5.1.3 Entwurf



5.1.4 Quelltext

5.1.4.1 Nebeneffekte.java

```java
package chapter_05;

/**
 * Klasse mit der Main-Methode
 * @author Sebastian
 *
 */
public class Nebeneffekte {

  public static void main(String[] args) {
    int x = 10;
    int y = ++x+x;
    int z = y+++--x;
    System.out.println("Der Wert von x lautet: " + x);
    System.out.println("Der Wert von y lautet: " + y);
    System.out.println("Der Wert von z lautet: " + z);
  }

}
```

5.1.5 Testdokumentation

Nach dem Start sollte x 10, y 23 und z 32 betragen, dies war auch der Fall.

5.1.6 Benutzungshinweise

Keine Besonderen Benutzungshinweise. Das Programm muss lediglich nur ausgeführt werden.

5.1.7 Anwendungsbeispiel

Nach dem man das Programm gestartet hat, sollte folgende Ausgabe erscheinen:

```
[sebastian@laptop bin]$ java Nebeneffekte
Der Wert von x lautet: 10
Der Wert von y lautet: 23
Der Wert von z lautet: 32
[sebastian@laptop bin]$
```

5.2 Teilaufgabe 2

5.2.1 Aufgabenstellung

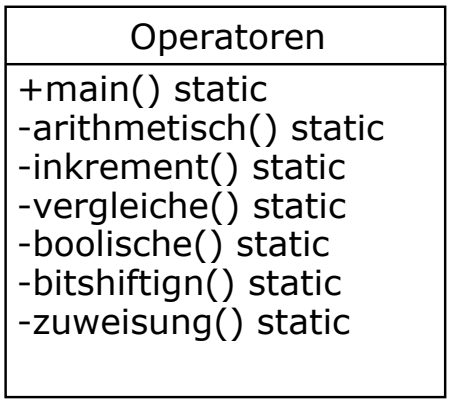
In der zweiten Teilaufgabe sollten wir ein Programm schreiben welches sämtliche Operatoren, die Java beinhaltet veranschaulichen.

5.2.2 Anforderungsdefinition

1. Verwende alle Operatoren in Java.

5.2.3 Entwurf

Es werden jeweils einzelne Methoden erstellt in denen die entsprechenden Operatoren ausgeführt werden.



5.2.4 Quelltext

5.2.4.1 Operatoren.java

```
package chapter_05;

@SuppressWarnings("unused")
public class Operatoren {
  //Schreiben Sie ein Programm, welches alle Operatoren in Java verwendet.
  /**
   * Klasse mit der Main-Methode
   * Dieses Programm solle alle Operatoren,
   * die in Java existieren verdeutlichen
```

```
10      * @param args
11      */
12     public static void main(String[] args) {
13       arithmetisch();
14       inkrement();
15       vergleiche();
16       boolesche();
17       bitshifting();
18       zuweisung();
19     }
20
21     private static void arithmetisch() {
22       System.out.println("Arithmetische Operatoren:");
23       System.out.println("23 + 34 = " + (23 + 34)); // Addition
24       System.out.println("54 - 32 = " + (54 - 32)); // Subtraktion
25       System.out.println("12 * 30 = " + 12 * 30);   // Multiplikation
26       System.out.println("56 / 12 = " + 56 / 12);   // Division
27       System.out.println("74 % 2  = " + 74 % 2);    // Teiler rest, Modulo-Operation,
             errechnet den Rest einer Division
28       int i;
29       System.out.println("int i = +3 = " + (i = +3)); // positives Vorzeichen
30       int n;
31       System.out.println("int n = -i = " + (n = -i)); //negatives Vorzeichen
32     }
33
34     private static void inkrement() {
35       int x = 10;
36       System.out.println("\nInkrement Operatoren:");
37       System.out.println("x   = " + x);
38       System.out.println("x++ = " + x++); //Postinkrement: Weist zuerst zu, dann
             hochzählen
39       System.out.println("x   = " + x);
40       System.out.println("++x = " + ++x); //Preinkrement: Zählt erst hoch, dann
             zuweisen
41       System.out.println("x   = " + x);
42       System.out.println("x-- = " + x--); //Postinkrement: Weist zuerst zu, dann
             hochzählen
43       System.out.println("x   = " + x);
44       System.out.println("--x = " + --x); //Preinkrement: Zählt erst hoch, dann
             zuweisen
45       System.out.println("x   = " + x);
46     }
47
48     private static void vergleiche() {
49       System.out.println("\nVergleichs Operatoren:");
50       System.out.println("37 == 2 = " + (37 == 2)); // gleich
51       System.out.println("1 != 2 = " + (1 != 2));   // ungleich
52       System.out.println("13 > 3 = " + (13 > 3));   // gröSSer
53       System.out.println("23 < 2 = " + (23 < 2));   // kleiner
54       System.out.println("23 >= 23 = " + (23 >= 23)); // gröSSer oder gleich
55       System.out.println("45 <= 44 = " + (45 <= 44)); // kleiner oder gleich
56     }
57
58     private static void boolesche() {
59       System.out.println("\nBoolesche Operatoren:");
60       System.out.println("!true = "  + !true);          // Negation
61       System.out.println("true && true = "  + (true && true));  // Und, ture 2, genau
             dann wenn alle Argumente true sind
62       System.out.println("true || false = "  + (true || false));  // Oder, true, wenn
             mindestens ein Operand true ist
63       System.out.println("true ^ true = "  + (true ^ true));    // Xor, true, wenn
             genau ein Operand true ist
64     }
```

```
65
66    private static void bitshifting() {
67      int bit = ~0b10111011 & 0xff;
68      System.out.println("\nBitweise Operatoren:");
69      System.out.println("0b10111011 = ~0b" + Integer.toString(bit, 2)); //Invertiert
            die Bits
70      System.out.println("0b10111011 = ~0b01000100"); //Invertiert die Bits
71      System.out.println("0b10101010 & 0b11111111 = " + Integer.toString(0b10101010 &
            0b11111111, 2)); // Verundet die Bits
72      System.out.println("0b10101010 | 0b01101001 = " + Integer.toString(0b10101010 |
            0b00101001, 2)); // Verodert die Bits
73      System.out.println("0b10101010 ^ 0b11111111 = " + Integer.toString(0b10101010 ^
            0b11111111, 2)); // Exklusives oder
74      System.out.println("0b10101010 >> 2 = " + Integer.toString(0b10101010 >> 2, 2))
            ;          // Rechtsshift
75      System.out.println("0b10101010 >>> 1 = " + Integer.toString(0b10101010 >>> 1,
            2));          // Rechtsshift mit Nullen auffüllen
76      System.out.println("0b10101010 << 1 = " + Integer.toString(0b10101010 << 1, 2))
            ;          // Linksverschiebung
77    }
78
79    private static void zuweisung() {
80      int a = 20;
81      System.out.println("\nZuweisung Operatoren:");
82      System.out.println("int a = 20");           // Einfache zuweisung
83      System.out.println("a += 10 => " + (a += 10));    // Addiert ein wert zu der
            Variable
84      System.out.println("a -= 20 => " + (a -= 20));    // Subtrahiert ein wert zu
            der Variable
85      System.out.println("a *= 7 => " + (a *= 7));    // Dividiert die Variable durch
             den angegebenen Wert und weist ihn zu
86      System.out.println("a /= 5 => " + (a /= 5));    // Multipliziert die Variable
            durch den angegebenen Wert und weist ihn zu
87      System.out.println("a %= 5 => " + (a %= 5));    // Ermittelt den Rest und weist
             ihn zu
88      System.out.println("a &= 12 => " + (a &= 12));    // Eine bitweise Verundung
89      System.out.println("a |= 10 => " + (a |= 10));    // Bitweise Veroderung
90      System.out.println("a ^= 30 => " + (a ^= 30));    // Exklusives oder auf Bit
            ebene
91      System.out.println("a <<= 3 => " + (a <<= 3));    // Linksverschiebung
92      System.out.println("a >>= 1 => " + (a >>= 1));    // Rechtsverschiebung
93      System.out.println("a >>>= 2 => " + (a >>>= 2));  // Rechtsverschiebung und
            Auffüllen mit Nullen
94    }
95
96  }
```

5.2.5 Testdokumentation

Es wurden alle Berechnungen korrekt ausgeführt.

5.2.6 Benutzungshinweise

Keine Besonderen Benutzungshinweise. Das Programm muss lediglich nur ausgeführt werden.

5.2.7 Anwendungsbeispiel

Nach dem man das Programm gestartet hat, sollte folgende Ausgabe erscheinen:

```
1 [sebastian@laptop bin]$ java Operatoren
2 Arithmeschie Operatoren:
3 23 + 34 = 57
4 54 - 32 = 22
```

```
12 * 30 = 360
56 / 12 = 4
74 % 2  = 0
int i = +3 = 3
int n = -i = -3

Inkrement Operatoren:
x   = 10
x++ = 10
x   = 11
++x = 12
x   = 12
x-- = 12
x   = 11
--x = 10
x   = 10

Vergleichs Operatoren:
37 == 2 = false
1 != 2 = true
13 > 3 = true
23 < 2 = false
23 >= 23 = true
45 <= 44 = false

Boolische Operatoren:
!true = false
true && true = true
true || false = true
true ^ true = false

Bitweise Operatoren:
0b10111011 = ~0b01000100
0b10101010 & 0b11111111 = 10101010
0b10101010 | 0b01101001 = 10101011
0b10101010 ^ 0b11111111 = 1010101
0b10101010 >> 2 = 101010
0b10101010 >>> 1 = 1010101
0b10101010 << 1 = 101010100

Zuweisungs Operatoren:
int a = 20
a += 10 => 30
a -= 20 => 10
a *= 7 => 70
a /= 5 => 14
a %= 5 => 4
a &= 12 => 4
a |= 10 => 14
a ^= 30 => 16
a <<= 3 => 128
a >>= 1 => 64
a >>>= 2 => 16
[sebastian@laptop bin]$
```

6 Kapitel 6

6.1 Teilaufgabe 1

6.1.1 Aufgabenstellung

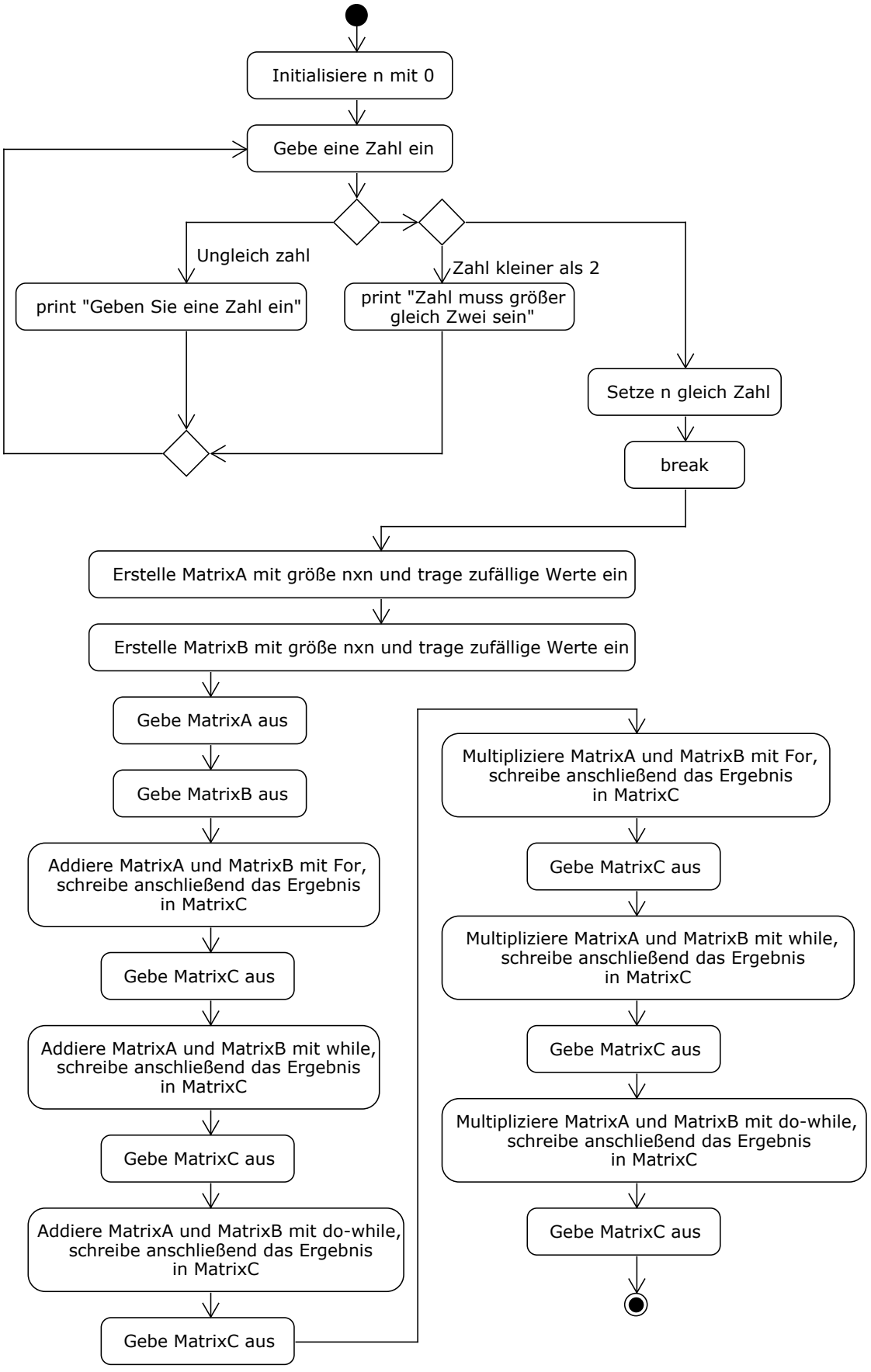
Im Grunde ist es die Selbe Aufgabe wie aus Kapitel 4, Teilaufgabe 2. Doch jetzt solle es auch für nxn Matrizen funktionieren. Die größ gibt an ende der Nutzer ein. Zusätzlich soll noch die Multiplikation der Matrizen auch mit while und do-while gelöst werden.

6.1.2 Anforderungsdefinition

1. Unser Input für die Größ der nxn Matrix.
2. Multiplikation mit for, while, und do-while.

6.1.3 Entwurf

Zuerst wird der Nutzer aufgefordert eine Zahl einzugeben. AnschlieSSend wird in einer do while Schleife geprüft ob die Nutzereingabe valide ist. Es wird einmal geprüft ob es sich um Buchstaben handelt und ob die Zahl kleiner als 2 ist. Wenn der Fall nicht eintritt, wird die Schleife beendet und es werden zwei nxn Matrizen erzeugt. Der Rest ist identisch wie aus Kapitel 4, nur das hier noch für die Berechnung eine weitere while und do while Schleife verwendet wird


Initialisiere n mit 0
Gebe eine Zahl ein
Ungleich zahl
Zahl kleiner als 2
print "Geben Sie eine Zahl ein"
print "Zahl muss größer gleich Zwei sein"
Setze n gleich Zahl
break
Erstelle MatrixA mit größe nxn und trage zufällige Werte ein
Erstelle MatrixB mit größe nxn und trage zufällige Werte ein
Gebe MatrixA aus
Gebe MatrixB aus
Addiere MatrixA und MatrixB mit For, schreibe anschließend das Ergebnis in MatrixC
Gebe MatrixC aus
Addiere MatrixA und MatrixB mit while, schreibe anschließend das Ergebnis in MatrixC
Gebe MatrixC aus
Addiere MatrixA und MatrixB mit do-while, schreibe anschließend das Ergebnis in MatrixC
Gebe MatrixC aus
Multipliziere MatrixA und MatrixB mit For, schreibe anschließend das Ergebnis in MatrixC
Gebe MatrixC aus
Multipliziere MatrixA und MatrixB mit while, schreibe anschließend das Ergebnis in MatrixC
Gebe MatrixC aus
Multipliziere MatrixA und MatrixB mit do-while, schreibe anschließend das Ergebnis in MatrixC
Gebe MatrixC aus

6.1.4 Quelltext

6.1.4.1 Matrizen.java

```
1  package chapter_06;
2
3  import java.util.Random;
4  import java.util.Scanner;
5  /**
6   * Klasse mit der Main-Methode
7   * Addiert und Multipliziert Matrizen
8   * @author Sebastian
9   *
10  */
11 public class Matrizen {
12
13   public static void main(String[] args) {
14     int[][] matrixA;
15     int[][] matrixB;
16
17     //Hier können sie die GröSSe definieren, z.B. 2,3 oder 5
18     System.out.println("Dieses Programm berechnet eine zufällig erstellte nxn
           Matrix");
19     System.out.print("Geben sie n an: ");
20     Scanner sc = new Scanner(System.in);
21
22     //Prüft ob der Userinput eine Zahl ist und ob diese gröSSer als Eins ist
23     int n = 0;
24     boolean isInt;
25     do {
26       isInt = sc.hasNextInt();
27       if (!isInt) {
28         System.out.println("Es dürfen nur Zahlen verwendet werden");
29         sc.next();
30       } else if ((n = sc.nextInt()) < 2)
31         System.out.println("Die Zahl muss gröSSer gleich 2 sein");
32       else
33         break;
34     } while (true);
35     /*
36      * Initialisierungsmethode wird mit dem Wert n aufgerufen.
37      * AnschlieSSend wird diese Matrix erzeugt und mit
38      * zufällig generierten Zahlen befüllt.
39      */
40     matrixA = initialize(n);
41     matrixB = initialize(n);
42
43     /*
44      * Zuerst werden die Beiden Matrizen A und B jeweils ausgegeben
45      */
46     System.out.println("Matrix A:");
47     printMatrix(matrixA);
48     System.out.println("Matrix B:");
49     printMatrix(matrixB);
50     /*
51      * AnschlieSSend werden die Matrizen hier Addiert
52      */
53     System.out.println("Addition von A und B:");
54     printMatrix(addition(matrixA, matrixB));
55     /*
56      * Und hier Multipliziert
57      */
58     System.out.println("Multiplikation von A und B:");
59     System.out.println("For Schleife");
```

```
60      printMatrix(multiplikationFor(matrixA, matrixB));
61      System.out.println("While Schleife");
62      printMatrix(multiplikationWhile(matrixA, matrixB));
63      System.out.println("Do-While Schleife");
64      printMatrix(multiplikationDoWhile(matrixA, matrixB));
65
66      sc.close();
67    }
68
69    /**
70     * Initialisierung des Arrays
71     * @param n Die grösSe der nxn Matrix
72     * @return matrix
73     */
74    private static int[][] initialize(int n) {
75      int[][] matrix = new int[n][n];
76      /*
77       * Bei der Initialisierung wird einmal durch das gesamt Array durch iteriert.
78       * Dabei werden dann mit Math.random() zufällige Zahlen rein geschrieben.
79       */
80
81      Random rand = new Random();
82      for (int i = 0; i < matrix.length; ++i)
83        for (int l = 0; l < matrix[i].length; ++l)
84          matrix[i][l] = rand.nextInt(200) - 100;
85
86      return matrix;
87    }
88
89    /**
90     * Addition der beiden Matrizen A und B
91     * @param matrixA
92     * @param matrixB
93     * @return Gibt ein neues Array mit den Addierten Werten zurück
94     */
95    private static int[][] addition(int[][] matrixA, int[][] matrixB) {
96      int[][] matrixAd = new int[matrixA.length][matrixA[0].length]; //Es wird ein
            neues Temporäres Array angelegt
97
98      for (int i = 0; i < matrixA.length; ++i) {
99        for (int n = 0; n < matrixA[i].length; ++n) {
100         matrixAd[i][n] = matrixA[i][n] + matrixB[i][n];
101       }
102     }
103
104     return matrixAd;
105   }
106
107   /**
108    * Multiplikation der beiden Matrizen A und B
109    * @param matrixA
110    * @param matrixB
111    * @return Gibt ein neues Array mit den Multiplizierten Werten zurück
112    */
113   private static int[][] multiplikationFor(int[][] matrixA, int[][] matrixB) {
114     int[][] matrixMult = new int[matrixB.length][matrixB[0].length];
115
116     //Hier die Variante mit For Schleifen
117     for (int HmatrixB = 0; HmatrixB < matrixB.length; ++HmatrixB)
118       for (int WmatrixB = 0; WmatrixB < matrixB[HmatrixB].length; ++WmatrixB)
119         for (int WmatrixA = 0; WmatrixA < matrixB.length; ++WmatrixA)
120           matrixMult[HmatrixB][WmatrixB] += matrixA[HmatrixB][WmatrixA] * matrixB[
              WmatrixA][WmatrixB];
```

```
121
122       return matrixMult;
123     }
124
125     /**
126      * Multiplikation der beiden Matrizen A und B
127      * @param matrixA
128      * @param matrixB
129      * @return Gibt ein neues Array mit den Multiplizierten Werten zurück
130      */
131     private static int[][] multiplikationWhile(int[][] matrixA, int[][] matrixB) {
132       int[][] matrixMult = new int[matrixB.length][matrixB[0].length];
133
134       int HmatrixB = 0;
135       int WmatrixB = 0;
136       int WmatrixA = 0;
137
138       //Hier die Variante mit While Schleifen
139       while (HmatrixB < matrixB.length) {
140         WmatrixB = 0;
141         while (WmatrixB < matrixB[HmatrixB].length) {
142           WmatrixA = 0;
143           while (WmatrixA < matrixB.length) {
144             matrixMult[HmatrixB][WmatrixB] += matrixA[HmatrixB][WmatrixA] * matrixB[
                    WmatrixA][WmatrixB];
145             ++WmatrixA;
146           }
147           ++WmatrixB;
148         }
149         ++HmatrixB;
150       }
151       return matrixMult;
152     }
153
154     /**
155      * Multiplikation der beiden Matrizen A und B
156      * @param matrixA
157      * @param matrixB
158      * @return Gibt ein neues Array mit den Multiplizierten Werten zurück
159      */
160     private static int[][] multiplikationDoWhile(int[][] matrixA, int[][] matrixB) {
161       int[][] matrixMult = new int[matrixB.length][matrixB[0].length];
162
163       int HmatrixB = 0;
164       int WmatrixB = 0;
165       int WmatrixA = 0;
166
167       //Hier die Variante mit Do-While Schleifen
168       do {
169         WmatrixB = 0;
170         do {
171           WmatrixA = 0;
172           do {
173             matrixMult[HmatrixB][WmatrixB] += matrixA[HmatrixB][WmatrixA] * matrixB[
                    WmatrixA][WmatrixB];
174             ++WmatrixA;
175           } while (WmatrixA < matrixB.length);
176           ++WmatrixB;
177         } while (WmatrixB < matrixB[HmatrixB].length);
178         ++HmatrixB;
179       } while (HmatrixB < matrixB.length);
180
181       return matrixMult;
```

```
182     }
183
184     /**
185      * Hier wird die Matrix ausgegeben
186      * @param matrix
187      */
188     private static void printMatrix(int[][] matrix) {
189       for (int[] y : matrix) {
190         for (int x: y)
191           System.out.print(x + "\t");
192         System.out.println();
193       }
194       System.out.println();
195     }
196
197 }
```

6.1.5 Testdokumentation

Bei Zahlen die kleiner als 2 waren, sowie Buchstaben, kam es zu einer entsprechenden Fehlermeldung. AnschlieSSend konnte man den Wert erneut eintippen. Bei den Ergebnissen der Matrizen wurden die Werte mit einem Onlinerechner verglichen, hierbei hat alles übereingestimmt.

6.1.6 Benutzungshinweise

Nach dem aufrufen des Programms, wird der Nutzer aufgefordert eine Zahl einzugeben. Diese muss größer als ein sein.

6.1.7 Anwendungsbeispiel

Nach dem man das Programm gestartet hat, sollte folgende Ausgabe erscheinen:

```
1  [sebastian@laptop bin]$ java Matrizen
2  Dieses Programm berechnet eine zufällig erstellte nxn Matrix
3  Geben sie n an: -45
4  Die Zahl muss grösSer gleich 2 sein
5  test
6  Es dürfen nur Zahlen verwendet werden
7  5
8  Matrix A:
9  78  88  96  91  50
10 17  16  14  17  77
11 34  1 45  21  42
12 63  1 92  76  57
13 84  59  13  85  46
14
15 Matrix B:
16 38  36  45  22  18
17 96  4 0 93  79
18 86  38  92  50  92
19 1 54  63  71  10
20 4 62  91  31  55
21
22 Addition von A und B:
23 116 124 141 113 68
24 113 20  14  110 156
25 120 39  137 71  134
26 64  55  155 147 67
27 88  121 104 116 101
28
29 Multiplikation von A und B:
30 For Schleife
```

```
19959 14822 22625 22711 20848
3711  6900  10131 6156  7263
5447  6676  10815 5884  7351
10706 13406 21274 13242 13572
10243 11196 14517 15446 10749

While Schleife
19959 14822 22625 22711 20848
3711  6900  10131 6156  7263
5447  6676  10815 5884  7351
10706 13406 21274 13242 13572
10243 11196 14517 15446 10749

Do-While Schleife
19959 14822 22625 22711 20848
3711  6900  10131 6156  7263
5447  6676  10815 5884  7351
10706 13406 21274 13242 13572
10243 11196 14517 15446 10749
[sebastian@laptop bin]$
```

6.2 Teilaufgabe 2

6.2.1 Aufgabenstellung

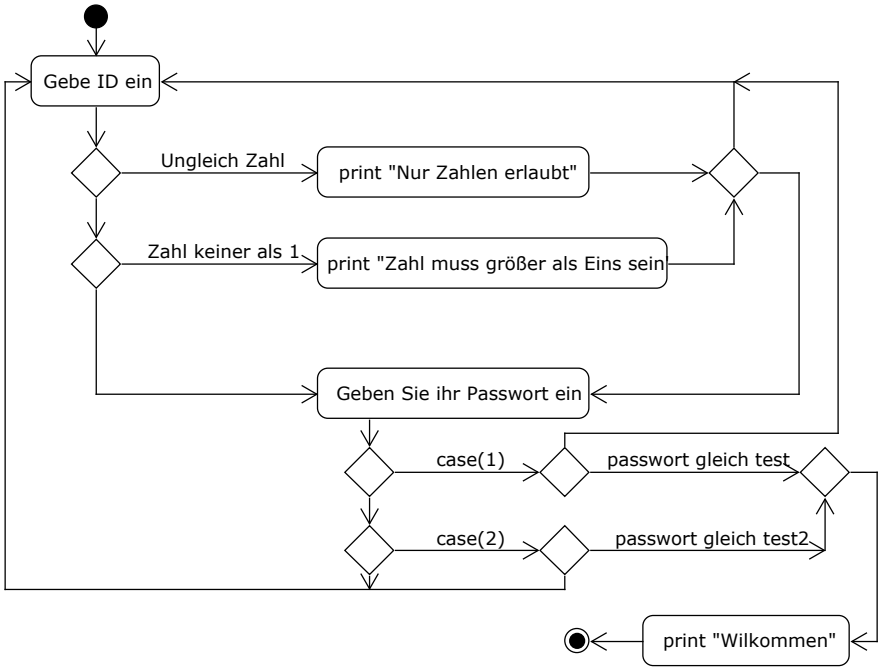
In der zweiten Teilaufgabe sollten wir ein Programm schreiben welches Sprunganweisungen in Java Sinnvoll verdeutlichen.

6.2.2 Anforderungsdefinition

1. Verwendung von Sprunganweisungen.
2. Mindestens ein switch-Anweisung.

6.2.3 Entwurf

Im folgenden ist ein Entwurf für ein einfaches Login. Zuerst wird der Nutzer aufgefordert seine ID anzugeben. Dabei wird geprüft ob es sich um eine Zahl handelt und ob diese Zahl größer gleich Eins ist. Anschließend wird wird in einem switch überprüft ob eine solche ID existiert und ob das angegebene Passwort auch das richtige ist. Wenn alles Stimmt wird eine Willkommens Nachricht ausgegeben. Andernfalls wird der Nutzer aufgefordert alles erneut einzutippen.



6.2.4 Quelltext

6.2.4.1 Sprunganweisungen.java

```java
package chapter_06;

import java.util.Scanner;

/**
 * Klasse mit der Main-Methode
 * @author Sebastian
 *
 */
public class Sprunganweisungen {

  public static void main(String[] args) {
    login();
  }

  /**
   * Kleine einfache Implementierung von Nutzern, mithilfe
   * einer switch-Anweisung
   * @param userID ID die beim login eingegeben wurde
   * @param userPw Password was bei login eingegeben wurde
   * @return Wenn die ID und das Passwort übereinstimmen,
   * wird true zurück gegeben
   */
  private static boolean userData(int userID, String userPw) {
    return switch (userID) {
      case 1 -> userPw.equals("hallo");
      case 112 -> userPw.equals("das");
      case 124 -> userPw.equals("ist");
      case 345 -> userPw.equals("nicht");
      case 653 -> userPw.equals("geheim");
      default -> false;
    };
  }

```

```
35    /**
36     * Hier befindet sich das Login feld
37     */
38    private static void login() {
39      int id;
40      Scanner sc = new Scanner(System.in);
41      do {
42        System.out.println("Willkommen...!");
43        System.out.print("ID      : ");
44
45        /*
46         * Eine kleine Abfrage die Prüft, ob die eingegebene
47         * ID nur aus Zahlen besteht
48         */
49        do {
50          if (!sc.hasNextInt()) {
51            System.out.println("Error, es dürfen nur Zahlen enthalten sein.");
52            sc.next();
53          } else if ((id = sc.nextInt()) < 1)
54            System.out.println("Die Zahl muss gröSSer gleich 1 sein");
55          else
56            break;
57          System.out.print("ID      : ");
58        } while (true);
59
60        System.out.print("Passwort: ");
61        /*
62         * Wenn ein Nutzer mit dem angegebenen Passwort
63         * nicht existiert, wird die ID zurückgesetzt
64         * und eine Fehlermeldung wird ausgegeben
65         */
66        if(!userData(id, sc.next())) {
67          System.out.println("Ihre Angaben sind leider falsch, versuchen Sie es
               erneut.");
68        } else
69          break;
70
71      /*
72       * Die Schleife wird solange durchlaufen, bis sich ein nutzer
73       * erfolgreich angemeldet hat.
74       */
75      } while (true);
76
77      System.out.println("Juhu, Sie haben sich eingeloggt");
78      sc.close();
79    }
80  }
```

6.2.5 Testdokumentation

Bei Zahlen die kleiner als 1 waren, sowie Buchstaben, kam es zu einer entsprechenden Fehlermeldung. AnschlieSSend konnte man den Wert erneut eintippen.

6.2.6 Benutzungshinweise

Nach dem aufrufen des Programms, wird der Nutzer aufgefordert seine NutzerID anzugeben, sowie anschließend sein Passwort. Bei inkorrekter eingaben, wird man erneut aufgefordert die Daten einzutippen.

6.2.7 Anwendungsbeispiel

Bei Erfolgreicher Anmeldung:

```
[sebastian@laptop bin]$ java Sprunganweisungen
Wilkommen...!
ID      : 1
Passwort: hallo
Juhu, Sie haben sich eingeloggt
[sebastian@laptop bin]$
```

Bei inkorrekter Anmeldung:

```
[sebastian@laptop bin]$ java Sprunganweisungen
Wilkommen...!
ID      : 12
Passwort: qwert
Ihre Angaben sind leider falsch, versuchen Sie es erneut.
ID      : -1
Passwort: hallo
Die Zahl muss grösS er gleich 1 sein
ID      :
[sebastian@laptop bin]$
```

7 Kapitel 7

7.1 Aufgabenstellung

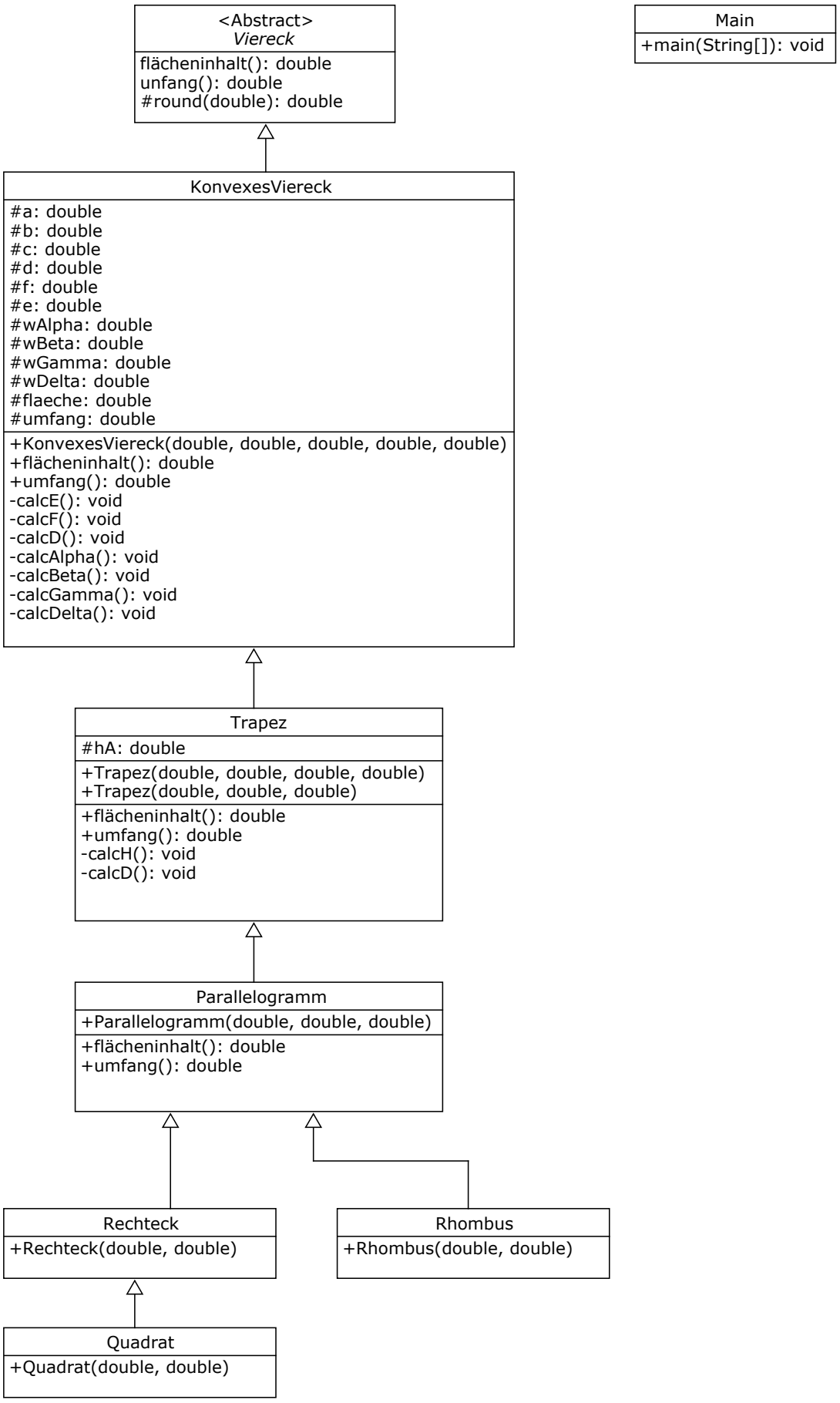
Wie sollen 7 Klassen Definieren, dabei sollen wir uns überlegen welche davon Instanziierbar sind und welche nicht. Anschließend soll jede Klasse sinnvolle Methoden und Attribute enthalten.

7.2 Anforderungsdefinition

1. Definiere folgende Klassen: Viereck, konvexes Viereck, Trapez, Parallelogramm, Rhombus, Rechteck, Quadrat
2. Definiere sinnvolle Methoden und Attribute

7.3 Entwurf

Für den Entwurf wurde ein Klassendiagramm angefertigt. Wobei hier die Klasse Viereck Abstrakt ist und die Wesentlichen Methoden beinhaltet. Beim Konvexen Viereck werden die wesentlichen Attribute eines Vierecks definiert, dazu auch deren Berechnung. Zudem muss hier auch die Methode für den Flächeninhalt und Umfang überschrieben werden. Das Trapez erbt vom Konvexen Viereck, Parallelogramm vom Trapez, Rhombus vom Parallelogramm, Rechteck von Parallelogramm ud Quadrat vom Rechteck.


<Abstract>
Viereck
flächeninhalt(): double
unfang(): double
#round(double): double
Main
+main(String[]): void
KonvexesViereck
#a: double
#b: double
#c: double
#d: double
#f: double
#e: double
#wAlpha: double
#wBeta: double
#wGamma: double
#wDelta: double
#flaeche: double
#umfang: double
+KonvexesViereck(double, double, double, double, double)
+flächeninhalt(): double
+umfang(): double
-calcE(): void
-calcF(): void
-calcD(): void
-calcAlpha(): void
-calcBeta(): void
-calcGamma(): void
-calcDelta(): void
Trapez
#hA: double
+Trapez(double, double, double, double)
+Trapez(double, double, double)
+flächeninhalt(): double
+umfang(): double
-calcH(): void
-calcD(): void
Parallelogramm
+Parallelogramm(double, double, double)
+flächeninhalt(): double
+umfang(): double
Rechteck
+Rechteck(double, double)
Rhombus
+Rhombus(double, double)
Quadrat
+Quadrat(double, double)

7.4 Quelltext

7.4.1 Main.java

```java
package chapter_07;

import chapter_07.figures.*;

public class Main {

    public static void main(String[] args) {

        KonvexesViereck konvexesViereck = new KonvexesViereck(60, 80, 50, 60, 70);
        System.out.println(konvexesViereck.umfang() + " " + konvexesViereck.
            flächeninhalt());

        Trapez trapez = new Trapez(10, 5, 5, 80);
        System.out.println(trapez.umfang() + " " + trapez.flächeninhalt());

        Trapez trapez2 = new Trapez(10, 20, 10);
        System.out.println(trapez2.umfang() + " " + trapez2.flächeninhalt());

        Parallelogramm parallelogramm = new Parallelogramm(10, 20, 20);
        System.out.println(parallelogramm.umfang() + " " + parallelogramm.
            flächeninhalt());

        Rhombus rhombus = new Rhombus(8, 30);
        System.out.println(rhombus.umfang() + " " + rhombus.flächeninhalt());

        Rechteck rechteck = new Rechteck(5, 10);
        System.out.println(rechteck.umfang() + " " + rechteck.flächeninhalt());

        Quadrat quadrat = new Quadrat(5);
        System.out.println(quadrat.umfang() + " " + quadrat.flächeninhalt());
    }
}
```

7.4.2 Viereck.java

```java
package chapter_07.figures;

abstract class Viereck {

    abstract double flächeninhalt();
    abstract double umfang();

    protected double round(double value) {
        return Math.round(value * 10000.0) / 10000.0;
    }
}
```

7.4.3 KonvexesViereck.java

```java
package chapter_07.figures;

public class KonvexesViereck extends Viereck {
    protected double a = 0;
```

```
5       protected double b = 0;
6       protected double c = 0;
7       protected double d = 0;
8
9       protected double f = 0;
10      protected double e = 0;
11
12      protected double wAlpha = 0;
13      protected double wBeta = 0;
14      protected double wGamma = 0;
15      protected double wDelta = 0;
16
17      protected double flaeche;
18      protected double umfang;
19
20      /**
21       * Konstruktor, hier werden die Attribute initialisiert
22       * @param a
23       * @param b
24       * @param c
25       * @param winkelBeta
26       * @param winkelGamma
27       */
28      public KonvexesViereck(double a, double b, double c, double winkelBeta, double
            winkelGamma) {
29          this.a = a;
30          this.b = b;
31          this.c = c;
32
33          this.wBeta = winkelBeta;
34          this.wGamma = winkelGamma;
35
36          calcE();
37          calcF();
38          calcD();
39          calcAlpha();
40          calcDelta();
41
42          try {
43              if (wAlpha + wBeta + wGamma + wDelta < 359.99)
44                  throw new Exception("Error, die Summe der Winkel ist under 360 Grad
                        " + (wAlpha + wBeta + wGamma + wDelta));
45          } catch (Exception exception) {
46              exception.printStackTrace();
47              //System.exit(0);
48          }
49
50      }
51
52      @Override
53      /**
54       * Gibt de flächeninhalt zurück
55       */
56      public double flächeninhalt() {
57          if (flaeche != 0)
58              return flaeche;
59          else
60              //Wenn der Flächeninhalt 0 ist, wird die fläche berechnet
61              return flaeche = (Math.sqrt(((a+f+d)/2) * ((a+f+d)/2-a) * ((a+f+d)/2-f)
                        * ((a+f+d)/2-d)) + Math.sqrt(((b+c+f)/2) * ((b+c+f)/2-b) * ((b+c+f)
                    /2-c) * ((b+c+f)/2-f)));
62      }
63
```

```
64      @Override
65      /**
66       * Gibt den umfang zurück
67       */
68      public double umfang() {
69          if (umfang != 0)
70              return umfang;
71          else
72              //Wenn der Umfang 0 ist, wird der umfang berechnet
73              return umfang = a + b + c + d;
74      }
75
76      /**
77       * Berechnet die Diagonale E
78       */
79      private void calcE() {
80          if (wBeta != 0)
81              e = Math.sqrt(a*a + b*b - 2 * a * b * Math.cos(Math.toRadians(wBeta)));
82          else
83              e = Math.sqrt(c*c + d*d - 2 * c * d * Math.cos(Math.toRadians(wDelta)))
                    ;
84      }
85
86      /**
87       * Berechnet die Diagonale F
88       */
89      private void calcF() {
90          if (wAlpha != 0)
91              f = Math.sqrt(a*a + d*d - 2 * a * b * Math.cos(Math.toRadians(wAlpha)))
                    ;
92          else
93              f = Math.sqrt(b*b + c*c - 2 * b * c * Math.cos(Math.toRadians(wGamma)))
                    ;
94      }
95
96      /**
97       * Berechnet die Seite D
98       */
99      private void calcD() {
100         double tmp = wBeta - Math.toDegrees(Math.acos((f*f + b*b - c*c) / (2 * f *
                b)));
101         d = Math.sqrt(a*a + f*f - 2 * a * f * Math.cos(Math.toRadians(tmp)));
102     }
103
104     /**
105      * Berechnet den Winkel Alpha
106      */
107     private void calcAlpha() {
108
109         wAlpha = round(Math.toDegrees(Math.acos((a*a + d*d - f*f) / (2 * a * d))));
110     }
111
112     /**
113      * Berechnet den Winkel Beta
114      */
115     private void calcBeta() {
116         wBeta = round(Math.toDegrees(Math.acos((a*a + b*b - e*e) / (2 * a * b))));
117     }
118
119     /**
120      * Berechnet den Winkel Gamma
121      */
122     private void calcGamma() {
```

```
123         wGamma = round(Math.toDegrees(Math.acos((b*b + c*c - f*f) / (2 * b * c))));
124     }
125 
126     /**
127      * Berechnet den Winkel Delta
128      */
129     private void calcDelta() {
130         wDelta = round(Math.toDegrees(Math.acos((c*c + d*d - e*e) / (2 * c * d))));
131     }
132 }
```

7.4.4 Trapez.java

```
1  package chapter_07.figures;
2  
3  public class Trapez extends KonvexesViereck {
4  
5      protected double hA;
6  
7      /**
8       * Konstruktor, hier werden die Attribute initialisiert
9       * @param a
10      * @param b
11      * @param c
12      * @param winkelBeta
13      */
14     public Trapez(double a, double b, double c, double winkelBeta) {
15         super(a, b, c, winkelBeta, 180 - winkelBeta);
16         /*System.out.println(wAlpha);
17         System.out.println(wBeta);
18         System.out.println(wGamma);
19         System.out.println(wDelta);*/
20         try {
21             if (c > a)
22                 throw  new Exception("Seite C muss kleiner als seine A sein");
23             if (wAlpha > 90.00 || wBeta > 90.00 || wGamma <= 90.00 || wDelta <=
                   90.00)
24                 throw new Exception("Alpha oder Beta ist grösSer als 90 Grad");
25         } catch (Exception exception) {
26             exception.printStackTrace();
27         }
28         calcH();
29         calcD();
30     }
31 
32     /**
33      * Konstruktor für ein Rechtwinkliges Trapez
34      * @param a
35      * @param b
36      * @param c
37      */
38     public Trapez (double a, double b, double c) {
39         super(a, b, c, 90, 90);
40 
41         calcH();
42         calcD();
43     }
44 
45     @Override
46     /**
47      * Gibt de flächeninhalt zurück
```

```
48          */
49         public double flächeninhalt() {
50             if (flaeche != 0)
51                 return flaeche;
52             else
53                 //Wenn der Flächeninhalt 0 ist, wird die fläche berechnet
54                 return flaeche = ((a + c) * hA) / 2;
55         }
56 
57         @Override
58         /**
59          * Gibt den umfang zurück
60          */
61         public double umfang() {
62             if (umfang != 0)
63                 return umfang;
64             else
65                 //Wenn der Umfang 0 ist, wird der umfang berechnet
66                 return umfang = a + b + c + d;
67         }
68 
69         /**
70          * Berechnet die Höhe
71          */
72         private void calcH() {
73             hA = b * Math.sin(Math.toRadians(wBeta));
74         }
75 
76         /**
77          * Berechnet die Seite D
78          */
79         private void calcD() {
80             d = Math.sqrt(Math.pow(a - c - Math.sqrt(b*b - hA*hA), 2) + hA*hA);
81         }
82 
83 }
```

7.4.5 Parallelogramm.java

```
1  package chapter_07.figures;
2 
3  public class Parallelogramm extends Trapez {
4 
5      /**
6       * Konstruktor, hier werden die Attribute initialisiert
7       * @param a
8       * @param b
9       * @param winkelAlpha
10      */
11     public Parallelogramm(double a, double b, double winkelAlpha) {
12         super(a, b, a);
13         wAlpha = winkelAlpha;
14         wBeta = 180 - winkelAlpha;
15     }
16 
17     @Override
18     /**
19      * Gibt de flächeninhalt zurück
20      */
21     public double flächeninhalt() {
22         if (flaeche != 0)
```

```
23              return flaeche;
24          else
25              //Wenn der Flächeninhalt 0 ist, wird die fläche berechnet
26              return flaeche = a * b * Math.sin(Math.toRadians(wAlpha));
27      }
28
29      @Override
30      /**
31       * Gibt den umfang zurück
32       */
33      public double umfang() {
34          if (umfang != 0)
35              return umfang;
36          else
37              //Wenn der Umfang 0 ist, wird der umfang berechnet
38              return umfang = 2*a + 2*b;
39      }
40  }
```

7.4.6 Rhombus.java

```
1   package chapter_07.figures;
2
3   public class Rhombus extends Parallelogramm {
4
5       /**
6        * Konstruktor, hier werden die Attribute initialisiert
7        * @param a
8        * @param hA
9        */
10      public Rhombus(double a, double winkelAlpha) {
11          super(a, a, winkelAlpha);
12      }
13  }
```

7.4.7 Rechteck.java

```
1   package chapter_07.figures;
2
3   public class Rechteck extends Parallelogramm {
4
5       /**
6        * Konstruktor, hier werden die Attribute initialisiert
7        * @param a
8        * @param b
9        */
10      public Rechteck(double a, double b) {
11          super(a, b, 90);
12      }
13  }
```

7.4.8 Quadrat.java

```
1   package chapter_07.figures;
2
3   public class Quadrat extends Rechteck {
```

```
4
5       /**
6        * Konstruktor, hier werden die Attribute initialisiert
7        * @param a
8        */
9       public Quadrat(double a) {
10          super(a, a);
11      }
12  }
```

7.5 Testdokumentation

7.6 Benutzungshinweise

7.7 Anwendungsbeispiel

Nach dem man das Programm gestartet hat, sollte folgende Ausgabe erscheinen:

```
1   [sebastian@laptop bin]$ java Main
2   223.2733105547887 2808.779545976002
3   35.17638090205041 49.99999999999999
4   50.0 120.0
5   32.0 45.256
6   30.0 50.0
7   20.0 25.0
8   [sebastian@laptop bin]$
```

8 Kapitel 8

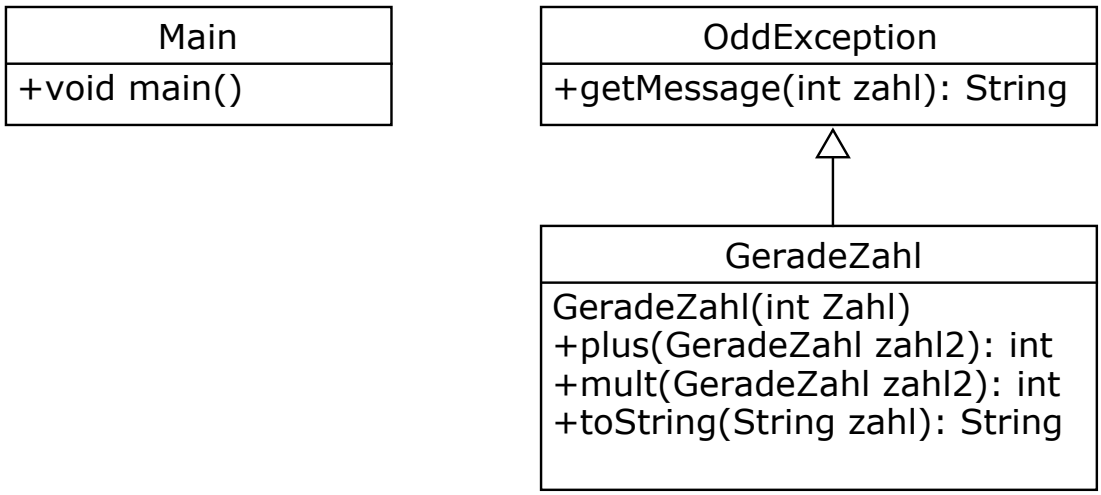
8.1 Aufgabenstellung

Wir sollen ein Klasse schreiben die Nur Gerade zahlen annimmt, andernfalls soll eine Exception geworfen werden. Zusätzlich kommen noch zwei Methoden in diese Klasse für die Multiplikation und Addition.

8.2 Anforderungsdefinition

1. Erstelle eine Klasse GeradeZahl.
2. Erstelle eine Methode für die Multiplikation und Addition.
3. Bei ungeraden Zahlen soll eine Exception geworfen werden.

8.3 Entwurf



8.4 Quelltext

8.4.1 Main.java

```java
package chapter_08;

/**
 * Klasse mit der Main-Methode
 * @author sebastian
 */
public class Main {

    public static void main(String[] args) throws OddException {
        GeradeZahl zahl1 = new GeradeZahl(10);
        GeradeZahl zahl2 = new GeradeZahl(21);
        GeradeZahl zahl3 = new GeradeZahl(30);
        GeradeZahl zahl4 = new GeradeZahl(13);

        System.out.println("Zahl 1 = " + zahl1.mult(zahl2));
        System.out.println("Zahl 2 = " + zahl2);
        System.out.println("Zahl 3 = " + zahl3.mult(zahl2));
        System.out.println("Zahl 4 = " + zahl4.plus(zahl1));
    }
}
```

8.4.2 GeradeZahl.java

```java
package chapter_08;

public class GeradeZahl {
    private int zahl1;

    /**
     * Konstruktor, hier wird geprüft ob es sich bei der Zahl um eine gerade Zahl
         handelt
     * @param zahl1
     */
    public GeradeZahl(int zahl1) throws OddException {
        this.zahl1 = zahl1;
```

```java

        //Wenn die Zahl ungerade ist, wird eine Exception geworfen
        if (zahl1%2 == 1) throw new OddException(zahl1);
    }

    /**
     * Addiert Zwei GeradeZahl objekte miteinander
     * @param zahl2
     * @return Git eine int Zahl zurück
     */
    public int plus(GeradeZahl zahl2) {
        return zahl1 + zahl2.getZahl();
    }

    /**
     * Multipliziert Zwei GeradeZahl objekte miteinander
     * @param zahl2
     * @return Gibt eine int Zahl zurück
     */
    public int mult(GeradeZahl zahl2) {
        return zahl1 * zahl2.getZahl();
    }

    /**
     * @return Liefert die Zahl zurück
     */
    public int getZahl() {
        return zahl1;
    }

    /**
     * Bei einem Print wird diese Methode ausgeführt
     * @return Liefert einen String mit der Zahl zurück
     */
    @Override
    public String toString() {
        return "" + getZahl();
    }
}
```

8.4.3 OddException.java

```java
package chapter_08;

public class OddException extends Exception {
    int zahl;

    public OddException(int zahl) {
        this.zahl = zahl;
    }

    //@Override
    public String getMessage() {
        return "Error, " + zahl + " ist keine Gerade Zahl! Die Zahl wurde um Eins
            erhöht.";
    }


}
```

8.5 Testdokumentation

Bei einer Ungeraden Zahl wurde eine Exception geworfen und wie erwarte die Zahl anschlieSSend um eins erhöht.

8.6 Benutzungshinweise

Keine Besonderen Benutzungshinweise. Das Programm muss lediglich nur ausgeführt werden.

8.7 Anwendungsbeispiel

Bei einer ungeraden Zahl sollte diese Ausgabe erscheinen.

```
[sebastian@laptop bin]$ java Main
Exception in thread "main" chapter_08.OddException: Error, 21 ist keine Gerade Zahl
    ! Die Zahl wurde um Eins erhöht.
at chapter_08.GeradeZahl.<init>(GeradeZahl.java:14)
at chapter_08.Main.main(Main.java:11)
[sebastian@laptop bin]$
```

Und bei einer Geraden diese.

```
[sebastian@laptop bin]$ java Main
Zahl 1 = 200
Zahl 2 = 20
Zahl 3 = 600
Zahl 4 = 24
[sebastian@laptop bin]$
```

9 Kapitel 10

9.1 Aufgabenstellung

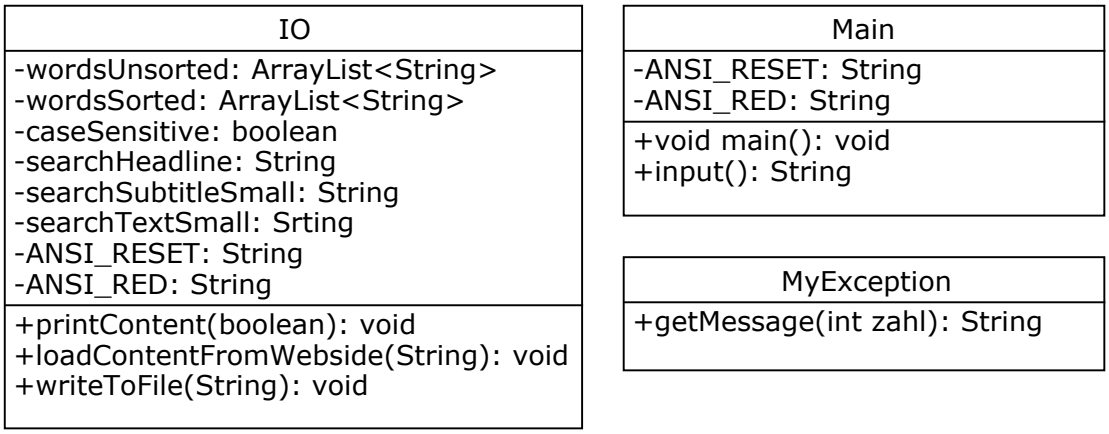
Wir sollen ein Programm schreiben Welches ein Artikel von der Tagesschau herunterlädt und anschließend in einer Datei abspeichert. Der Nutzer kann angeben, welche Datei er einlesen möchte, anschließend kann er die gezählten Wörter in Alphabetischer Reihenfolge ausgeben lassen. Zusätzlich soll über ein Startparameter entschieden werden, ob Groß- und Kleinschreibung beachtet wird.

9.2 Anforderungsdefinition

1. Ein Artikel der Tagesschau mit mehr als 500 Wörtern soll runter geladen werden.
2. Die Wörter sollen in einer geeigneten Collection abgelegt werden,
3. Die Wörter sollen Sortiert ausgegeben können.
4. Bei der Ausgabe der Wörte wird zusätzlich die Häufigkeit mit angegeben.
5. Beim Start soll entschieden werden ob Groß- oder Kleinschreibung.

9.3 Entwurf

Zum Entwurf wurde ein Klassendiagramm gewählt welches die Methoden und deren Attribute der jeweiligen Klassen veranschaulicht. DIe IO Klasse befasst sich mit dem Einlesen einer Datei, sowie auch das anlegen neuer Dateien. Des weiterem kann die Klasse auch einen Artikel von der Tagesschau heruntergeladen werden, hierfür muss nur ein Link angegeben werden. Wenn ein Artikel heruntergeladen wurden, kann dieser eingelesen werden und anschließend Sortiert ausgeben werden.



9.4 Quelltext

9.4.1 Main.java

```
1  package chapter_10;
2
3  import java.util.Scanner;
4
5  /**
6   * Klasse mit der Main-Methode
7   * @author sebastian
8   */
9  public class Main {
10     private static final String ANSI_RESET = "\u001B[0m";
11     private static final String ANSI_RED = "\u001B[31m";
12
13     public static void main(String[] args) throws MyException {
14         boolean caseSensitive;
15         /*
16          * Prüft ob bei dem aufruf des Programmes ein Parameter übergeben wurde.
17          * Wenn dies der fall ist und der Parameter true oder false ist, wird
                dieser
18          * in einer Variable gespeichert. Andernfalls wird eine exception geworfen.
19          */
20         if (args.length != 0 && (args[0].equalsIgnoreCase("true") || args[0].
               equalsIgnoreCase("false"))) {
21             caseSensitive = Boolean.parseBoolean(args[0]);
22         } else
23             throw new MyException();
24
25         IO io = new IO(caseSensitive);
26         int option;
27
28         Scanner sc = new Scanner(System.in);
29         do {
30             //Hier werden die auswahlmöglichkeiten ausgegeben
31             System.out.println("Bitte wähle einer der Folgenden Optionen aus:");
32             System.out.println("1) Einen Artikel von der Tagesschau herunterladen."
                   );
33             System.out.println("2) Eine Datei laden.");
34             System.out.println("3) Wörter ausgeben.");
35             System.out.println("0) Programm Beenden.");
36             do {
37                 /*
```

```
38                  * Wenn der Nutzer einen Buchstaben oder eine Zahl auSSerhalb des
                        wertebereiches
39                  * angibt wird eine entsprechende Fehlermeldung angegeben.
40                  */
41                 if (!sc.hasNextInt()) {
42                     System.out.println("Es sind nur Zahlen erlaubt");
43                     sc.next();
44                 } else if ((option = sc.nextInt()) > 4 || option < 0)
45                     System.out.println("Zahl auSSerhalb des Bereiches!");
46                 else
47                     break;
48             } while (true);
49
50             //Bei einer erfolgreichen eingabe, wird eine dieser Optionen ausgeführt
51             switch (option) {
52                 case 1:
53                     System.out.println("Bitte gebe den Link an:");
54                     io.loadContentFromWebsite(input());
55                     break;
56                 case 2:
57                     System.out.println("Bitte geben Sie den Datei namen an:");
58                     io.readFromFile(input());
59                     break;
60                 case 3:
61                     io.printContent(true);
62                     break;
63             }
64         } while (option > 0);
65
66         sc.close();
67     }
68
69     //Der Input wird hier zu einem String zusammengeführt
70     public static String input() {
71         Scanner sc = new Scanner(System.in);
72         String tmp;
73         tmp = sc.next();
74         tmp += sc.nextLine();
75         return tmp;
76     }
77 }
```

9.4.2 IO.java

```
1  package chapter_10;
2
3  import java.io.*;
4  import java.net.URL;
5  import java.nio.charset.Charset;
6  import java.nio.charset.StandardCharsets;
7  import java.util.ArrayList;
8  import java.util.Collections;
9  import java.util.Scanner;
10
11 public class IO{
12     private static ArrayList<String> wordsUnsorted;
13     private static ArrayList<String> wordsSorted;
14     private final boolean caseSensitive;
15
16     private static final String searchHeadline = "class=\"meldungskopf__headline--
           text\"";
17     private static final String searchSubtitleSmall = "class=\"meldung_";
```

```
18      private static final String searchTextSmall = "textabsatz columns twelve\"";
19
20      private static final String ANSI_RESET = "\u001B[0m";
21      private static final String ANSI_RED = "\u001B[31m";
22
23      public IO(boolean caseSensitive){
24          this.caseSensitive = caseSensitive;
25      }
26
27      /**
28       * Listet die einzelnen wörter aus dem ausgewählten Artikel auf und gibt
29       * zusätzlich der Anzahl an
30       * @param sorted Noch keinen nutzen
31       */
32      public void printContent(boolean sorted) {
33          //Zuerst wird geprüft ob eine Datei eingelesen wurde
34          if (wordsSorted != null) {
35              ArrayList<String> tmpWords = new ArrayList<>(wordsSorted);
36              //Der erste eintrag wird in einen String geschrieben
37              String currentWord = tmpWords.get(0);
38              int countWords = 0;
39
40              for (int i = 0; i < tmpWords.size(); ++i) {
41                  /*
42                   * Falls das Wort übereinstimmt mit dem zwischengespeichertem word,
                         wird der
43                   * Counter um eins erhöht und der aktuelle eintrag gelöscht
44                   */
45                  if (caseSensitive ? currentWord.equals(tmpWords.get(i)) :
                        currentWord.equalsIgnoreCase(tmpWords.get(i))) {
46                      ++countWords;
47                      tmpWords.remove(i--);
48                  /*
49                   * Falls die Wörter unterschiedlich sind, wird das aktuelle Word
                         ausgegeben plus
50                   * deren Anzahl, anschlieSSend wird aktuelle word mit dem neuen
                         ersetzt und der
51                   * counter zurückgesetzt
52                   */
53                  } else {
54                      System.out.println(currentWord + " " + countWords + "x");
55                      currentWord = tmpWords.get(i);
56                      countWords = 1;
57                  }
58              }
59          } else
60              //Ausgabe, falls keine Datei eingelesen wurde
61              System.out.println(ANSI_RED + "Es wurde noch keine Datei Geladen!" +
                    ANSI_RESET);
62      }
63
64      /**
65       * Lädt einen Artikel von der Tagesschau herunter
66       * @param link Link von dem Artikel
67       */
68      public void loadContentFromWebsite(String link) {
69          ArrayList<String> webContent = null;
70          Scanner sc = null;
71
72          try {
73              System.out.println("Loading Content");
74              /*
75               * Baut eine verbindung zu Seite auf und lädt den Inhalt herunter
```

```
76              */
77              sc = new Scanner(new URL(link).openStream());
78              webContent = new ArrayList<>();
79              String line;
80
81              //Wenn inhalt gefunden wurde, wird die Schleife ausgeführt
82              while (sc.hasNextLine()) {
83                  /*
84                   * Jetzt gehen Zeile für Zeile durch den Inhalt.
85                   * Da wir uns nur für den Artikel interessieren suchen wir nach
                       drei verschiedenen HTML
86                   * Tags, einmal für dei beiden Überschriften und da wo sich der
                       Text befinden.
87                   * Wenn wir diese Zeile gefunden haben entfernen wir anschlieSSend
                       alle HTML
88                   * Tags aus dieser Zeile, sodass wir nur noch den Text über haben
89                   */
90                  line = sc.nextLine();
91                  if (line.contains(searchTextSmall)) {
92                      line = sc.nextLine();
93                      //System.out.println(line.stripLeading().replaceAll("&amp;",
                           "&").replaceAll("\\<.*?\\>\\h*", ""));
94                      webContent.add(line.stripLeading().replaceAll("&amp;", "&").
                           replaceAll("\\<.*?\\>\\h*", ""));
95                  }
96                  if (line.contains(searchHeadline) || line.contains(
                       searchSubtitleSmall)) {
97                      //System.out.println(line.stripLeading().replaceAll
                           ("\\<.*?\\>\\h*", ""));
98                      webContent.add(line.stripLeading().replaceAll("\\<.*?\\>\\h*",
                           ""));
99                  }
100             }
101             System.out.println("Loading done");
102         } catch (IOException e) {
103             System.out.println(ANSI_RED + "Es konnte keine Verbindung hergestellt
                    werden. Bitte überprüfen Sie die angegebene address, oder ihre
                    Internetverbindung!" + ANSI_RESET);
104         } finally {
105             if (sc != null)
106                 sc.close();
107         }
108
109         writeToFile(webContent);
110     }
111
112     /**
113      * Hier wird der Artikel in eine txt gespeichert
114      * @param content Text, der von der Webseite runtergeladen wurde
115      */
116     public void writeToFile(ArrayList<String> content) {
117         FileWriter out = null;
118
119         if (content != null && content.size() != 0)
120             try {
121                 //Setzt den Dateinamen fest
122                 out = new FileWriter(content.get(0).replaceAll("[^a-zA-ZäöüÄÖÜSS&%\\
                        h-]", "") + ".txt", Charset.forName("UTF-8"));
123
124                 //Fügt jede Zeile vom content hinzu und fügt ein Zeilenumbruch am
                        ende hinzu, zudem werden die Umlaute ersetzt
125                 for (String s : content)
126                     out.append(s.replace("ü", "ue")
```

```
                                    .replace("ö", "oe")
                                    .replace("ä", "ae")
                                    .replace("ß", "ss")
                                    .replaceAll("Ü(?=[a-zäöüß ])", "Ue")
                                    .replaceAll("Ö(?=[a-zäöüß ])", "Oe")
                                    .replaceAll("Ä(?=[a-zäöüß ])", "Ae")
                                    + "\n");
                System.out.println("Data written do file called: " + content.get(0)
                    .replaceAll("[^a-zA-ZäöüÄÖÜß%&\\h-]", "") + ".txt\n");
            } catch (IOException e) {
                System.out.println("Datei konnte nicht erstellt werden");
            } finally {
                if (out != null) {
                    try {
                        out.close();
                    } catch (IOException e) {
                        e.printStackTrace();
                    }
                }
            }
        else
            System.out.println(ANSI_RED + "Error, kein Inhalt gefunden!" +
                ANSI_RESET);
    }

    public void readFromFile(String filename) {
        //Lädt die Datei
        try (Scanner sc = new Scanner(new FileReader(filename + ".txt"))) {
            wordsUnsorted = new ArrayList<>();
            wordsSorted = new ArrayList<>();

            while (sc.hasNextLine()) {
                /*
                 * Läuft durch jede Zeile und entfernt alle unerwünschten Zeichen
                 * AnschlieSSen wird bei Jeden Leerzeichen ein split gemacht
                 */
                String[] line = sc.nextLine().replaceAll("ă", "").replaceAll("[^a-
                    zA-Z\\h-]", "").replaceAll("[\\h]-", " ").split(" ");
                for (String words : line)
                    /*
                     * Hier wird noch kurz geprüft der String befüllt ist
                     * AnschlieSSend wird er in einer Collection gespeichert
                     */
                    if (!words.equals("")) {
                        wordsUnsorted.add(words);
                        wordsSorted.add(words);
                    }
            }
            //Zum schluss wird die Collection noch Alphabetisch sortiert
            Collections.sort(wordsSorted);
            Collections.sort(wordsSorted, String.CASE_INSENSITIVE_ORDER);
            System.out.println("Datei wurde Erfolgreich geladen\n");
        } catch (IOException e) {
            System.out.println(ANSI_RED + "Error, es wurde noch keine Datei geladen
                !" + ANSI_RESET);
        }
    }
}
```

9.4.3 MyException.java

```
package chapter_10;
```

```
2
3  public class MyException extends Exception {
4
5      @Override
6      public String getMessage() {
7          return "Error, kein gültiger start Parameter, nur true oder false ist
               erlaubt!";
8      }
9
10
11 }
```

9.5 Testdokumentation

- Startparameter:

  Bei einem inkorrekten Startparameter wird eine Exception geworfen.

- Menüauswahl:

  Bei falscher Eingabe wurde eine entsprechende Fehlermeldung ausgegeben.

- Artikel herunterladen:

  Wenn der Link nicht existiert kommt eine Fehlermeldung.

- Datei Laden:

  Wenn keine Datei gefunden wird mit den Namen kommt eine Fehlermeldung.

- Wörter ausgeben:

  Wenn keine Datei geladen wurde, kommt eine entsprechende Fehlermeldung.

9.6 Benutzungshinweise

Bei dem Aufruf des Programmes muss mit true oder false angegeben werden, ob Groß- und Kleinschreibung zu beachten ist. Anschließend kann man in dem Menü auswählen ob man einen Artikel von der Tagesschau herunterladen möchte, eine Datei einlesen oder die Wörter in Alphabetischer Reihenfolge ausgibt.

9.7 Anwendungsbeispiel

```
1   [sebastian@laptop bin]$ java Main
2   Exception in thread "main" chapter_10.MyException: Error, nur true oder false
        erlaubt
3   at chapter_10.Main.main(Main.java:21)
4   [sebastian@laptop bin]$
5   [sebastian@laptop bin]$ java Main
6   Bitte wähle einer der Folgenden Optionen aus:
7   1) Einen Artikel von der Tagesschau herunterladen.
8   2) Eine Datei laden.
9   3) Wörter ausgeben.
10  0) Programm Beenden.
11  1
12  Bitte gebe den Link an:
13  https://www.tagesschau.de/inland/gesellschaft/johnsonjohnson-103.html
14  Loading Content
15  Loading done
16  Data written do file called: Warum punktet der neue Impfstoff.txt
17
18  Bitte wähle einer der Folgenden Optionen aus:
19  1) Einen Artikel von der Tagesschau herunterladen.
20  2) Eine Datei laden.
21  3) Wörter ausgeben.
```

```
  0) Programm Beenden.
  a
  Es sind nur Zahlen erlaubt
  b
  Es sind nur Zahlen erlaubt
  2
  Bitte geben Sie den Datei namen an:
  Warum punktet der neue Impfstoff
  Datei wurde Erfolgreich geladen

  Bitte wähle einer der Folgenden Optionen aus:
  1) Einen Artikel von der Tagesschau herunterladen.
  2) Eine Datei laden.
  3) Wörter ausgeben.
  0) Programm Beenden.
  3
  ab 4x
  aber 4x
  abgeschwaechten 1x
  aehnliche 1x
  aehnliches 1x
  aendern 1x
  Alle 2x
  allen 2x
  allerdings 1x
  Allergien 1x
  allergischen 1x
  als 2x
  also 1x
  alsoauch 1x
  Altersgruppe 1x
  am 1x
  amp 1x
  an 3x
  anderen 3x
  anhaelt 1x
  ansteckend 1x
  Antikoerper 2x
  Anwendung 1x
  Anzahl 1x
  Arzneimittel-Agentur 1x
  Arzneimittelhersteller 2x
  AstraZeneca 4x
  AstraZeneca-Impfstoff 1x
  Auch 3x
  auch 6x
  auf 2x
  aus 3x
  ausreichende 1x
  Ausserdem 1x
  basierte 1x
  bauen 1x
  Bauplan 2x
  Bei 8x
  bei 8x
  beide 1x
  beiden 1x
  bekommen 1x
  beobachteten 1x
  bereits 4x
  Berichte 1x
  Beschwerden 1x
  bildet 2x
```

```
  BioNTechPfizer 5x
  BioNTechPfizernbspbei 1x
  bisher 1x
  bisherigen 1x
  bleibt 1x
  Brasilien 2x
  britischen 1x
  Coronavirus 5x
  Coronavirus-Erbgutes 1x
  Dabei 1x
  dafuer 1x
  Damit 1x
  damit 3x
  Darauf 1x
  darauf 1x
  darueber 1x
  Das 4x
  das 5x
  dass 4x
  Datenlage 2x
  Dazu 1x
  dem 4x
  den 14x
  denen 1x
  denn 1x
  dennoch 1x
  Der 5x
  der 25x
  des 7x
  deuten 1x
  Deutschland 4x
  Die 3x
  die 23x
  dies 1x
  Diese 3x
  diese 1x
  direkt 1x
  Dort 1x
  dort 1x
  Dosis 1x
  drei 2x
  ebenfalls 1x
  ein 4x
  Ein-Mal-Dosis 1x
  eine 5x
  einen 3x
  einer 2x
  eines 1x
  eineSchutzwirkung 1x
  einfachen 1x
  eingeraeumt 1x
  Einschraenkung 1x
  Einstichstelle 1x
  Empfehlung 2x
  empfiehlt 1x
  empfindlicher 1x
  entscheidet 1x
  entsprechende 1x
  Entwicklung 1x
  er 1x
  Erbmaterials 1x
  Ergebnis 1x
  Ergebnisse 1x
```

```
  Ergebnissen 1x
  Erkrankungen 1x
  erreicht 1x
  Erste 1x
  ersten 2x
  es 5x
  etwas 1x
  EU-Arzneimittelbehoerde 1x
  Europaeische 1x
  Europageaendert 1x
  Exakte 1x
  Faellen 1x
  Fall 1x
  flaechendeckend 1x
  fuer 3x
  funktioniert 1x
  Gefrierschranktemperaturenstabil 1x
  gegeben 1x
  gegen 6x
  gegenueber 1x
  gehoeren 1x
  geimpft 6x
  Geimpfte 1x
  geimpfte 1x
  Geimpften 1x
  gelagert 1x
  Gelenkschmerzen 1x
  genau 1x
  generelle 1x
  genetische 2x
  geringer 1x
  geringeren 1x
  getestet 2x
  gibt 2x
  gleiche 1x
  Grad 2x
  groesseren 1x
  Grossbritannien 1x
  grosser 1x
  Grund 1x
  haeufigsten 1x
  haltbar 1x
  handelt 1x
  hat 7x
  hatteBioNTechbekannt 1x
  hatten 2x
  heisst 1x
  hier 1x
  hindass 1x
  hingegen 1x
  hoch 1x
  hohe 1x
  ihre 1x
  ihrer 1x
  im 2x
  Immunsystem 2x
  Impfdosis 1x
  impfen 1x
  Impfkommission 2x
  Impfstoff 11x
  Impfstoffe 4x
  Impfstoffen 3x
  Impfstoffs 1x
```

```
Impfung 4x
In 3x
in 20x
Infektionen 1x
Information 1x
Inhaltsstoffe 1x
Israel 1x
ist 11x
Jaehrige 1x
Jahren 4x
jeden 1x
Johnson 18x
kam 1x
keine 1x
klangen 1x
klinischen 2x
Koch-Institut 1x
koennen 2x
koennte 1x
Koerper 1x
kommenden 1x
konnten 1x
Kopf- 1x
Kuehlschranktemperaturen 1x
Laendern 1x
lag 2x
Lagerung 1x
lange 3x
lassen 1x
laufen 2x
laut 2x
liegen 1x
Mal 2x
manche 1x
mehrere 1x
mehreren 1x
meisten 1x
Menschen 6x
menschliche 1x
Minus 2x
Minusgraden 1x
mit 5x
mithilfe 1x
Moderna 6x
Monaten 1x
mRNA-Impfstoffen 2x
Muedigkeit 1x
muessen 1x
muss 1x
Mutation 1x
Mutationen 2x
mutierten 1x
nach 3x
nachgebaut 1x
Nachteile 1x
neben 1x
Nebenwirkungen 2x
neuartigen 1x
neue 2x
nicht 4x
niedriger 1x
noch 4x
noetig 1x
```

```
  normalen 2x
  Notfallzulassung 1x
  Nun 1x
  Nur 1x
  nur 4x
  Ob 1x
  ob 1x
  Oberflaechenproteine 1x
  ObTransport 1x
  oder 4x
  Personen 1x
  Praeparat 2x
  Praeparate 1x
  Praeparats 1x
  Prioritaet 1x
  Probanden 2x
  Prozent 6x
  pruefen 1x
  punktet 1x
  reagiert 2x
  Reaktionen 1x
  RKI 1x
  Robert 1x
  schaedlich 1x
  Schmerzen 1x
  schnell 1x
  Schuettelfrost 1x
  schuetzen 1x
  schuetzt 1x
  Schutz 1x
  Schutzwirkung 3x
  Schwere 1x
  schwere 1x
  schweren 1x
  sehr 1x
  sein 1x
  sich 6x
  Sicht 1x
  sie 2x
  sind 7x
  so 1x
  sogenannten 1x
  soll 1x
  sollten 1x
  sondern 1x
  Staendigen 1x
  StiKo 2x
  Studien 5x
  Studienvorliegen 1x
  Suedafrika 3x
  suedafrikanischen 1x
  Symptomen 1x
  tatsaechlich 1x
  Teil 1x
  Teile 2x
  Teils 1x
  Todesfaelle 1x
  Traegerviren 1x
  Traegervirus 2x
  transportiert 1x
  trotz 1x
  ueber 1x
  uebertragen 1x
```

```
  um 1x
  und 15x
  ungefaehrlich 1x
  Unklar 1x
  unklar 1x
  Unternehmen 1x
  unterschiedliche 1x
  untersucht 1x
  Untersuchungen 1x
  urspruengliche 1x
  USA 1x
  Variante 1x
  Varianten 1x
  Vektorimpfstoff 1x
  VektorimpfstoffenbspDas 1x
  verbessert 1x
  Vergleich 1x
  vergleichbar 1x
  verhindert 1x
  verimpft 2x
  Verlaeufe 1x
  verpackt 1x
  Vertraeglichkeit 1x
  verwenden 1x
  viele 1x
  Vier 1x
  vier 2x
  Viren 2x
  Virus 1x
  von 21x
  vor 2x
  Vor- 1x
  Vorteil 2x
  Vorteile 1x
  war 1x
  Warum 1x
  weitertragen 1x
  Welche 3x
  welche 1x
  Welcher 1x
  wenigen 1x
  weniger 2x
  wenn 2x
  werden 8x
  Wie 3x
  wie 6x
  wieder 1x
  wird 4x
  Wirken 1x
  wirksam 1x
  Wirksamkeit 4x
  Wirkstoffe 1x
  wirkt 1x
  Wirkungsweise 1x
  wo 1x
  Wochen 2x
  worden 2x
  wurde 3x
  Zahlen 1x
  zeigen 2x
  zeigte 1x
  Zelle 1x
  Zellen 1x
```

```
  Zu 1x
  zu 3x
  zudem 1x
  zufolge 2x
  zugelassen 1x
  zugelassenen 2x
  zunaechst 1x
  zurueckgenommenNun 1x
  Zusammenhang 1x
  zwei 1x
  zweiten 1x
  Bitte wähle einer der Folgenden Optionen aus:
  1) Einen Artikel von der Tagesschau herunterladen.
  2) Eine Datei laden.
  3) Wörter ausgeben.
  0) Programm Beenden.
  0
  [sebastian@laptop bin]$
```