2 Eine einfache Programmiersprache

Eine Programmiersprache soll

- Datenstrukturen anbieten
- Operationen auf Daten erlauben
- Kontrollstrukturen zur Ablaufsteuerung bereitstellen

Als Beispiel betrachten wir MiniJava.

Variablen

Variablen dienen zur Speicherung von Daten.

Um Variablen in MiniJava zu nutzen müssen sie zunächst eingeführt, d.h. deklariert werden.

Variablen

Beispiel:

```
int x, result;
```

Diese Deklaration führt die beiden Variablen mit den Namen `x` und `result` ein.

- Das Schlüsselwort `int` besagt, dass diese Variablen ganze Zahlen („Integers“) speichern sollen.

  `int` heißt auch Typ der Variablen `x` und `result`.
- Variablen können dann benutzt werden, um anzugeben, auf welche Daten Operationen angewendet werden sollen.
- Die Variablen in der Aufzählung sind durch Kommas „,“ getrennt.
- Am Ende steht ein Semikolon „;“.

Operationen – Zuweisung

Operationen gestatten es, Werte von Variablen zu ändern. Die wichtigste Operation ist die Zuweisung.

Beispiele:

- `x = 7;`
  Die Variable `x` erhält den Wert `7`.
- `result = x;`
  Der Wert der Variablen `x` wird ermittelt und der Variablen `result` zugewiesen.
- `result = x + 19;`
  Der Wert der Variablen `x` wird ermittelt, `19` dazu gezählt und dann das Ergebnis der Variablen `result` zugewiesen.

Operationen – Zuweisung

Achtung:

- Java bezeichnet die Zuweisung mit „=“ anstatt „:=“ (Erbschaft von C...)
- Eine Zuweisung wird mit „;“ beendet.
- In der Zuweisung `x = x + 1;` greift das `x` auf der rechten Seite auf den Wert vor der Zuweisung zu.

Operationen – Input/Output

MiniJava enthält Operationen um Daten (Zahlen) einlesen bzw. ausgeben zu können.

Beispiele:

- `x = read();`
  Liest eine Folge von Zeichen ein und interpretiert sie als ganze Zahl, deren Wert sie der Variablen `x` als Wert zuweist.
- `write(42);`
  Schreibt `42` auf die Ausgabe.
- `write(result);`
  Bestimmt den Wert der Variablen `result` und schreibt dann diesen auf die Ausgabe.
- `write(x-14);`
  Bestimmt den Wert der Variablen `x`, subtrahiert `14` und schreibt das Ergebnis auf die Ausgabe.

Operationen – Input/Output

Achtung:

- Das argument der `write`-Operation in den Beispielen ist ein `int`.
- Um es ausgeben zu können muss es erst in eine Zeichenfolge umgewandelt werden, d.h. einen `String`

In MiniJava können auch direkt Strings ausgegeben werden:

Beispiel:

- `write("Hello World!!!");`
  Schreibt `Hello World!!!` auf die Ausgabe.

Kontrollstrukturen – Sequenz

Sequenz:

```
1 int x, y, result;
2 x = read();
3 y = read();
4 result = x + y;
5 write(result);
```

- Zu jedem Zeitpunkt wird nur eine Operation ausgeführt.
- Jede Operation wird genau einmal ausgeführt.
- Die Reihenfolge, in der die Operationen ausgeführt werden, ist die gleiche, in der sie im Programm stehen.
- Mit Beendigung der letzten Operation endet die Programm-Ausführung.

Sequenz alleine erlaubt nur sehr einfache Programme.

Kontrollstrukturen – Selektion

Selektion (bedingte Auswahl):

```
1 int x, y, result;
2 x = read();
3 y = read();
4 if (x > y)
5     result = x - y;
6 else
7     result = y - x;
8 write(result);
```

- Zuerst wird die Bedingung ausgewertet
- Ist sie erfüllt, wird die nächste Operation ausgeführt.
- Ist sie nicht erfüllt, wird die nächste Operation nach dem `else`-Zweig ausgeführt.

Kontrollstrukturen – Selektion

Beispiel:

- Statt einer einzelnen Operation können die Alternativen auch aus Statements bestehen:

```
1 int x;
2 x = read();
3 if (x == 0)
4     write(0);
5 else if (x < 0)
6     write(-1);
7 else
8     write(+1);
```


Kontrollstrukturen – Selektion

Beispiel:

- . . . oder aus (geklammerten) Folgen von Operationen und Statements:

```
1 int x, y;
2 x = read();
3 if (x != 0) {
4     y = read();
5     if (x > y)
6         write(x);
7     else
8         write(y);
9 } else
10    write(0);
```


Kontrollstrukturen – Selektion

Beispiel:

- . . . eventuell fehlt auch der `else`-Teil:

```
1 int x, y;
2 x = read();
3 if (x != 0) {
4     y = read();
5     if (x > y)
6         write(x);
7     else
8         write(y);
9 }
```

Auch mit Sequenz und Selektion kann noch nicht viel berechnet werden. . .

Kontrollstrukturen – Iteration

Iteration (wiederholte Ausführung)

```
1 int x, y;
2 x = read(); y = read();
3 while (x != y) {
4     if (x < y)
5         y = y - x;
6     else
7         x = x - y;
8 }
9 write(x);
```

Das Programm erfüllt die Spezifikation, dass es für $x, y \in \mathbb{N} \setminus \{0\}$, den GGT berechnet. Bei falscher Eingabe terminiert es eventuell nicht.
Im allgemeinen sollte man Eingaben vom Benutzer immer auf Zulässigkeit prüfen.

- Zuerst wird die Bedingung ausgewertet.
- Ist sie erfüllt, wird der Rumpf des `while`-statements ausgeführt.
- Nach Ausführung des Rumpfs wird das gesamte `while`-statement erneut ausgeführt.
- Ist die Bedingung nicht erfüllt fährt die Programmausführung hinter dem `while`-statement fort.

2 Eine einfache Programmiersprache

Eine Sprache mit dieser Eigenschaft nennt man auch turingvollständig.

Theorem (↑Berechenbarkeitstheorie)
Jede (partielle) Funktion auf ganzen Zahlen, die überhaupt berechenbar ist, läßt sich mit Selektion, Sequenz, und Iteration, d.h., mithilfe eines MiniJava-Programms berechnen.

Beweisidee

- Was heißt berechenbar?
  Eine Funktion heißt berechenbar wenn man sie mithilfe einer Turingmaschine berechnen kann.
- Schreibe ein MiniJava-Programm, das eine Turingmaschine simuliert.

Für dieses Theorem ist es wichtig, dass die `int`-Variablen von MiniJava beliebige ganze Zahlen speichern können. Sobald man dies einschränkt (z.B. 32 Bit) ist die entstehende Sprache immer noch sehr mächtig aber streng formal nicht mehr turingvollständig.

2 Eine einfache Programmiersprache

MiniJava-Programme sind ausführbares Java. Man muss sie nur geeignet dekorieren.

Beispiel: das GGT-Programm.

```
1  int x, y;
2  x = read();
3  y = read();
4  while (x != y) {
5      if (x < y)
6          y = y - x;
7      else
8          x = x - y;
9  }
10 write(x);
```


Ein Java-Programm

```
1  public class GGT extends MiniJava {
2      public static void main (String[] args) {
3          int x, y;
4          x = read();
5          y = read();
6          while (x != y) {
7              if (x < y)
8                  y = y - x;
9              else
10                 x = x - y;
11         }
12         write(x);
13     } // Ende der Definition von main();
14 } // Ende der Definition der Klasse GGT;
```

Datei "GGT.java"

Ein Java-Programm

Erläuterungen:

- Jedes Programm hat einen Namen (hier GGT)
- Der Name steht hinter dem Schlüsselwort `class` (was eine Klasse ist, was `public` ist lernen wir später)
- Der Dateiname muss zum Programmnamen „passen“, d.h. in diesem Fall `GGT.java` heißen.
- Das MiniJava-Programm ist der Rumpf des Hauptprogramms, d.h. der Funktion `main()`.
- Die Programmausführung eines Java-Programms startet stets mit einem Aufruf dieser Funktion `main()`.
- Die Operationen `write()` und `read()` werden in der Klasse `MiniJava` definiert.
- Durch `GGT extends MiniJava` machen wir diese Operationen innerhalb des GGT-Programms verfügbar.

```
1  import javax.swing.JOptionPane;
2  import javax.swing.JFrame;
3  public class MiniJava {
4      public static int read() {
5          JFrame f = new JFrame();
6          String s = JOptionPane.showInputDialog(f, "Eingabe:");
7          int x = 0; f.dispose();
8          if (s == null) System.exit (0);
9          try { x = Integer.parseInt(s.trim ());
10         } catch (NumberFormatException e) { x = read (); }
11         return x;
12     }
13     public static void write(String x) {
14         JFrame f = new JFrame ();
15         JOptionPane.showMessageDialog(f, x, "Ausgabe",
               JOptionPane.PLAIN_MESSAGE);
16         f.dispose ();
17     }
18     public static void write (int x) { write(""+x); }
19 }
```

Datei: "MiniJava.java"

Ein Java-Programm

Weitere Erläuterungen:

- Jedes Programm sollte Kommentare enthalten, damit man sich selbst später noch darin zurecht findet!
- Ein Kommentar in Java hat etwa die Form:
  `// Das ist ein Kommentar!!!`
- Wenn er sich über mehrere Zeilen erstrecken soll dann

```
/* Dieser Kommentar ist verdammt
   laaaaaaaaaaaang
      */
```


2 Eine einfache Programmiersprache

Das Programm GGT kann nun übersetzt und dann ausgeführt werden:

```
raecke> javac GGT.java
raecke> java GGT
```

- Der Compiler javac liest das Programm aus den Dateien `GGT.java` und `MiniJava.java` ein und erzeugt für sie JVM-Code, den er in den Dateien `GGT.class` und `MiniJava.class` ablegt.
- Das Laufzeitsystem java liest die Dateien `GGT.class` und `MiniJava.class` ein und führt sie aus.

Wichtige Erweiterung – Arrays

Arrays enthalten eine Gruppe von Variablen auf die über einen index zugegriffen wird:

- Deklariere Variablen a[0],...,a[99] und b[0],...,b[4]:

```
int[] a = new int[100], b = new int[5];
```

- Greife auf Element a[5] zu:

```
int i;
int[] a = new int[100];
a[5] = 1;   // a[5] ist jetzt 1
i = 5;
a[i] = 7;   // a[5] ist jetzt 7
i = 7;
a[i-2] = 8; // a[5] ist jetzt 8
```


Ausblick

MiniJava ist sehr primitiv

Die Programmiersprache Java bietet noch eine Fülle von Hilfsmitteln an, die das Programmieren erleichtern sollen.

Insbesondere gibt es

- viele weitere Datentypen (nicht nur int) und
- viele weitere Kontrollstrukturen

...kommt später in der Vorlesung!