1 Übung 3

(ignorieren wir, dass Blatt 8 Übung 5 maximum nicht minimum definiert)

minimum arbeitet von $List\ Nat \to Nat$, da map selbst auf listen operiert und auf jedem Element "f" (bzw hier length:$List\ a \to Nat$) ausführt. muss map length auf listen von listen operieren und minimum gibt die minimale länge der listen in den listen der Listen zurück:

```
minimum.(map length)::List (List a)->Nat
```

2.

map ersetzt jedes element der List mit einem transformierten, aber typgleichen element der Liste:

benutzen die foldl mit argumentreihenfolge anders herum? MINDFUCK

es gehört $Foldable\ t \Rightarrow (b \to a \to b) \to b \to ta \to b$

UND DAS GLEICHE BEI DER ARGUMENTFUNKTION!!

```
-- reverse is necessary because the construction is resolved from back to front (so the first i
map f xs =reverse $foldl  Nil (\x-> Cons (f x)) xs


reverse xs = foldL c g xs Nil
 where
         --c::List a->List a
         c = id
         g::a->(List a->List a)->(List a->List a)
         g y f ys = Cons (f y) ys
```

2 Übung 4

Hintergrund:

die n-te quadratzahl kann von der vorherigen über:

$$n^2 = (n-1)^2 + (n-1) + n$$

für diese berechnung brauchen wir also die zahl n selbst, sowie das vorherige quadrat

```
-- initialfall 0**2 = 0 zu (n,n**2)
c = (0,0)
```

```
h x = (Suc(fst x), (snd x)+(fst x)+ Suc(fst x))
g = fst
```

für die 2. Funktion:

```
-- wir dürfen annehemen, dass n=0 wahr ist (laufvariable, letzter treffer)
c= (0,0)
h x= if p (fst x) then (Suc(fst x), fst x) else (Suc(fst x), snd x)
g= snd
```