Question 1:
For all n, P(n) holds if P(0) holds and P(n)->P(n+1)
Base case:
P(0): power 0 $x=x^0$
power 0 x
=1.0 by definition of power
$=x^0$ by properties of zero exponent
Inductive case:
P(n+1):
show: power n+1 $x=x^{n+1}$
given: power n $x=x^n$

power n+1 x
=x*power(n+1-1) x
=x*power n x by definition of power
$=x*x^n$ by inductive hypothesis
$=x^{n+1}$ by definition of exponent

Question 2:
For all n in nat, P(n) holds if P(Zero) holds and P(n)->P(Succ n)
Base case:
P(Zero): power Zero $x=x^{toInt(Zero)}$

power Zero x
=1.0 by definition of power
$=x^0$ by properties of exponent
$=x^{toInt(Zero)}$ by definition of toInt
Inductive case:
P (Succ n):
show power Succ n $x=x^{toInt(Succ\ n)}$
given power n $x=x^{toInt(n)}$

power Succ n x
=x*power n x by definition of power
$=x*x^{toInt(n)}$ by definition of inductive hypothesis
$=x^{toInt(n)+1}$ by definition of exponent
$=x^{toInt(Succ\ n)}$ by definition of toInt

Question 3:
For all l2, P(l1, l2) holds if P([],l2) holds and P(xs,l2)->P(x::xs,l2)
P (l1, l2): reverse (append l1 l2) = append (reverse l2) (reverse l1)
Induction is on l1
Lemma 1: append lst []=lst
Base case:

append [] []=[]
append [] []
=[] by definition of append
inductive case:
show append x::xs []=x::xs
given append xs []=xs
append x::xs []
=x::append xs [] by definition of append
=x::xs by inductive hypothesis

Lemma 2: append (append l1 l2) l3=append l1 (append l2 l3)
Base case: append (append [] l2) l3=append [] (append l2 l3)
append (append [] l2) l3
=append l2 l3 by definition of append
=append [] (append l2 l3) by definition of [] and append
inductive case: P (x::xs, l2,l3):
show: append (append x::xs l2) l3=append (x::xs) (append l2 l3)
given: append (append xs l2) = append xs (append l2 l3)
append (append x::xs l2) l3
=append (x:: append xs l2) l3 by definition of append
=x::append (append xs l2) l3   by definition of append
=x:: append xs (append l2 l3) by definition of append
=append (x::xs) (append l2 l3) by definition of append
Base case:
P ([], l2): reverse (append [] l2) = append (reverse l2) (reverse [])

reverse (append [] l2)
=reverse l2 by definition of append
=append (reverse l2) [] by Lemma 1:append lst [] = lst
=append (reverse l2) (reverse []) by definition of reverse
Inductive case:
P (x::xs, l2):
show: reverse (append x::xs l2) = append (reverse l2) (reverse x::xs)
given: reverse (append xs l2) = append (reverse l2) (reverse xs)

reverse (append x::xs l2)
=reverse (x:: (append xs l2)) by definition of append
=append (reverse (append xs l2)) [x] by definition of reverse
=append (append (reverse l2) (reverse xs)) [x] by inductive hypothesis
=append (reverse l2) (append (reverse xs) [x]) by Lemma 2: append (append l1 l2) l3=append l1 (append l2 l3)
=append (reverse l2) (reverse x::xs) by definition of reverse

Question 4:
For all l2, P(l1, l2) holds if P([],l2) holds and P(xs,l2)->P(x::xs,l2)
P (l1, l2): someupper (l1 @l2) = someupper l1 || someupper l2
Induction is on l1
Base case:
P ([], l2): someupper ([] @l2) = someupper [] || someupper l2
someupper ([] @l2)
=someupper l2 by properties of [] and l2
=false ||someupper l2 by disjunction
=someupper [] ||somupper l2 by definition of someupper
Inductive case:
P (x::xs, l2):
show: someupper (x::xs @l2) = someupper x::xs || someupper l2
given: someupper (xs @l2) = someupper xs || someupper l2

someupper (x::xs @l2)
=someupper (x::(xs@l2)) by properties of :: and @
=isupper x||someupper (xs @ l2) by definition of someupper
=isupper x|| somupper xs ||someupper l2 by inductive hypothesis
=(isupper x ||someupper xs)|| someupper l2 by properties of or
=someupper x::xs ||someupper l2 by definition of someupper

Question 5:
For all lst, P(lst) holds if P([]) holds and P(xs)->P(x::xs)
Base case:
P([]):
someupper []=foldupper []
someupper []
=false by difition of someupper
=false|| false by disjunction
=foldr upper [] false ||false by definition of foldr
=foldupper [] by definition of foldupper

Inductive case:
P(x::xs)
show: someupper x::xs=foldupper x::xs
given someupper xs=foldupper xs

someupper x::xs
=isupper x ||someupper xs by definition of someupper
=isupper x ||foldupper xs by inductive hypothesis
=isupper x||foldr upperor xs false by definition of foldupper
=upperor x (foldr upperor xs falseby) definition of upperor

=foldr upperor x::xs false by definition of foldr
=folduppwe x::xs by definition of foldupper

Question 6:
For all t, P(t) holds if P(Leaf v) holds and P(t1),P(t2)->P(Branch (t1,t2))

Base case
P(Leaf v):
mintree (Leaf v) = fold_mintree (Leaf v)

mintree (Leaf v)
=v by definition of mintree
=tfold (fun x->x) f (leaf v) by definition of tfold
=fold_mintree (Leaf v) by definition of fold_mintree

Inductive case:
P(Branch (t1,t2)):
show : mintree (Branch (t1, t2))=fold_mintree (Brach (t1, t2))
given: mintree t1=fold_mintree t1
given: mintree t2=fold_mintree t2

mintree (Branch (t1, t2))
=min (mintree t1) (mintree t2) by definition of mintree
=min (fold_mintree t1) (fold_mintree t2) by inductive hypothesis
=min (tfold fun x->x min t1) (tfold fun x->x min t2) by definition of fold_mintree
=tfold (fun x->x) min Brach (t1, t2) by definition of tfold
=fold_mintree t by definition of fold_mintree