

Übungen zur „Deskriptiven Programmierung“ Blatt 9

Ein „pretty printer“ ist ein Programm, das Daten für das menschliche Auge aufbereitet. Daten können zum Beispiel XML-Dokumente, Tabellen, aber auch Programme sein. So gibt es etwa für viele Programmiersprachen „pretty printer“, die Programme nach bestimmten, vorgegebenen Richtlinien formatieren. Betrachten wir ein konkretes Beispiel: Das folgende Haskell-Programm definiert einen Datentyp *Stat*, mit dem Statements einer imperativen Sprache repräsentiert werden können.

```
type Expr = String
data Stat = Expr Expr -- expression statement
          | While Expr Stat -- iteration statement
          | If Expr Stat Stat -- selection statement
          | Block [Stat] -- compound statement
          | Return Expr -- jump statement
deriving (Show)
```

Hier ist ein Beispiel für ein Statement (binäre Suche in einem Feld).

```
binsearch :: Stat
binsearch =
  Block [
    Expr "low = 0",
    Expr "high = n - 1",
    While "low <= high" (
      Block [
        Expr "mid = (low + high) / 2",
        If "x < v[mid]" (
          Expr "high = mid - 1" (
            If "x > v[mid]" (
              Expr "low = mid + 1" (
                Return "mid"
              )
            )
          )
        ),
        Return "-1"
      ]
    )
  ]
```

Wenn wir *binsearch* mit Hilfe der Methode *show* in einen String überführen, geht die Struktur verloren.

```
Main> putStrLn (show binsearch)
Block [Expr "low = 0",Expr "high = n - 1",While "low <= high" (Block [Expr "mid
= (low + high) / 2",If "x < v[mid]" (Expr "high = mid - 1" (If "x > v[mid]" (Ex
pr "low = mid + 1" (Return "mid")))],Return "-1"]
```

Ein „pretty printer“ ersetzt die abstrakte Syntax durch konkrete Syntax und hebt die Struktur des Datums, hier des Programms, durch Einrückung hervor.

```

Main> putStrLn (render (Page 50) (pretty binsearch))
{
  low = 0;
  high = n - 1;
  while (low <= high)
  {
    mid = (low + high) / 2;
    if (x < v[mid])
      high = mid - 1;
    else
      if (x > v[mid])
        low = mid + 1;
      else
        return mid;
  }
  return -1;
}

```

Die Angabe *Page 50* spezifiziert eine maximale Zeilenbreite von 50 Zeichen. Wird die Breite erhöht, kann unter Umständen die Höhe der Ausgabe verringert werden, indem Konstrukte vollständig in eine Zeile gesetzt werden.

```

Main> putStrLn (render (Page 100) (pretty binsearch))
{
  low = 0;
  high = n - 1;
  while (low <= high)
  {
    mid = (low + high) / 2;
    if (x < v[mid]) high = mid - 1; else if (x > v[mid]) low = mid + 1; else return mid;
  }
  return -1;
}

```

Aufgabe 26. Entwerfe eine Bibliothek von Kombinatoren, um die Programmierung von „pretty printern“ zu unterstützen. Verwende die Bibliothek, um einen „pretty printer“ für den Datentyp *Stat* zu programmieren.

Aufgabe 27. Welche Gesetze erfüllen Deine Kombinatoren?