Übung 6 – Musterlösung

Aufgabe 1: Schreiben Sie eine Funktion reverse[T](l:List[T]):List[T] die eine beliebige Liste unter der Verwendung von foldLeft oder foldRight umdreht.

Aufgabe 2: Schreiben Sie eine Funktion *foldLeft[B,T] (base :B, l:List[T]) (B,T)=>b)* unter Verwendung von foldRight.

Aufgabe 3: Gegeben sei die folgende Liste, die ausdrückt, welche Programmiersprachen, welche Programmiersprachen welche Paradigmen unterstützen:

val Paradigmen=List(("erlang", "funktional"), ("erlang", "logisch"), ("prolog", "logisch"), ("scala", "funktional"), ("scala", "objektorientiert"), ("scapla", "logisch"), ("java","objektorientiert"))

a) Ermitteln Sie unter Verwendung von foldLeft oder foldRight welches Paradigma wie häufig in der Liste vorkommt wurde.

b) Berechnen Sie aus dem Ergebnis von a die relative Häufigkeit, mit der ein Paradigma vorkommt.

Aufgabe 4: Gegeben sei die Funktion mapReduce mit dem folgenden Implementierung:

```
def mapReduce[S,B,R](mapFun:(S=>B), redFun:(R,B)=>R, base:R, l:List[S]):R =
        l.map(mapFun).foldLeft[R](base)(redFun)
```

Berechnen Sie mit dieser Funktion für alle Werte der Eingabeliste die kleinsten Primteiler und addieren Sie diese.

Aufgabe 5: Schreiben Sie eine Funktion partial mit der folgenden Signatur:
def partial[A,B,C](a:A,f:(A,B)=>C):B=>C
Die Funktion bekommt als Parameter:

- eine Funktion mit 2 Variablen sowie
- ein Wert mit der die Funktion belegt werden soll.

Sie soll eine Funktion zurück liefern, bei der erste Parameter bereits belegt wurde.

Beispiel: partial(1, (a,b)=>a+b) soll eine Funktion zurückliefern, die einen Wert um 1 erhöht.