Compiler
— Blatt 2 —

Prof. Dr. Oliver Braun

Fakultät für Informatik und Mathematik
Hochschule München

Sommersemester 2014

03.04.14 12:10

Aufgabe — Endlicher Automat

Entwickeln Sie einen endlichen Automat der `Integer` akzeptiert.

a) Geben Sie das Zustandsübergangsdiagramm an.

b) Geben Sie den Endlichen Automat als mathematisches Objekt an.

c) Implementieren Sie den endlichen Automat in Haskell.

Schauen Sie sich dazu noch einmal den endlichen Automat aus der Vorlesung an.

Sie können Ihren Automat mit folgendem Modul testen

```haskell
module Test where

import Main      ( accept  )
import Data.Char ( isDigit )
import Test.QuickCheck

prop_acceptInteger :: Integer -> Bool
-- prop_acceptInteger i = accept $ show i

-- mit einem Lambda-Ausdruck
-- prop_acceptInteger = \i -> accept $ show i

-- ohne den Parameter hinzuschreiben, Funktionskomposition
```

```
prop_acceptInteger = accept . show

prop_doNotAcceptString :: String -> Bool
prop_doNotAcceptString = not . accept . filter (not . isDigit)

main :: IO ()
main = do
  verboseCheck prop_acceptInteger
  verboseCheck prop_doNotAcceptString
```

In Ihrem `Main`-Modul muss, wie im Beispiel aus der Vorlesung, eine Funktion

```
accept :: String -> Bool
```

definiert sein, die `True` genau dann zurück gibt, wenn der `String` einem `Integer` entspricht und `False` sonst.

Versuchen Sie als erstes das gesamte Test-Modul zu **verstehen**. Informationen über die verwendeten Funktionen finden Sie im Haskell Center, in Hoogle oder direkt in der Dokumentation der *GHC standard libraries*.