Aufgabe 1

```
length1 :: [a] -> Int
length1 xs = sum map mach1 xs
mach1 :: a -> Int
mach1 x = 1
```

Aufgabe 3

```
iter::Int -> (a -> a) -> a -> a
iter 0 f a = a
iter n f a = (iter (n-1) f f a)
```

Aufgabe 4

```
sumOfSquares :: (Num a) => [a] -> a
sumOfSquares xs = foldr (+) 0 (map (^2) xs)
```

Aufgabe 5

```
layIndex :: layIndex::[(Word, [Int])] -> IO()
layIndex a putStr (toString a)
   where toSring ((w,z):ws) = w ++ "␣" ++
    (filter (`notElem` "[]") (show z) ++ "\n" ++ (toString ws)
         toString [] = ""
```

Wiederholung Sortieralgorithmen
Insertion Sort (Sortieren beim Einfügen)

```
insert :: (Ord a) => a -> [a] -> [a]
insert x [] = [x]
insert x (y:ys)
  | x == y = x:y:ys
  | otherwise = y:(insert x ys)

iSort :: (Ord a) => [a] -> [a]
iSort [] = []
iSort (a:as) = insert a (iSort as)
```

Merge Sort

```
merge [] as = as
merge as [] = as
merge (a:as) (b:bs)
  | a == b = a: merge as (b:bs)
  | otherwise b: merge (a:as) bs
mSort [] = []
```

```
mSort (x:[]) = [x]
mSort xs = merge (mSort)
```

Quick Sort

```
qSort [] = []
qSort (x:xs) 0 qSort
```

Selection Sort (Sortieren beim Aussuchen)