COMP3007A(Fall 2018) – "Programming Paradigms"
Sample Solution for Haskell Structural Induction

Question 1:

Base Case : nothingSpecial [] = myFoldr (myFilter [] even) (+) 0

LHS : nothingSpecial []
= 0 [nsb]

RHS : myFoldr (myFilter [] even) (+) 0
= myFoldr [] (+) 0 [ftb]
= 0 [frb]

Inductive Assumption :
nothingSpecial t = myFoldr (myFilter t even) (+) 0 [ia]

Inductive Case:
nothingSpecial(h:t) = myFoldr (myFilter (h:t) even) (+) 0

Case 1 : (even h) == True

LHS: nothingSpecial (h:t)
= (nothingSpecial t) + h [nsr1]

RHS: myFoldr (myFilter (h:t) even) (+) 0
= myFoldr h:(myFilter t even) (+) 0 [ftr1]
= (+) h (myFoldr (myFilter t even) (+) 0) [frr]
= (+) h (nothingSpecial t) [ia]
= (nothingSpecial t) + h

Case 2 : (even h) == False

LHS: nothingSpecial (h:t)
= (nothingSpecial t) [nsr2]

RHS: myFoldr (myFilter (h:t) even) (+) 0
myFoldr (myFilter t even) (+) 0 [ftr2]

<u>**COMP3007A(Fall 2018) – "Programming Paradigms"**</u>
Sample Solution for Haskell Structural Induction

Question 2:

Base Case : when the list has only one element
myHead (myReverse [e]) = myLast [e]

LHS : myHead (myReverse [e])
= myHead ((myReverse[]) ++ [e]) [rer]
= myHead ([] ++ [e]) [reb]
= myHead [e]
= e [hdb]

RHS : myLast [e]
= e [ltb]

Inductive Assumption :
myHead (myReverse t) = (myLast t)

Inductive Case:
myHead (myReverse (h:t)) = (myLast (h:t))

LHS : myHead (myReverse (h:t))
= myHead ((myReverse t) ++ [h]) [rer]
= myHead (myReverse t) [hdb]

RHS : myLast (h:t)
= myLast t [lth]

By Inductive Assumption, we prove that
myHead (myReverse (h:t)) = (myLast (h:t))

COMP3007A(Fall 2018) – "Programming Paradigms"
Sample Solution for Haskell Structural Induction

Question 3:

Base Case 1 : when there is only one element in the list
And n is 0
myTake 0 (myDrop 0 [x]) = [elementAt 0 [x]]

LHS : myTake 0 (myDrop 0 [x])
= myTake 0 [x] [dpb]
= [x] [teb]

RHS : [elementAt 0 [x]]
= [x] [eab]

Base Case 2 : when there are more than one elements in the list
And n is 0
myTake 0 (myDrop 0 (h:t)) = [elementAt 0 (h:t)]

LHS : myTake 0 (myDrop 0 (h:t))
= myTake 0 (h:t) [dpb]
= [h] [teb]

RHS : [elementAt 0 (h:t)]
= [h] [eab]

Inductive Assumption :
myTake 0 (myDrop n t) = elementAt n t

Inductive Case:
myTake 0 (myDrop n (h:t)) = elementAt n (h:t)

LHS : myTake 0 (myDrop n (h:t))
= myTake 0 (myDrop (n - 1) t) [dpr]

RHS : elementAt n (h:t)
= elementAt (n - 1) t [ear]

By Inductive Assumption, we prove that
myHead (myReverse (h:t)) = (myLast (h:t))

Question 4:

Base Case : when there is only one element in the list

myLast [e] = elementAt ((myLength [e]) - 1) [e]

LHS : myLast [e]

= e [ltb]

RHS : elementAt ((myLength [e]) - 1) [e]

= elementAt (((myLength []) + 1) - 1) [e] [lhr]

= elementAt ((o + 1) - 1) [e] [lhb]

= elementAt 0 [e]

= e [eab]

Inductive Assumption :

myLast t = elementAt ((myLength t) - 1) t

Inductive Case:

myLast (h:t) = elementAt ((myLength (h:t)) - 1) (h:t)

LHS : myLast (h:t)

= myLast t [ltr]

RHS : elementAt ((myLength (h:t)) - 1) (h:t)

= elementAt (((myLength t) + 1) – 1) (h:t)[lhr]

= elementAt (myLength t) (h:t)

= elementAt ((myLength t) – 1) t [ear]

By Inductive Assumption, we prove that

myLast (h:t) = elementAt ((myLength (h:t)) - 1) (h:t)

COMP3007A(Fall 2018) – "Programming Paradigms"
Sample Solution for Haskell Structural Induction

Question 5:

Base Case : count NothingNode >= height NothingNode

LHS : count NothingNode
= 1 [ctb]

RHS : height NothingNode
= 1 [htb]

Inductive Assumption :
count a >= height a [ia1]
count b >= height b [ia2]
Inductive Case:
count (SomethingNode a b) >= height (SomethingNode a b)

LHS : count (Something a b)
= (count a) + (count b) [ctr]
>= (height a) + (count b) [ia1]
>= (height a) + (height b) [ia2]

RHS : height (Something a b)
= (max (height a) (height b)) + 1 [htr]

Case 1 : height a > height b

LHS : (height a) + (height b)
>= (height a) + 1 [base case]

RHS : (max (height a) (height b)) + 1
= (height a) + 1 [mx1]

Case 2 : height a < height b

LHS : (height a) + (height b)
>= 1 + (height b) [base case]
= (height b) + 1

RHS : (max (height a) (height b)) + 1
= (height b) + 1 [mx2]