

# INFORMATIK I

Tutorium 4 — 08. November 2024

haskell.hs

Name Last  
Universität Münster



L<sup>A</sup>T<sub>E</sub>X-Vorlage von  
Florian Sihler

# Übungsblatt 3

1

## Aufgabe 1: a) 'Spezifikation' erfüllt?

Lies eine **Zahl** von der Tastatur ein, **berechne die Quadratwurzel**, und **gib das Ergebnis am Bildschirm aus**.

## Lösung 1: a)

1. Vollständig: Welche Zahlendarstellung? Negative Zahlen? ✗
2. Detailliert: Welche Grundoperationen sind erlaubt? ✗
3. Unzweideutig: Was heißt „ausgeben“? ✗

## Aufgabe 1: b) 'Spezifikation' erfüllt?

Berechne die Länge des Wortes als Binärzahl, und zwar in der Formulierung aus Aufgabe 4 von Übungsblatt 2.

**Aufgabe 4** (Trennungspunkt: 4%)  
Geben Sie eine Turingmaschine an, welche jedes eingelesene Wort über dem Alphabet  $\Sigma = \{0, 1\}$  akzeptiert und welche die Länge des Wortes als Binärzahl der Ausgabe produziert. Definieren Sie folgende Funktionen:

- Die Länge (Schreibzahl) eines Wortes  $w$  ist die Anzahl der Zeichen in  $w$ .
- Berechnen Sie die Länge des Wortes  $w$  (Schreibzahl) von  $w$ .
- Am Ende der Auswertung einer Turingmaschine steht die Länge des Wortes  $w$  (Schreibzahl) der Länge (Schreibzahl) und nicht die Länge des Wortes  $w$ .
- Geben Sie die Turingmaschine als Tupel an. Stellen Sie die Übergangsfunktion  $\delta$  (Schreibzahl der Länge in der Vorzeichen und in der verschobenen Länge).
- Geben Sie die Bedeutung der von Ihnen verwendeten Zustände an.
- Wenn Sie zusätzliche Details einfügen möchten, erläutern Sie dies.
- Es reicht Ihnen aus, wenn die Turingmaschine mit dem korrekten Wort macht. Dokumentieren Sie Ihre Konstruktion.

## Lösung 1: b)

1. Vollständig: ✓
2. Detailliert: mathematisch formales Maschinenmodell ✓
3. Unzweideutig: Turingmaschine ✓

# EUKLIDISCHER ALGORITHMUS

## Aufgabe 2: a)

Führen Sie den Algorithmus *schrittweise* für  $m = 27$  und  $n = 12$  aus.

## Lösung 2: a)

▷  $m = 27, n = 12$

**if**  $m = 0$  **then**

$\text{result} \leftarrow n$

**else**

**while**  $n \neq 0$  **do**

**if**  $m > n$  **then**

$m \leftarrow m - n$

**else**

$n \leftarrow n - m$

**end if**

**end while**

$\text{result} \leftarrow m$

**end if**

▷  $\text{result} \leftarrow 3$

## Aufgabe 2: b)

Welches Ergebnis produziert der Algorithmus? Warum führen die Subtraktionen zum gewünschten Ergebnis?

## Lösung 2: b)

- Der Algorithmus berechnet den größten gemeinsamen Teiler (ggT) von  $m$  und  $n$ . ( $m = 0 \implies \text{return } n$ ;  $n = 0 \implies \text{return } m$ )
- Jeder gemeinsame Teiler von  $m > n$  muss auch Teiler von  $m - n$  sein:  
 $m = t \cdot k_1, n = t \cdot k_2 \implies m - n = t \cdot k_1 - t \cdot k_2 = t \cdot (k_1 - k_2)$

## Aufgabe 2: c)

Geben Sie das Ergebnis in Form einer Nachbedingung an.

## Lösung 2: c)

`result = ggT(m,n)`: `result` ist Teiler von  $m$  und  $n$  und für jede Zahl  $z : z \mid m \wedge z \mid n$  gilt  $z \leq \text{result}$ .

## Lösung 3: b) Arbeit mit der Kommandozeile

```
mkdir Studium  
cd Studium  
mkdir 2024-WiSe  
mkdir 2024-WiSe/Informatik-I  
cd 2024-WiSe/Informatik-I  
mkdir Folien  
mkdir Übungen
```



## Lösung 3: b) Arbeit mit der Kommandozeile

```
cd ~/Downloads
mv 00_Organisatorisches_v2.pdf ~/Studium/2024-WiSe/↵
    Informatik-I/Folien
cd ~/Studium/2024-WiSe/Informatik-I/Übungen
ghc -o hello hello.hs
ls                                #> hello hello.hi hello.hs hello.o
/hello                           #> Hello World!
```

## Aufgabe 4

Programmieren Sie folgende Funktionen in Haskell:

- a) inchesToCentimeters
- b) footToCentimeters
- c) circleArea

## Lösung 4: a) inchesToCentimeters

```
-- 1 Zoll = 2,54 cm
centimetersPerInch :: Float
centimetersPerInch = 2.54

-- Einfach nur multiplizieren :D
inchesToCentimeters :: Float -> Float
inchesToCentimeters inches = inches * centimetersPerInch
```

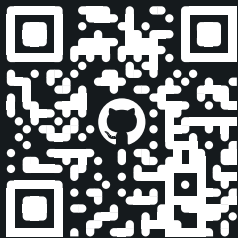
## Lösung 4: b) footToCentimeters

```
-- 1 Fuß = 12 Zoll
inchesPerFoot :: Float
inchesPerFoot = 12

-- Multiplizieren und Umwandeln
footToCentimeters :: Float -> Float
footToCentimeters foot = inchesToCentimeters (foot * ←
    inchesPerFoot)
```

## Lösung 4: c) circleArea

```
-- Radius -> cm -> pi * r^2
circleArea :: Float -> Float
circleArea radiusFoot = pi * (footToCentimeters radiusFoot)^2
```



Die Tutoriumsfolien sind jetzt auch auf [GitHub](#) !

**Name Last**

Münster, 13. Januar 2025

[name.last@uni-muenster.de](mailto:name.last@uni-muenster.de)