4 Grundlagen der Syntax

4.1 Anweisungen

Anweisungen (Befehle, Kommandos, Statements) sind Programm-Einheiten, die "etwas bewirken".

```
print("Hallo");
a =7 +3;
```

4.2 Bezeichner

Bezeichner benennen Programmelemente (Klassen, Objekte, Methoden, Funktionen oder Daten). Dabei ist die Groß- bzw. Kleinschreibung relevant (a ist nicht gleich A).
Bezeichner können mit einem Buchstaben, einem Unterstrich _ oder einem Dollarzeichen $ beginnen. Danach sind Zahlen zulässig.
Wir merken uns: `Bezeichner sind vom Programmierer (frei) wählbare Programmelemente.`

4.3 Konventionen in Python

Bestimmte syntaktische Elemente sind innerhalb von Konventionen festgelegt. Dazu zählen

- **Klassennamen** beginnen mit einem Großbuchstaben
- **Methodennamen** beginnen mit einem Kleinbuchstaben
- **Konstanten** bestehen aus Großbuchstaben (z.B. PI)
- **Attribute** beginnen mit Kleinbuchstaben
- **Paketnamen** bestehen ausschließlich aus Kleinbuchstaben

4.4 Blöcke durch Einrückung

Blöcke werden durch geschweifte Klammern gebildet:
anweisun; ...
Sie werden als eine Anweisung behandelt, können geschachtelt werden und bestimmen (auch) die Sichtbarkeit von Variablen.
Lokale Variablen sind Variablen, die innerhalb eines Blockes deklariert werden und die auch nur in diesem Block sichtbar sind.
Klassen und Methoden bestehen aus mindestens einem Anweisungsblock.

4.5 Schlüsselworte

In Python existieren verschiedene reservierte Wörter für Elemente der Programmiersprache. Diese nennt man **Schlüsselwörter**, sie dürfen nicht für eigene Bezeichner verwendet werden.

Tabelle der Schlüsselwörter:

False	def	if	raise
None	del	import	return
True	elif	in	try
and	else	is	while
as	except	lambda	with
assert	finally	nonlocal	yield
break	for	not	class
from	or	continue	global
pass			

4.6 Deklarationen

Deklaration (Vereinbarung) legt die **Attribute**, also die Eigenschaften wie Typ, Name, Zugriffsrechte, Sichtbarkeit usw. von Programmelementen (Variablen, Klassen, Methoden, ...) fest.

Für die Deklaration von Daten in Java gilt folgende Syntax:
[modifier] typ datum1, datum2, ...;

- modifier (Modifikatoren) legen Eigenschaften von Programmelementen fest und werden an dieser Stelle noch nicht behandelt
- typ legt den Typ der Daten fest. Bei typ kann es sich um die Angabe eines Grunddatentyps oder um eine Klasse handeln (Referenzdatentyp).

Beispiele für Deklarationen:

```
int i, j, k; //Deklaration von drei Variablen vom Typ int (Grunddatentyp)
String str; //Deklaration der Variable str; str ist Objekt der Klasse String
Point p1,p2; // Deklaration von zwei Referenztypen p1 und p2 der Klasse Point
```

Deklarationen von Daten sind in Klassen, Methoden und Programmblöcken zulässig. Die Stellung der Deklaration bestimmt unter anderem die Sichtbarkeit der Daten. Die einfachste Form ist die Deklaration von lokalen Variablen innerhalb eines Blockes.

4.7 Eingabe-Applikation

Die einfachste Möglichkeit, Eingaben von der Tastatur an ein Java-Programm zu übergeben, besteht darin, die Eingaben gleich beim Aufruf des Programms als so genannte Aufrufparameter anzugeben.
Die Aufrufparameter werden als Folge von Zeichenketten (Zeichenfolge, Strings) unter dem Namen gespeichert, der in der Methode main() als String-Name angegeben ist.

In public static void main(String[] eingabe) enthält eingabe die Liste der beim Aufruf eingegebenen Zeichenfolgen.
Beispiel:
Das Programm gibt in einer Schleife die beim Programmstart angegebenen Zeichenfolgen nacheinander aus:

```
public class Eingabe1
{
        public static void main(STring[] args) {
                int i = 0;
                while(i < eingabe.length){
                        System.out.println(eingabe[i]);
                        i++;
                }
        }
}
```

4.8 Ausgabe-Applikation

Alle Beispiele zu grundlegenden Programmierelementen können anhand einer einfachen Ausgabe-Applikation ausprobiert werden.
Die folgende "einfache" Applikation kann als Gerüst für solche Ausgabeschritte in Java benutzt werden:

```
public class Ausgabe1 {
        public static void main(String[] args) {
                double w;
                w = 999.99;
                System.out.println("Hallo Java " + w);
        }
}
```

An dieser Stelle ist fpr das Verständnis zunächst nur wichtig:

- Jede Applikation muss die Zeile public class NameXyz() enthalten.
- Der Name NameXyz ist frei wählbar, aber wenn er geändert wird, muss auch der Name der Quelltext-Datei geändert werden
- Geschweifte Klammern {} umschließen Programmteile.
- Jede Applikation erfordert die Zeile:
  public static void main(String[] args)
- Innerhalb der runden Klammern von main() muss String vereinbart und ein frei wählbarer Name (z.B. args oder str, abc, ...) angegeben werden. Die eckigen Klammern bei String oder hinter dem Namen stehen.

Mit System.out. println () wird eine Zeichenfolge auf die Kommandozeile ausgegeben.

4.9 Ausdrücke

Ausdrücke sind in Java die kleinsten ausführbaren Einheiten innerhalb eines Programms. Ein Ausdruck besteht mindestens aus einem Operator und einem Operanden. Jeder Ausdruck hat einen Rückgabewert.

Beispiel:

ax − ay

Der Operator + (arithmetischer Operator) verknüpft die beiden Operanden ax und ay, der Rückgabewert ist die Summe der beiden Operanden (wenn beide Operanden Fließkomma- oder Ganzzahl-Daten sind).

Eigenschaften von Ausdrücken

- **Rückgabewert**
  Jeder Ausdruck hat einen Rückgabewert. Wert und Typ des Rückgabewertes ergeben sich aus dem Operator und den Operanden.
- **Auswertungsreihenfolge**
  Bindungs- und Assoziativitätsregeln (z.B. "Punkt- vor Strichrechnung")
- **Klammerung**
  Durch Klammerung lässt sich die Auswertungsreihenfolge festlegen/verändern.
- **Nebeneffekte**
  Veränderung von Variablenwerten ohne explizite Zuweisung

4.10 Zuweisungen

Die Zuweisung von Werten an Variablen geschieht in Java mit dem einfachen Gleichheitszeichen Der Wert des Ausdrucks recht vom Gleichheitszeichen wird der Variablen links vom Gleichheitszeichen zugewiesen. Der Typ des Ausdrucks und der Typ der Variable müssen zueinander passen.

Beispiel:

```
int a;
a =3 * 6;
```

4.11 Kommentare

Kommentare dienen der besseren Lesbarkeit (Dokumentation) von Programm-Quelltexten. Sie werden vom Compiler ignoriert. Der Standars-Kommentar beginnt mit /* und endet mit */. Er kann sich über mehrere Zeilen erstrecken. Dabei gilt: Kommentare können nicht geschachtelt werden.

Beispiel:

```
/* Das ist ein Kommentar*/
```

Zeilenkommentar

Java (und die meisten C-Compiler) kennen den Zeilenkommentar. Der Zeilenkommentar beginnt it // und endet am Zeilenende.

Beispiel:

```
int j = 0;    // Das ist der Kommentar
```