1.

map:: (a->b)->[a]->[b]

map f [] = [] (map.1)

map f (x: xs) = f x: map f xs (map.2)

(++) :: [a]->[a]->[a]

[] ++ ys = ys (++.1)

(x: xs) ++ ys = x: (xs ++ ys) (++.2)

map f (xs ++ ys) = (map f xs) ++ (map f ys)

Induction on xs

Base case xs = []

map f (xs ++ ys)

= map f ys (++.1)

(map f xs) ++ (map f ys)

= [] ++ (map f ys) (map.1)

= map f ys (++.1)

Induction step (xs = x: xs'):

map f (xs ++ ys)

= map f ((x:xs') ++ ys)

= map f (x: xs' ++ ys) (++.2)

= f x: map f (xs' ++ ys) (map.2)

(map f xs) ++ (map f ys)

= (map f (x: xs')) ++ (map f ys)

= (f x: (map f xs')) ++ (map f ys) (map.2)

= f x: ((map f xs') ++ (map f ys)) (++.2)

= f x: map f (xs' ++ ys) (by IH)

2.

zip :: [a]->[b]-> [(a, b)]

zip [ ] _ = [ ] (zip.1)

zip _ [ ] = [ ] (zip.2)

zip (x: xs) (y: ys) = (x, y): zip xs ys (zip.3)

(1)

Induction on ps

Base case (ps = [ ]):

zip (fst (unzip ps)) (snd (unzip ps))

= zip (fst ([ ], [ ])) (snd ([ ],[ ])) (unzip.1)

= zip ([ ]) ([ ]) (by definition of fst, snd)
= [ ] (zip.1)

Induction step (ps = (p1, p2): ps’):
zip (fst (unzip ((p1, p2): ps’))) (snd (unzip ((p1, p2): ps’)))
= zip (fst ((p1: xs, p2: ys) where (xs, ys) = unzip ps’)) (snd (unzip ((p1, p2): ps’))) (unzip.2)
= zip (p1:xs where (xs, ys) = unzip ps’) (snd (unzip ((p1, p2): ps’))) (by definition of fst)
= zip (p1: fst (unzip ps’)) ((snd (unzip ((p1, p2): ps’))) (by definition of fst)
= zip (p1: fst (unzip ps’)) (snd ((p1: xs, p2: ys) where (xs, ys) = unzip ps’)) (unzip.2)
= zip (p1: fst (unzip ps’)) (snd (p2: ys where (xs, ys) = unzip ps’)) (by definition of snd)
= zip (p1: fst (unzip ps’)) (p2: snd (unzip ps’)) (by definition of snd)
= (p1, p2): zip (fst (unzip ps’)) (snd (unzip ps’)) (zip.3)
= (p1, p2): ps’ (by IH)
= ps

(2)
xs,ys 同时为有限列表，并且二者长度相同

xs,ys 同时为有限列表，并且二者长度相同

Induction on xs, ys

Base case xs = [ ], ys = [ ]
unzip (zip xs ys)
= unzip [ ] (zip.1)
= ([ ], [ ]) (unzip.1)
= (xs, ys)

Induction step (xs= x: xs’, ys = y: ys’, length xs’ = length ys’):
unzip (zip (x: xs’) (y: ys’))
= unzip ((x,y): zip xs’ ys’) (zip.3)
= (x: xs”, y: ys”) where (xs”, ys”) = unzip (zip (xs’, ys’)) (unzip.2)
= (x: xs’, y: ys’) (by IH)
= (xs, ys)