Fingerübungen

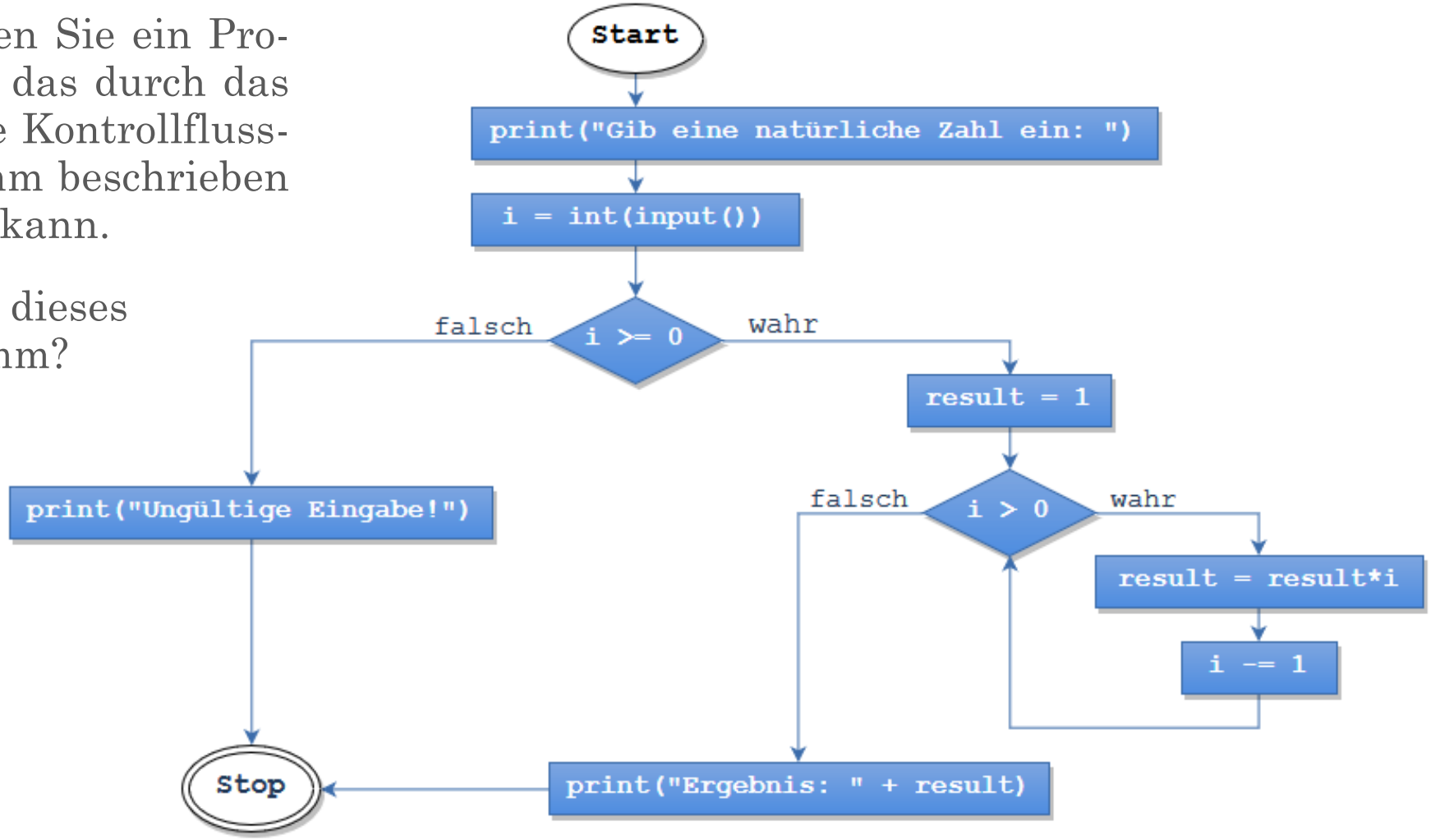
Woche 03 – 25. Mai 2016

Kontrollflussdiagramme

Kontrollflussgraph („*flowchart*“) und Ablaufsteuerung mit `if` und `while`

Schreiben Sie ein Programm, das durch das folgende Kontrollflussdiagramm beschrieben werden kann.

Was tut dieses Programm?



Die Funktion `range()`

Zahlensequenzen (*int*) mittels **`range(`**`[start,]` **`stop`**`[, step]`**`)`**

(1) **range()**

Überlegen Sie sich zunächst, welche Werte (*int*) in den nachfolgenden Sequenzen enthalten sind, und ***überprüfen*** Sie Ihre Prognose anschließend mithilfe der *Python Shell*:

a) `range(5)`
b) `range(-4, 7)`
c) `range(17, 17)`
d) `range(17, 18)`
e) `range(6, 3)`
f) `range(6, 3, -1)`
g) `range(0, 4, 2)`
h) `range(-4, 6, 3)`
i) `range(4, 6, 0.5)`

Hinweis: Sie können die Funktion `list()` nutzen, um alle Werte einer Zahlensequenz aufzulisten! Schreiben Sie dazu bspw.:

```
print(list(range(5)))
```

(2) [*Zusatzaufgabe*] **Sequenz gesucht!**

Welche Parameter (`start`, `stop` und `step`) können der `range`-Funktion übergeben werden, damit diese die Zahlen **`-3`, `1` und `5`** (und **nur** diese Zahlen) enthält? Es gibt mehrere Möglichkeiten, finden Sie mind. zwei!

for-Schleifen

Iterieren mit `for`-Statements, `range()`-Funktion und `in`-Operator

(1) Iterieren und Quadrieren

Schreiben Sie ein Programm, das nacheinander für alle *ungeraden* Zahlen von 1 bis `n` (jeweils inklusive) das Quadrat ausgibt. `n` ist dabei natürlich eine Variable, d. h. die Ausgabe für `n = 7` würde wie rechts dargestellt aussehen.

Sie müssen `n` *nicht* vom Benutzer abfragen.

```
1
9
25
49
```

(2) [*Zusatzaufgabe*] `in`-Operator bei Strings

Der `in`-Operator („Element von“) kann u. a. auch auf Strings (*str*) angewandt werden. Warum das so ist wird im Laufe der Vorlesung noch klar. *Wenden* Sie den Operator nun auf Strings an, indem Sie `for`-Schleifen und/oder die *Python Shell* verwenden. Welchen Ergebnistyp hat der `in`-Operator und wie erfolgt die Auswertung bei Strings? Beispieltests wären:

Auf der Python Shell:

```
- "A" in "abcdefg"
- "el" in "Hello World"
```

Als Programm:

```
for i in "aeiou 123":
    print(i)
```

`for`-Schleifen

`for-else`-Statements mit `break` und `continue`

(3) [*Zusatzaufgabe*] Wird nicht besprochen (→ „Lösung“ auf Moodle)!

`for`-Schleifen können (wie `while`-Schleifen) einen **`else`-Block** besitzen, der ausgeführt wird, sobald alle Elemente aus `range(…)` behandelt wurden. *Experimentieren Sie* mit dem `else`-Teil und den beiden Statements **`break`** und **`continue`**! Welchen Wert hat die Zählervariable (z. B. `i`) beim Erreichen des `else`-Teils bzw. danach? Unterscheidet sich der `else`-Block von dem „normalen“ Teil, der nach der Schleife folgt (siehe Aufgabe (3) zu `while`-Schleifen → Zusammenhang mit `break`)?

Versuchen Sie sich auch an **verschachtelten Schleifen** (*„nested loops“*)!