

Lösung Übungsblatt Nr. 7 / ALP 1 zur Abgabe 16.01.2004

Aufgabe 2:

```
getnpos :: String->Int->Char
getnpos (head:tail) 1 = head
getnpos (head:tail) p = getnpos tail (p-1)

radixsort :: [String]->Int->[String]
radixsort [] foo = []
radixsort endrecursion 0 = endrecursion
radixsort liste cnt = radixsort ([x | x <- liste, [(getnpos x cnt)]=="0"]++[y | y <- liste, [(getnpos
y cnt)]=="1"]) (cnt-1)

callradix :: [String]->[String]
callradix [] = []
callradix (head:tail) = radixsort ([head] ++ tail) (length head)

Main> callradix ["10101","11100","01010","00001","11001","01011"]
["00001","01010","01011","10101","11001","11100"]
Main> callradix ["11","10","00","01"]
["00","01","10","11"]
Main> callradix ["10000","01100","01110","10001","10101","01111"]
["01100","01110","01111","10000","10001","10101"]
Main>
```

Der Radix-Sort Algorithmus arbeitet eine gegebene Liste von Zahlen ab und sortiert sie. Dazu schaut er sich zuerst die letzte Stelle jeder Zahl an und sortiert die gegebenen Zahlen nur danach. Das gleiche macht er einzeln für jede Stelle die die gegebenen Zahlen haben, bis zur ersten Stelle und fertig.

Das eigentliche Sortieren in dieser Implementation wird über die Erstellung einer neuen Liste per Filter realisiert, es werden 2 Listen erstellt und zusammengefügt, wobei die erste Liste nur alle Zahlen mit "0" an der x-ten Stelle und die zweite Liste nur Zahlen mit "1" an der x-ten Stelle enthält. Beide Listen hintereinander kombiniert ergeben dann die gewünschte 0-1 Ordnung.

Aufgabe 1:

Leider nicht gelöst.