Einfache Ein- und Ausgabe mit Java

1. Hallo-Welt !

Das erste Programm soll einen Text zum Bildschirm schicken. Es kann mit jedem beliebigen Texteditor erstellt werden.

```
/**
Die Klasse hello sendet einen Text :um Bildschirm.
*/

/* Datei : hello.java */

public class hello
{
    // Standardkopf
      public static void main(String[] args)
    {
        // Ausgabe
        System.out.println("Hello, World!");
    }
}
```

Der Name der Klasse (nach der Angabe class) muß dem Dateinamen entsprechen, weiterhin endet der Dateiname mit ●***java***. Das Programm wird mit ***javac hello.java*** kompiliert. Es entsteht die Datei ***hello.class*** mit dem Bytecode. Diese kann nun mit ***java hello*** abgearbeitet werden.

Programmlauf

```
Hello, World!
```

2. Elementare Datentypen

Elementarer	Größe in Bit	Minimum	Maximum
BOOLEAN	1	—	—
CHAR	16	*Unicode : \u0000*	Unicode: \uFFFF
BYTE	8	-128	127
SHORT	16	-32768	32767
INT	32	-2147483648	2147483647
LONG	64	-9223372036854775808	9223372036854775807
FLOAT	32	-3,40282347E+38	+3,40282347E+38
DOUBLE	64	-1,7976931348623157E+308	+1,7976931348623157E+308

Tabelle: Elementare Datentypen in Java

A. Harder FHL

3. Bildschirmausgabe

Für eine Bildschirmausgabe bietet Java die Methode ***System.out.println()*** an:

```
/*Meldung auf der Systemkonsole
Bildschirmausgabe
bildaus.java
*/

public class bildaus
{
  // Standardkopf
  public static void main (String[] args)
  {
      // Ausgabe
      System.out.println("Datenbank") ;
  }
}
```

Listing: bildaus.java

Der Befehl

```
System.out.println ("Datenbank") ;
```

führt zur Ausgabe des Textes *Datenbank* auf dem Bildschirm. Das Programm muß kompiliert werden und kann dann

gestartet werden. Das Programm gibt die Meldung

```
Datenbank
```

auf dem Bildschirm aus.

4. Ein-/Ausgabe

4.1 . Programm mit einer Ganzzahl (Integer) als Startparameter

(gekürzt)

4.2 Programm mit einer Dezimalzahl (Double) als Startparameter

(gekürzt)

5. Arithmetik

Das folgende Programm ***arithmetik.java*** demonstriert die Anwendung der arithmetischen Operatoren:

```
/**
Arithmetik in Java
Vereinbarungen und Arithmetik
arithmetik.java
*/

import java.io.*;
import java.lang.*;

public class arithmetik
{
 public static void main(String[] args)

 {
   int Zahl1 = 5;
   int Zahl2 = 3;
  int Ergebnis;

  double Zahl3 = 5.7;
  double Zahl4 = 3.8;
  double Resultat;

  System.out.println("Zahl1    : "+Zahl1);
  System.out.println("Zahl2    : "+Zahl2);
  Ergebnis = Zahl1 + Zahl2;
  System.out.println("Summe    : "+Ergebnis);
  Ergebnis = Zahl1 - Zahl2;
  System.out.println("Differenz: "+Ergebnis);
  Ergebnis = Zahl1 * Zahl2;
  System.out.println("Produkt  : "+Ergebnis);

  System.out.println("Zahl3    : "+Zahl3);
  System.out.println("Zahl4    : "+Zahl4);
  Resultat = Zahl3 + Zahl4;
  System.out.println("Summe    : "+Resultat);
  Resultat = Zahl3 - Zahl4;
  System.out.println("Differenz: "+Resultat);
  Resultat = Zahl3 * Zahl4;
  System.out.println("Produkt  : "+Resultat);
  Resultat = Zahl3 / Zahl4;
  System.out.println("Quotient : "+Resultat);
   }
}
```

Listing: arithmetik.java

6. Der Modulo-Operator

Häufig ist es interessant, den Rest einer Division zu ermitteln. So ist z.B.

25 / 7 = 3 Rest 4 (d.h. 25 = 3 * 7 + 4)

oder auch

36 / 10 = 3 Rest 6 (d.h. 36 = 3 * 10 + 6)

Ist der Rest gleich 0, liegt Teilbarkeit vor. Mathematisch wird diese Berechnungsverfahren durch ***mod*** formalisiert, also

25 mod 7 = 4
36 mod 10 = 6
36 mod 9 = 0

In Java steht als Modulo-Operator das % zur Verfügung. Der Ausdruck

25 % 7 hat den Wert 4 und
36 % 10 den Wert 6.

7. Ein-/Ausgabe

Das folgende Programm ***Eingabe.java*** demonstriert die Eingabe einer Zeichenkette:

```
import java.io.* ;
class Eingabe
{
  public static void main (String args[])
  {
BufferedReader In = new BufferedReader (new InputStreamReader (System.in)) ;
      String expr = new String() ;

      try
      {
            System.out.print ("Eingabe: ") ;
             expr = In.readLine() ;
             System.out.println("Ergebnis :   " + expr)
      }
      catch   ( lOException  e)
      {
          System.out.println("Problem mit  der  Eingabe!");
          System. exit(-1) ;
      }
  } // Ende  main()
} // Ende class   Eingabe
```

Listing: Eingabe.java

Ein paar Bemerkungen: Mit der Anweisung

```
import java.io.*
```

wird der Zugriff auf das Paket (package) ***java.io.**** ermöglicht. Der Stern deutet an, daß alle Klassen des Paketes importiert werden sollen. Es folgt die Klassenvereinbarung mit

class Eingabe { }

Der Klassenname muß gleich dem Programmnamen - Endung ***.java*** sein. Also: ***Eingabe*** als Klassenname für das Programm ***Eingabe.java***.

Innerhalb der Klasse Eingabe wird das Programm ***main ()*** vereinbart.

Vorabinfo: Durch die Angabe ***public*** *ist ein Zugriff von außen auf die Funktion* ***main()*** *möglich.* ***static*** *beinhaltet, daß die Funktion zu einer Klasse gehört und nicht zu einem Objekt. Mit void wird angegeben, daß die Funktion* ***main()*** *keinen Rückgabewert liefert. Mit String* ***args[]*** *können dem Programm beim Aufruf Werte übergeben werden. Diese Technik wird hier nicht genutzt, trotzdem müssen die entsprechenden Angaben gemacht werden.*

Wie folgt wird das Einlesen von Daten über den Systemeingabestrom ermöglicht:

BufferedReader In = new BufferedReader(new InputStreamReader (System.in)) ;

Anschließende Zeile schafft die Möglichkeit, einen String abzulegen. String

expr = new String();

Der Block mit ***try {}*** soll nun von Java abgearbeitet werden. Falls das fehlschlägt, wird die Ausnahmeroutine

catch (lOException e) {}

abgearbeitet. Für die Ein- und Ausgabe im Block ***try{}*** sind

System.out.print ("Eingabe : ") ;

Expr =. In.readLine() ;

System.println("Ergebnis: " + expr);

verantwortlich.