O.Schwahlen
it.schule stuttgart
Breitwiesenstrasse 20-22
70565 Stuttgart
schwahlen@its-stuttgart.de

CalcResi

Entwickeln Sie ein Programm, das nach der Eingabe der aufgedruckten Farben eines Widerstands, den entsprechenden Nennwert (inkl. Toleranz) laut IEC-Farbkodierung auf dem Monitor aus gibt. Im ersten Schritt sollen nur Widerstände mit vier Farbringen betrachtet werden.

Kenn-farbe	1. Ring (Wert der 1. Ziffer)	2. Ring (Wert der 2. Ziffer)	3. Ring Multi-plikator	4. Ring Toleranz
keine	–	–	–	± 20 %
silber	–	–	10^{-2}	± 10 %
gold	–	–	10^{-1}	± 5 %
schwarz	0	0	10^{0}	–
braun	1	1	10^{1}	± 1 %
rot	2	2	10^{2}	± 2 %
orange	3	3	10^{3}	–
gelb	4	4	10^{4}	–
grün	5	5	10^{5}	± 0,5 %
blau	6	6	10^{6}	–
violett	7	7	10^{7}	–
grau	8	8	10^{8}	–
weiß	9	9	10^{9}	–

```
CalcResi ═══════════════════════════════════════
═══ das Programm zur Decodierung des IEC-Farbcodes

Farbcodes des 1. und 2. Rings:
──────────────────────────────

0)schwarz 1)braun   2)rot     3)orange  4)gelb
5)grün    6)blau    7)violett 8)grau    9)weiß

Bitte geben Sie nun den Farbcode des 1.Rings ein: 3
Bitte geben Sie nun den Farbcode des 2.Rings ein: 4

Farbcodes des 3.Rings:
──────────────────────

1)silber   2)gold      3)schwarz  4)braun
5)rot      6)orange    7)gelb     8)grün
9)blau    10)violett  11)grau    12)weiß

Bitte geben Sie nun den Farbcode des 3.Rings ein: 5

Farbcodes des 4.Rings:
──────────────────────

1)keine  2)silber  3)gold
4)braun  5)rot     6)grün

Bitte geben Sie nun den Farbcode des 4.Rings ein: 6

═══════════════════════════════════════════════════
Ihr Widerstand hat den Nennwert: 3400 Ohm +-0.5%
```

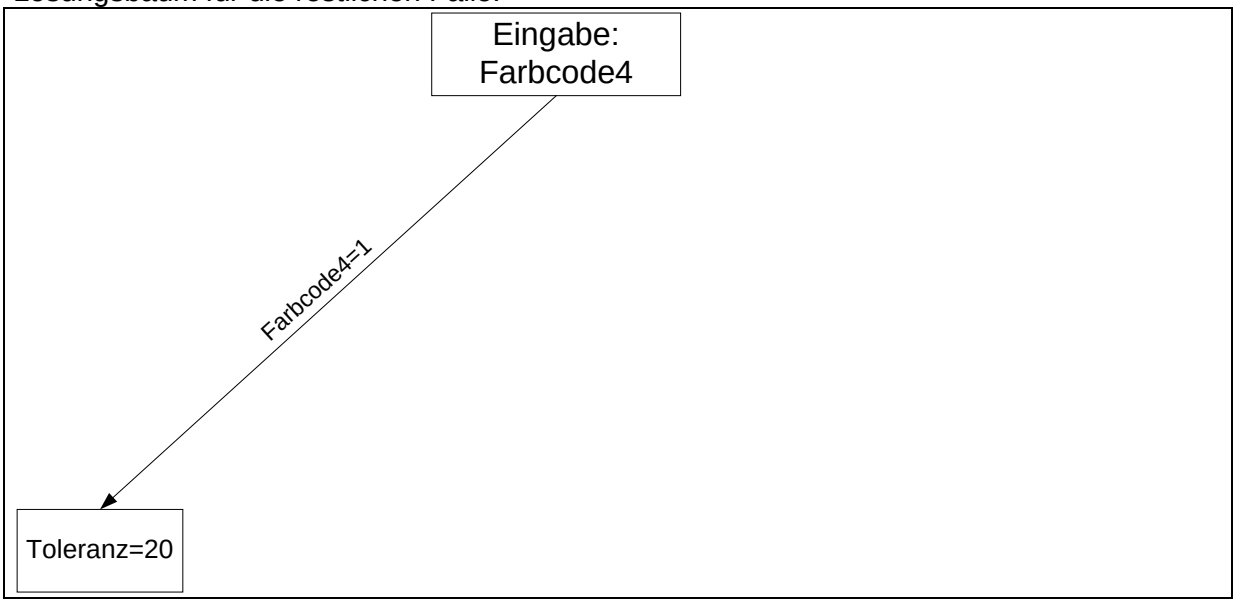
Ein mögliches Screendesign ist oben dargestellt. Die Farbkodierung entnehmen Sie dem Bild links.
Beispiel:

Ring	1.Ring	2.Ring	3.Ring	4.Ring
Farbe	orange	gelb	rot	grün
Eingabe	3	4	5	6
Wert	3	4	10^2	0,5

Somit hat der Widerstand einen Nennwert von 3400 Ω ± 0,5%.

1. Lösungsbaum

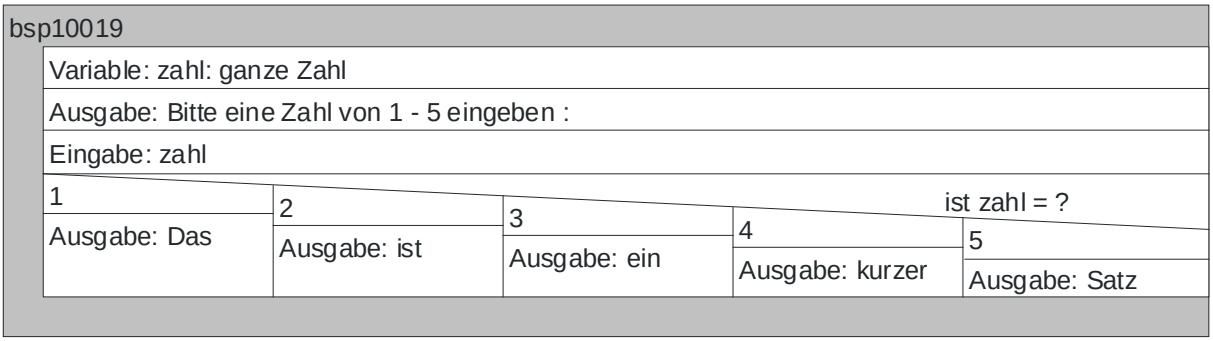
Zeichnen Sie einen Lösungsbaum für die Farbdekodierung des 4.Rings. Der Lösungsweg für den Fall, dass der 4. Farbring keine Farbe hat ist bereits eingezeichnet. Vervollständigen Sie den Lösungsbaum für die restlichen Fälle.



O.Schwahlen
it.schule stuttgart
Breitwiesenstrasse 20-22
70565 Stuttgart
schwahlen@its-stuttgart.de

Ergänzter Auszug aus:
Erlenkötter, Helmut: C++; Objektorientiertes Programmieren von Anfang an. Reinbek bei Hamburg, 2000

Bei der Vorstellung der if-Anweisung wurde erwähnt, dass es nicht sinnvoll ist, das if zu stark zu verschachteln, da die Konstruktionen unübersichtlich werden. Als Ersatz bietet sich die Anweisung *switch* an. Die Hauptaufgabe eines *switch* ist es somit, den Programmfluss übersichtlicher zu gestalten. Ein Beispiel sehen Sie im folgenden:




```
// Beispiel
public static void main(String[] args) {
{
  int zahl;
  System.out.println("\nBitte eine Zahl von 1 - 5 eingeben : ");
  BufferedReader buffRead = new BufferedReader(new InputStreamReader(System.in));
  try {
    String zeile = buffRead.readLine();
  } catch (IOException ex) {
    ex.printStackTrace();
  }
  zahl = Integer.parseInt(zeile);

  switch(zahl) {
    case 1:
      System.out.println ("Das ");
    case 2:
      System.out.println ("ist ");
    case 3:
      System.out.println ("ein ");
    case 4:
      System.out.println ("kurzer ");
    case 5:
      System.out.println ("Satz");
  }
}
```

Hinter *switch* folgt in Klammern ein Integerwert. Dieser Wert wird nun im folgenden, durch geschweifte Klammern begrenzten Anweisungsblock hinter einer Anweisung *case* gesucht. Das Programm «hangelt» sich an den *case*-Anweisungen entlang, bis es die richtige findet. Dann läuft das Programm an dieser Stelle normal weiter. Wie Sie am folgenden Programmlauf erkennen können, setzt das Programm seinen Lauf ab *case* 2: fort und läuft von da ab ohne Unterbrechung weiter.

```
Bitte eine Zahl von 1 - 5 eingeben :2
ist ein kurzer Satz
```

Diese Eigenschaft unterscheidet Java (auch C++) von den meisten anderen Sprachen, in denen es üblich ist, dass hinter einem *case* nur die Befehle bis zum nächsten *case* ausgeführt werden und das Programm dann hinter dem gesamten *case*-Block weitermacht. Das können wir natürlich auch in Java (bzw. C++) realisieren, wie Sie im nächsten Programm sehen können.

O.Schwahlen
it.schule stuttgart
Breitwiesenstrasse 20-22
70565 Stuttgart
schwahlen@its-stuttgart.de

bsp10020

Variable: zeichen: ein Zeichen

Ausgabe: Bitte geben Sie einen Buchstaben ein :

Eingabe: zeichen

ist zeichen = ?

a oder A	b oder B	c oder C	default
Ausgabe: Buchstabe A	Ausgabe: Buchstabe B	Ausgabe: Buchstabe C	Ausgabe: Kenn ich nicht!

```
// bsp10020.cpp
public static void main(String[] args) {
{
  char zeichen;
  System.out.println("\nBitte geben Sie einen Buchstaben ein : ");
  BufferedReader buffRead = new BufferedReader(new InputStreamReader(System.in));
  try {
    String zeile = buffRead.readLine();
  } catch (IOException ex) {
    ex.printStackTrace();
  }
  switch(zeile.charAt(0))
  {
    case 'A':
    case 'a':
      System.out.println ("Buchstabe A");
      break;
    case 'B':
    case 'b':
      System.out.println ("Buchstabe B");
      break;
    case 'C':
    case 'c':
      System.out.println ("Buchstabe C");
      break;
    default:
      System.out.println ("Kenn ich nicht!");
  }
}
```

Die nächste Abbildung zeigt vier Programmdurchläufe mit ihren Ausgaben auf dem Bildschirm.

```
Bitte geben Sie einen Buchstaben ein :a
Buchstabe A
```

```
Bitte geben Sie einen Buchstaben ein :b
Buchstabe B
```

```
Bitte geben Sie einen Buchstaben ein :c
Buchstabe C
```

```
Bitte geben Sie einen Buchstaben ein :k
Kenn ich nicht!
```

In diesem Programm haben Sie eine weitere Anweisung kennen gelernt:
die Break-Anweisung. Sie dient dazu, aus einem *switch* - wie hier gezeigt - oder aus einer *for*-, *do*- oder *while*-Schleife zu springen. Den weitaus häufigsten Einsatz findet *break* jedoch beim *switch*. Es verzweigt hinter die Schleife, in der es steht, bzw. hinter den *switch*-Block.

Hinweis: Vergessen Sie das *break* in einer *switch*-Anweisung nicht!

Neben dem Einsatz von *break* demonstriert das Programm, dass *switch* nicht nur mit *int*-, sondern auch mit *char*-Typen arbeiten kann. Als Regel gilt:
Hinter *switch* darf nur ein integraler Ausdruck (*char*, *int*, *short long*) stehen und hinter *case* nur eine typgleiche Konstante. Vollständige logische Ausdrücke, wie es in einigen anderen Sprachen möglich ist, sind nicht erlaubt.