Arbeitsblatt #1

Einführung in die Programmierung mit *Go*

8. März 2015

1 Start

- Sofern Sie einen Rechner der RBI verwenden: Melden Sie sich an; das *go*-System ist bereits vorinstalliert.
- Ansonsten: Installieren Sie *go* auf Ihrem Laptop. Folgen Sie den auf der *go*-Webseite[1] aufgeführten Schritten.

2 Hallo, Welt!

Erzeugen Sie in einem Verzeichnis Ihrer Wahl das Programm `hallo.go`:

```
package main

import "fmt"

func main() {
    fmt.Println("Hallo, Welt!")
}
```

Führen Sie das Programm mit `go run hallo.go` aus, um sicherzustellen, daß die Installation korrekt funktioniert.

3 Formatierte Ausgabe

Das Paket `fmt` stellt mehrere Funktionen zur Ausgabe von Text zur Verfügung:

- `fmt.Print(`x_1`, ..., `x_n`)`:
  Druckt nacheinander x_1 bis x_n auf die Standardausgabe aus
- `fmt.Println(`x_1`, ..., `x_n`)`:
  Wie `Print`, aber druckt auch einen anschließenden Zeilenumbruch
- `fmt.Printf(`*`fmt`*`, `x_1`, ..., `x_n`)`:
  Druckt gemäß der Formatierungsspezifikation *fmt* formatierten Text aus

a. Führen Sie folgenden Befehl aus:
`fmt.Printf("A %v B %T\n", 1, 1)`

b. Verändern Sie die Parameter; verwenden Sie auch Parameter anderer Typen. Finden Sie heraus:

- Was bedeutet das '`%v`'?
- Was bedeutet das '`%T`'?

[1] `https://golang.org/doc/install`

- Was bedeutet das '\n'?

c. '%v' und '%T' werden von der *Go*-Spezifikation auch als Format-'Verben' bezeichnet. Was passiert, wenn die Anzahl der Format-Verben nicht der Anzahl der zusätzlichen Parameter (nach *fmt*) entspricht?

d. Verwenden Sie die Format-Verben '%b' und '%x' mit ganzen Zahlen. Was passiert?

4 Ganze Zahlen

Betrachten Sie das folgende Programmfragment:

```
var x int32 = 23
var y int64 = x
```

Beachten Sie, daß die vorige Aufgabe Ihnen zeigt, wie Sie die Typen von Variablen ausgeben können.

- Erlaubt das Typsystem diese Zuweisung?
- Erlaubt das Typsystem eine solche Zuweisung, wenn der Typ `int` statt `int32` (bzw. `int` statt `int64`) ist?
- Was ist der Typ einer Ganzzahlen-Konstante wie `23`? Erklären Sie, wie Ihre Untersuchungsergebnisse zueinander passen.
- Sie können Typen auch explizit konvertieren; indem Sie `int32(x)`, wird `x` zu einem Wert des Typs `int32` umgewandelt. Was passiert, wenn Sie in einen Typen konvertieren, in dem die betreffende Zahl nicht mehr dargestellt werden kann (z.B. eine negative Zahl in einen Typen ohne Vorzeichen)? Versuchen Sie dies sowohl durch Konvertierung einer Konstante als auch durch Konvertierung einer Variablen.

5 Sichtbarkeit von Variablen

Verschiedene Programmiersprachen haben verschiedene Regeln zur Sichtbarkeit und Deklaration von Variablen. In dieser Aufgabe untersuchen Sie die Sichtbarkeitsregeln von *Go*.

Betrachten Sie folgendes Programm:

```
package main

import "fmt"

func main() {
    ...
}

var s string = "Zeichenkette"
```

a. Kann `main` auf die globale Variable `s` zugreifen?

b. Kann `main` eine eigene, neue Variable mit dem Namen `s` deklarieren?

- Falls ja:
  - Kann `main` selbst die gleiche Variable mehrfach deklarieren (per `var`)?
  - Kann `main` selbst die gleiche Variable mehrfach deklarieren (per `:=`)?
  - Ist eine solche Mehrfachdeklaration erlaubt, wenn eine bereits deklarierte Variable zusammen mit einer zweiten Variablen deklariert wird (z.B. `var s, neu string`)?
  - Ist eine solche Mehrfachdeklaration erlaubt, wenn eine bereits deklarierte Variable zusammen mit einer zweiten Variablen per `:=` deklariert wird?

6 Unnötige Ausführung

Betrachten Sie das folgende Programmfragment:

```
func f() int {
   ...
}
func g() int {
   ...
}
func h() int {
   ...
}

func main() {
    if (f() > g()) && (h() > 0) {
          ...
    }
}
```

Die Bedingung in `main` besteht aus zwei Konjunkten:

- `(f() > g())`
- `(h() > 0)`

Falls das erste Konjunkt nicht wahr ist (also `f()` nicht größer als `g()`), steht bereits fest, daß die gesamte Bedingung nicht wahr ist. In manchen Programmiersprachen wird daher eine sogenannte *Kurzschlußauswertung* durchgeführt: Wenn das erste Konjunkt den Wert `false` ergibt, wird das zweite Konjunkt übersprungen, also nicht ausgewertet.

a. Überprüfen Sie: Verwendet *Go* Kurzschlußauswertung für `&&` ?