1) Define compose

2) Rewriting code on board

addlist . x . y . = myfunc' p1 . x ++ y :: [2 | 2 <- x, 2 > x] - y

where p1 . x = x <= y

Define myfunc

3) which is more general map or reduce? foldr

x more general than y means

y can be written using x

4) zip . [1,2,3] . [4,5,6] = [(1,4), (2,5), (3,6)]

zip . [] . [4,5,6] = []

Define zip

what happens if second list empty & not the first?

5) cons . x . y = x :: y

whats the type of cons?

write it in fully explicit form

6) Does above type of cons allow it to be a foldable operator - with foldl? with foldr?

const . a . b = a what does this function do? whats its type?