5 Methodenaufrufe

5.1. Implementieren Sie die Struktogramme von Aufgabe 4.5 und 3.6 von Lsg_Struktogramm.pdf in einer Klasse und testen Sie das Programm.

```
 2 * A0501Schaltjahr.java
 5 package strProgAufg;
 6 import input.Eingabe;
 8 * @author stk
13 public class A0501Schaltjahr
14 {
15     /**
16      * @param args
17      * Kurzbeschreibung: Ermittle die Anzahl Schaltjahre im Bereich
18      * des eingelesenen Start- und Endjahres
19      */
20     public static void main(String[] args)
21     {
22         int iAnzahl = 0;
23         int iJahr;
24         int iStartJahr, iEndJahr;
25
26         iStartJahr = Eingabe.getInt("Startjahr = ");
27         iEndJahr   = Eingabe.getInt("Endjahr = ");
28
29         for (iJahr = iStartJahr; iJahr <= iEndJahr; iJahr++)
30         {
31             if (A0501Schaltjahr.istSchaltjahr(iJahr))
32             {
33                 iAnzahl++;
34             }
35         }
36         System.out.printf("Der Bereich enthält %d Schaltjahre\n", iAnzahl);
37     }
39     /**
40      * @param iJahr Jahr das geprüft werden soll
41      * @return true, wenn iJahr Schaltjahr, false sonst
42      * Kurzbeschreibung: Ermittle ob iJahr ein Schaltjahr ist
43      */
44     public static boolean istSchaltjahr(int iJahr)
45     {
46         boolean bRueckgabe = false;
47
48         if (iJahr % 4 == 0)
49         {
50             if (iJahr % 100 == 0)
51             {
52                 if (iJahr % 400 == 0)
53                 {
54                     bRueckgabe = true;
55                 }
56             }
57             else
58             {
59                 bRueckgabe = true;
60             }
61         }
62         return bRueckgabe;
63     }
64 }
```

5.2. In der Java Dokumentation (https://docs.oracle.com/) finden Sie eine Beschreibung der

Bibliotheksmethode Math.random(). Erkundigen Sie sich über die Funktionalität dieser Methode. Erstellen Sie eine Methode mit Namen `zZahl`, die folgenden Anforderungen genügt:
- Die Methode besitzt zwei ganzzahlige Parameter: iStartwert, iEndwert
- Die Methode liefert als Rückgabewert eine Zufallszahl z: iStartwert <= z <= iEndwert

Testen Sie Ihre Methode zZahl, indem Sie in der Main-Methode das Struktogramm von Aufgabe 4.9 von Lsg_Struktogramm.pdf implementieren.

```
11  */
12 public class A0502ZufallsZahlen
13 {
14     /**
15      * @param args
16      * Kurzbeschreibung:
17      */
18     public static void main(String[] args)
19     {
20         int iAnzahl6 = 0;
21         for (int n = 0; n < 500; n++)
22         {
23             if (zZahl(1, 6) == 6)
24             {
25                 iAnzahl6++;
26             }
27         }
28         System.out.printf("Häufigkeit der 6 absolut %d\n", iAnzahl6);
29         System.out.printf("Relative Häufigkeit der 6 %6.2f %%\n",
30                           iAnzahl6 / 500.0 * 100);
31     }
32
33     public static int zZahl(int iStartwert, int iEndwert)
34     {
35         double dZahl = Math.random(); // 0 <= dZahl < 1
36         // Transformation y = m x + b mit m = (iEndwert - iStartwert + 1) und b = iStartwert
37         return (int)((iEndwert - iStartwert + 1) * dZahl + iStartwert);
38         // alternativ: return (int)((dZahl *(iEndwert*10)) % (iEndwert-iStartwert + 1)) +iStartwert;
39     }
40 }
```

5.3. Implementieren Sie das Struktogramm von Aufgabe 4.7 von Lsg_Struktogramm.pdf und verwenden Sie Ihre bereits erstellte Methode zZahl von Aufgabe 5.2

5.4. **Programm Bruchrechnen:** Erstellt werden soll ein Programm, das zwei Brüche addieren, subtrahieren, multiplizieren und dividieren kann

a) Im Hauptprogramm (main) sollen der Zähler (iZ1) und Nenner (iN1) des ersten Bruchs, sowie der Zähler (iZ2) und Nenner (iN2) des zweiten Bruchs als ganze Zahlen eingegeben werden.
Anschließend soll der Benutzer die Möglichkiet erhalten auszuwählen, ob er die Brüche addieren, subtrahieren, multiplizieren oder dividieren möchte.

b) Erstellen Sie eine Klasse Bruchrechnen, die die Methoden `addieren()`, `subtrahieren()`, `multiplizieren()` und `dividieren()` enthält und definieren Sie diese Methoden. Alle Methoden sollen als Übergabe-Parameter die Zähler und Nenner beider Brüche erhalten und keinen Rückgabewert zurückgeben. In den Methoden soll der Zähler und Nenner des Ergebnisses berechnet und auf dem

Bildschirm ausgegeben werden!
Rufen Sie die Methoden an entsprechender Stelle im Hauptprogramm auf.

c) Ergänzen Sie die Klasse Bruchrechnen um die Methode `berechneGGT` (siehe Aufgabe 4.8 von Lsg_Struktogramm.pdf). Nutzen Sie diese Methode in den Methoden `addieren(), subtrahieren(), multiplizieren() und dividieren()` um den Ergebnisbruch vor der Ausgabe zu kürzen.

d) **Zusatzaufgabe**: Wenn eine Methode frei von Ein- und Ausgabe Anweisungen ist, dann kann sie flexibler angewendet werden. Deshalb soll das Programm so geändert werden, dass die Rechenmethoden das Ergebnis nicht mehr am Bildschirm anzeigen, sondern als String an den Aufrufer (hier main) zurückgeben. Zu diesem Zweck wird die Klasse Bruchrechnen um eine weitere Methode namens `ToString` ergänzt, die dann in den Rechenmethoden zur Erzeugung des Rückgabestring benutzt werden kann.
Die Methode besitzt folgenden Kopf (Signatur):

```
public static String ToString(int zaehler, int nenner)
```

Die Methode hat folgende Funktionalität:
- Bsp.: Ist der zaehler = 1 und der nenner = 2 dann ist die Rückgabe "1/2"
- Bsp.: Ist der zaehler = 3 und der nenner = 2 dann ist die Rückgabe "1 1/2"
- Bsp.: Ist der zaehler = 6 und der nenner = 2 dann ist die Rückgabe "3"

```java
/*
 * A0504Bruchrechnen.java
 */
package strProgAufg;
import input.Eingabe;
/**
 * @author stk Kurzbeschreibung: Programm zum Rechnen mit Brüchen
 *                               Programm ermöglicht das Rechnen mit Brüchen für
 *                               die Grundrechenarten
 */
public class A0504Bruchrechnen
{
    /**
     * @param args
     */
    public static void main(String[] args)
    {
        // Variablendeklaration:
        int z1, n1, z2, n2;
        char rechenart;

        // Eingabe:
        z1 = Eingabe.getInt("Bitte geben Sie den ersten Zähler ein: ");
        n1 = Eingabe.getInt("Bitte geben Sie den ersten Nenner ein: ");
        z2 = Eingabe.getInt("Bitte geben Sie den zweiten Zähler ein: ");
        n2 = Eingabe.getInt("Bitte geben Sie den zweiten Nenner ein: ");

        System.out.println("Wollen Sie Brüche addieren (+), subtrahieren (-) , "
                + "multiplizieren (*) oder dividieren (/)?  ");
        rechenart = Eingabe.getChar();

        System.out.printf("\n%d/%d %c %d/%d = ", z1, n1, rechenart, z2, n2);

        // Aufruf der verschiedenen Methoden in der Fallunterscheidung der Rechenart:
        switch (rechenart)
        {
        case '+':
            Bruchrechnen.addieren(z1, n1, z2, n2);
            break;
        case '-':
            Bruchrechnen.subtrahieren(z1, n1, z2, n2);
            break;
        case '*':
            Bruchrechnen.multiplizieren(z1, n1, z2, n2);
            break;
        case '/':
            Bruchrechnen.dividieren(z1, n1, z2, n2);
            break;
        default:
            System.out.println("falsche Eingabe");
            break;
        }
    }
}
```

```
59  class Bruchrechnen
60  {
61      public static int berechneGGT(int a, int b)
62      {
63          int rest;
64
65          do
66          {
67              rest = a % b;
68              a = b;
69              b = rest;
70          } while (rest != 0);
71
72          if (a < 0)
73              a = -a; // ggT generell positiv
74
75          return a; // a enthält den ggT
76      }
77
78      public static void addieren(int z1, int n1, int z2, int n2)
79      {
80          int ergebnis_n, ergebnis_z, ggT;
81
82          ergebnis_z = z1 * n2 + z2 * n1;
83          ergebnis_n = n1 * n2;
84
85          ggT = berechneGGT(ergebnis_z, ergebnis_n);
86          ergebnis_z = ergebnis_z / ggT;
87          ergebnis_n = ergebnis_n / ggT;
88
89          System.out.printf("%d/%d  Ergebnis der Addition", ergebnis_z, ergebnis_n);
90      }
91
92      public static void subtrahieren(int z1, int n1, int z2, int n2)
105
106     public static void multiplizieren(int z1, int n1, int z2, int n2)
119
120     public static void dividieren(int z1, int n1, int z2, int n2)
133
134 }
```

Zusatzaufgabe d):

```java
/* A0504BruchrechnenZusatz.java
package strProgAufg;

import input.Eingabe;

 * @author stk Kurzbeschreibung: Programm zum Rechnen mit Brüchen
public class A0504BruchrechnenZusatz
{
     * @param args
    public static void main(String[] args)
    {
        // Variablendeklaration:
        int z1, n1, z2, n2;
        char rechenart;

        // Eingabe:
        z1 = Eingabe.getInt("Bitte geben Sie den ersten Zähler ein: ");
        n1 = Eingabe.getInt("Bitte geben Sie den ersten Nenner ein: ");
        z2 = Eingabe.getInt("Bitte geben Sie den zweiten Zähler ein: ");
        n2 = Eingabe.getInt("Bitte geben Sie den zweiten Nenner ein: ");

        System.out.println(
                "Wollen Sie Brüche addieren (+), subtrahieren (-) , multiplizieren (*)
        rechenart = Eingabe.getChar();

        System.out.printf("\n%d/%d %c %d/%d = ", z1, n1, rechenart, z2, n2);

        // Aufruf der verschiedenen Methoden in der Fallunterscheidung der Rechenart:
        switch (rechenart)
        {
        case '+':
            System.out.println(Bruchrechnen2.addieren(z1, n1, z2, n2));
            break;
        case '-':
            System.out.println(Bruchrechnen2.subtrahieren(z1, n1, z2, n2));
            break;
        case '*':
            System.out.println(Bruchrechnen2.multiplizieren(z1, n1, z2, n2));
            break;
        case '/':
            System.out.println(Bruchrechnen2.dividieren(z1, n1, z2, n2));
            break;
        default:
            System.out.println("falsche Eingabe");
            break;
        }
    }

}
```

```
60  class Bruchrechnen2
61  {
62      public static int berechneGGT(int a, int b)
75
76      public static String ToString(int zaehler, int nenner)
77      {
78          String erg;
79
80          if (zaehler % nenner == 0)
81              erg = String.format("%d", zaehler / nenner); // Kein Bruch, sondern ganze Zahl
82          else
83              if (Math.abs(zaehler) > Math.abs(nenner)) // Bruch enthält eine ganze Zahl
84                  erg = String.format("%d %d/%d", zaehler / nenner,
85                          Math.abs(zaehler % nenner), Math.abs(nenner));
86              else
87                  erg = String.format("%d/%d", zaehler, nenner); // echter Bruch
88
89          return erg;
90      }
91
92      public static String addieren(int z1, int n1, int z2, int n2)
93      {
94          int ergebnis_n, ergebnis_z, ggT;
95
96          ergebnis_z = z1 * n2 + z2 * n1;
97          ergebnis_n = n1 * n2;
98
99          ggT = berechneGGT(ergebnis_z, ergebnis_n);
100         ergebnis_z = ergebnis_z / ggT;
101         ergebnis_n = ergebnis_n / ggT;
102
103         return ToString(ergebnis_z, ergebnis_n);
104     }
105
106     public static String subtrahieren(int z1, int n1, int z2, int n2)
119
120     public static String multiplizieren(int z1, int n1, int z2, int n2)
133
134     public static String dividieren(int z1, int n1, int z2, int n2)
147 }
```

5.5. Erstellen Sie ein Programm, dass die Unicode-Tabelle im Bereich der darstellbaren Zeichen ab 0x20 bis 0xFF am Bildschirm wie folgt anzeigt:

```
Start des Programms Unicode Tabelle

   |  0|  1|  2|  3|  4|  5|  6|  7|  8|  9|  A|  B|  C|  D|  E|  F|
-------------------------------------------------------------------------
 20|   |  !|  "|  #|  $|  %|  &|  '|  (|  )|  *|  +|  ,|  -|  .|  /|
 30|  0|  1|  2|  3|  4|  5|  6|  7|  8|  9|  :|  ;|  <|  =|  >|  ?|
 40|  @|  A|  B|  C|  D|  E|  F|  G|  H|  I|  J|  K|  L|  M|  N|  O|
 50|  P|  Q|  R|  S|  T|  U|  V|  W|  X|  Y|  Z|  [|  \|  ]|  ^|  _|
 60|  `|  a|  b|  c|  d|  e|  f|  g|  h|  i|  j|  k|  l|  m|  n|  o|
 70|  p|  q|  r|  s|  t|  u|  v|  w|  x|  y|  z|  {|  ||  }|  ~|  ⌂|
 80|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|
 90|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|  ?|
 A0|   |  ¡|  ¢|  £|  ¤|  ¥|  ¦|  §|  ¨|  ©|  ª|  «|  ¬|  -|  ®|  ¯|
 B0|  °|  ±|  ²|  ³|  ´|  µ|  ¶|  ·|  ¸|  ¹|  º|  »|  ¼|  ½|  ¾|  ¿|
 C0|  À|  Á|  Â|  Ã|  Ä|  Å|  Æ|  Ç|  È|  É|  Ê|  Ë|  Ì|  Í|  Î|  Ï|
 D0|  Ð|  Ñ|  Ò|  Ó|  Ô|  Õ|  Ö|  ×|  Ø|  Ù|  Ú|  Û|  Ü|  Ý|  Þ|  ß|
 E0|  à|  á|  â|  ã|  ä|  å|  æ|  ç|  è|  é|  ê|  ë|  ì|  í|  î|  ï|
 F0|  ð|  ñ|  ò|  ó|  ô|  õ|  ö|  ÷|  ø|  ù|  ú|  û|  ü|  ý|  þ|  ÿ|
```

Zusatzaufgabe: Ändern Sie das Programm so ab, dass man durch Anpassen jeweils einer symbolischen Konstanten (final) die Spaltenzahl und die Spaltenbreite einfach verändern kann. Beispiel: Anzahl Spalten auf 8 und Spaltenbreite auf 6 geändert.

```
Start des Programms Unicode Tabelle

      |     0|     1|     2|     3|     4|     5|     6|     7|
-------------------------------------------------------------------
    20|      |     !|     "|     #|     $|     %|     &|     '|
    28|     (|     )|     *|     +|     ,|     -|     .|     /|
    30|     0|     1|     2|     3|     4|     5|     6|     7|
    38|     8|     9|     :|     ;|     <|     =|     >|     ?|
    40|     @|     A|     B|     C|     D|     E|     F|     G|
    48|     H|     I|     J|     K|     L|     M|     N|     O|
    50|     P|     Q|     R|     S|     T|     U|     V|     W|
    58|     X|     Y|     Z|     [|     \|     ]|     ^|     _|
    60|     `|     a|     b|     c|     d|     e|     f|     g|
    68|     h|     i|     j|     k|     l|     m|     n|     o|
    70|     p|     q|     r|     s|     t|     u|     v|     w|
    78|     x|     y|     z|     {|     ||     }|     ~|      |
    80|     ?|     ?|     ?|     ?|     ?|     ?|     ?|     ?|
    88|     ?|     ?|     ?|     ?|     ?|     ?|     ?|     ?|
    90|     ?|     ?|     ?|     ?|     ?|     ?|     ?|     ?|
    98|     ?|     ?|     ?|     ?|     ?|     ?|     ?|     ?|
    A0|      |     ¡|     ¢|     £|     ¤|     ¥|     ¦|     §|
    A8|     ¨|     ©|     ª|     «|     ¬|     -|     ®|     ¯|
    B0|     °|     ±|     ²|     ³|     ´|     µ|     ¶|     ·|
    B8|     ¸|     ¹|     º|     »|     ¼|     ½|     ¾|     ¿|
    C0|     À|     Á|     Â|     Ã|     Ä|     Å|     Æ|     Ç|
    C8|     È|     É|     Ê|     Ë|     Ì|     Í|     Î|     Ï|
    D0|     Ð|     Ñ|     Ò|     Ó|     Ô|     Õ|     Ö|     ×|
    D8|     Ø|     Ù|     Ú|     Û|     Ü|     Ý|     Þ|     ß|
    E0|     à|     á|     â|     ã|     ä|     å|     æ|     ç|
    E8|     è|     é|     ê|     ë|     ì|     í|     î|     ï|
    F0|     ð|     ñ|     ò|     ó|     ô|     õ|     ö|     ÷|
    F8|     ø|     ù|     ú|     û|     ü|     ý|     þ|     ÿ|
```

```java
public class A0505UniCodeTab
{

    /**
     * @param args
     */
    public static void main(String[] args)
    {
        final int WEITE = 4; // Breite des Ausgabebereichs für ein Zeichen in der Tabelle
        final int SPALTEN = 16;
        //---------------------------------------------------------------------------
        int i;              // Zählervariablen
        int iUmbruch = 0x20 % SPALTEN;
        String format1;
        String format2;

        format1 = "%" + WEITE + "H|";
        format2 = "%" + WEITE + "c|";

        System.out.println("Start des Programms Unicode Tabelle\n");

        System.out.printf(format2,' ');     // Tabellenkopf ausgeben
        for (i = 0; i <= SPALTEN - 1; i++)
            System.out.printf(format1, i);

        System.out.println();
        for (i = 1; i <= WEITE; i++) System.out.printf("-");
        for (i = 1; i <= SPALTEN * (WEITE + 1); i++) System.out.printf("-");

        for ( i=0x20; i<=0xff; i++)     // Tabelle ausgeben
        {

            if (i % SPALTEN == iUmbruch)
            {                                   // Zeilenvorschub und
                System.out.printf("\n" + format1, i); // Zeilenkopf ausgeben
            }
            System.out.printf(format2, (char)i);   // Unicode-Zeichen ausgeben
        }
        System.out.printf("\nEnde des Programms.");
    }
}
```