HW02

BY ZIHAN ZHOU

1

2

2.1

$$\begin{cases} \text{isize(iEmpty)} & = & 0 \\ \text{isize(iLeaf)} & = & 1 \\ \text{isize}(\text{iNode}(t_L, i, t_R)) & = & \text{isize}(t_L) + \text{isize}(t_R) \end{cases}$$

2.2

$$\begin{cases} \text{validate(iEmpty)} & = & \text{True} \\ \text{validate(iLeaf)} & = & \text{True} \\ \text{validate}(\text{iNode}(t_L, i, t_R)) & = & \text{validate}(t_L) \,\text{and}\, \text{validate}(t_R) \,\text{and}\, i == \text{isize}(t_R) - \text{isize}(t_L) \end{cases}$$

2.3

$$\begin{cases} \text{isize}'(\text{iEmpty}) & = & 0 \\ \text{isize}'(\text{iLeaf}) & = & 1 \\ \text{isize}'(\text{iNode}(t_L, i, t_R)) & = & \text{isize}'(t_L) \times 2 + i \end{cases}$$

2.4

$$\begin{cases} \text{tiltLeft(iEmpty)} & = & \text{iEmpty} \\ \text{tiltLeft(iLeaf)} & = & \text{iLeaf} \\ \text{tiltLeft}(\text{iNode}(t_L, i, t_R)) & = & \begin{cases} \text{if } i \leqslant 0 & : & \text{iNode}(\text{tiltLeft}(t_L), i, \text{tiltLeft}(t_R)) \\ \text{if } i > 0 & : & \text{iNode}(\text{tiltLeft}(t_R), -i, \text{tiltLeft}(t_L)) \end{cases} \end{cases}$$

2.5

Proposition 1. *. For all t ∈ TT, if validate(t) = true , then iSize' (t) = iSize(t).*

Proof. by tree induction on t.

There are three cases: *iempty, ileaf, inode.*

Base case *t=iempty*:

to shhow: *if validate(iempty), then iSzize'(iempty)=iSize(iempty)*

by definition we know that no matter what condition: $\text{isize(imepty)} = 0 = \text{iszei}'(\text{iempty})$

Base case2 *t=iLeaf:*

to shhow: *if validate(ileaf), then iSzize'(ileaf)=iSize(ileaf)*

same, by def, we know it is always true.

Inductive case *t=iNode*(t_L, i, t_R)*:*

to show if $\mathit{validate}(\text{iNode}(t_L, i, t_R))$ *then* $\mathit{iSize}'(\text{iNode}(t_L, i, t_R)) = \mathit{iSize}(\text{iNode}(t_L, i, t_R))$

our IH is the proposition 1 holds for t_L and t_R

suppose validate(iNode(t_L, i, t_R))

$$\begin{aligned}
\text{iSize}'(\text{iNode}(t_L, i, t_R)) &= \text{isize}'(t_L) \times 2 + i && \text{by def of isze'} \\
&= \text{isize}'(t_L) \times 2 + \text{isize}(t_R) - \text{isize}(t_L) && \text{by def validate} \\
&= \text{isize}(t_L) \times 2 + \text{isize}(t_R) - \text{isize}(t_L) && \text{by IH} \\
&= \text{isize}(t_L) + \text{isize}(t_R) && \text{by math} \\
&= \text{isize}(\text{iNode}(t_L, i, t_R)) && \text{by def if isze}
\end{aligned}$$

□

2.6

Proposition 2. *For all t $\in TT$, if validate(t) = true , then validate(tiltLeft(t)) = true .*

Lemma 3. *For all t $\in TT$, iSize(tiltLeft(t)) = iSize(t).*

Proof. by tree induction on t

There are three cases: *iempty, ileaf, inode.*

Base case *t=iempty*:

to show if validate(iempty) = true , then validate(tiltLeft(iempty)) = true

by def, validate(tiltLeft(iempty))=validate(iempty) which is always true

Base case2 *t=iLeaf:*

to show if validate(ileaf) = true , then validate(tiltLeft(ileaf)) = true

same way as base case 1, this is true.

Inductive case *t=iNode*(t_L, i, t_R)*:*

IH: proposition 2 holds for tR and tL

to show if validate(iNode(t_L, i, t_R)) = true , then validate(tiltLeft(iNode(t_L, i, t_R))) = true

suppose validate(iNode(t_L, i, t_R))

two inductive case, if i<=0 or i>0

i<=0:

$$\begin{aligned}
&\text{validate}(\text{tiltLeft}(\text{iNode}(t_L, i, t_R))) \\
&=\text{validate}(\text{iNode}(\text{tiltLeft}(t_L), i, \text{tiltLeft}(t_R)) && \text{def of tilt} \\
&=\text{validate}(\text{tiltLeft}(t_L)) \text{ and validate}(\text{tiltLeft}(t_R)) \text{ and } i == \text{isize}(t_R) - \text{isize}(t_L) && \text{def of vali,lema3} \\
&=\text{True and True and } i == \text{isize}(t_R) - \text{isize}(t_L) && \text{IH} \\
&=\text{True and True and True} && \text{def of vali}
\end{aligned}$$

i>0:

$$\begin{aligned}
&\text{validate}(\text{tiltLeft}(\text{iNode}(t_L, i, t_R))) \\
&=\text{validate}(\text{iNode}(\text{tiltLeft}(t_R), -i, \text{tiltLeft}(t_L)) && \text{def of tilt} \\
&=\text{validate}(\text{tiltLeft}(t_R)) \text{ and validate}(\text{tiltLeft}(t_L)) \text{ and } -i == \text{isize}(t_L) - \text{isize}(t_R) && \text{def of vali,lema3} \\
&=\text{True and True and -}i == \text{isize}(t_L) - \text{isize}(t_R) && \text{IH} \\
&=\text{True and True and True} && \text{def of vali}
\end{aligned}$$

□

3

3.1

Yes they all are.

for $\mathbb{I}, \mathbb{P}$, they have $\mathbb{Z}$ in their def, since Z is inducitive domain, then they are inductive doman.

for $\mathbb{O}$, it can be north in O, O turn right still in O.

for $\mathbb{L}_1$, empty insttruction is in it, and any L1 elem with 1 more instruction is also in.

3.2

$$\text{turnRight}((x,y,o)) = \begin{cases} \text{if } o = \text{north} & (x,y,\text{east}) \\ \text{if } o = \text{east} & (x,y,\text{south}) \\ \text{if } o = \text{south} & (x,y,\text{west}) \\ \text{if } o = \text{west} & (x,y,\text{north}) \end{cases}$$

3.3

$$\text{move}(n,(x,y,o)) = \begin{cases} \text{if } o = \text{north} & (x,y+n,\text{north}) \\ \text{if } o = \text{east} & (x+n,y,\text{east}) \\ \text{if } o = \text{south} & (x,y-n,\text{south}) \\ \text{if } o = \text{west} & (x-n,y,\text{west}) \end{cases}$$

3.4

$$\begin{cases} \text{getPosition}\,([\,],(x,y,o)) = (x,y,o) \\ \text{getPosition}\,(R \circ l,(x,y,o)) = \text{getPosition}(l,\text{turnRight}((x,y,o))) \\ \text{getPosition}\,(sn \circ l,(x,y,0)) = \text{getPosition}(l,\text{move}(n,(x,y,o))) \end{cases}$$

3.5

Proposition 4. *For all l1, l2 ∈ LI and for all p ∈ P,*

getPosition(l1∘l2, p) = getPosition(l2, getPosition(l1, p))

Proof. by list induction on l_1

3 cases: one for l_1 is [] , two for l_1 is not [ ]

Base case: l_1 is [ ]

to show: getPosition($[\,]\circ l_2$, p) = getPosition(l_2, getPosition([ ], p))

by def, getPosition(l_2, getPosition([ ], p))=getPosition(l_2, p)=getPosition($[\,]\circ l_2$, p)

Inductive case 1 $l_1 = R\circ l_1'$

to show :getPosition($R\circ l_1'\circ l_2$, p) = getPosition(l_2, getPosition($R\circ l_1'$, p))

IH: getPosition($l_1'\circ l_2$, p) = getPosition(l_2, getPosition(l_1', p))

$$\begin{array}{lcll} \text{getPosition}(R\circ l_1'\circ l_2,\ \text{p}) & = & \text{getPosition}(l_1'\circ l_2, \text{turnRight}(p)) & \text{by def} \\ & = & \text{getPosition}(l_2,\ \text{getPosition}(l_1',\ \text{turnRight}(p))) & \text{by IH} \\ & = & \text{getPosition}(l_2,\ \text{getPosition}(R\circ l_1',\ \text{p})) & \text{by def} \end{array}$$

Inductive case 2 $l_1 = sn\circ l_1'$

to show :getPosition($sn\circ l_1'\circ l_2$, p) = getPosition(l_2, getPosition($sn\circ l_1'$, p))

IH: getPosition($l_1' \circ l_2$, p) = getPosition(l_2, getPosition(l_1', p))

$$\begin{array}{lcll} \text{getPosition}(sn \circ l_1' \circ l_2,\ \text{p}) & = & \text{getPosition}(l_1' \circ l_2, \text{move}(n,p)) & \text{by def} \\ & = & \text{getPosition}(l_2,\ \text{getPosition}(l_1',\ \text{move}(n,p))) & \text{by IH} \\ & = & \text{getPosition}(l_2,\ \text{getPosition}(sn \circ l_1',\ \text{p})) & \text{by def} \end{array}$$

□

3.6

$$\begin{cases} \text{goback}([\,]) & = [\,] \\ \text{goback}(R \circ l) & = \text{goback}(l) \circ [R, R, R] \\ \text{goback}(sn \circ l) & = \text{goback}(l) \circ s{-}n \end{cases}$$

3.7

Proposition 5. *For all $l \in L$ and $p \in P$*

getPosition($l \circ$ (goBack(l)), p)$=p$

Proof. by list induction on l

3 cases: one for l_1 is [] , two for l_1 is not [ ]

Base case: l is [ ]

to show getPosition([ ] $\circ$ (goBack([ ])), p)$=p$

getPosition([ ] $\circ$ (goBack([ ])), p)=getPosition([ ] $\circ$ [ ], p) by def

$= p$ by def

Inductive case 1 l=$sn \circ l'$

to show getPosition($sn \circ l' \circ$ (goBack($sn \circ l'$)), p)$=p$

IH getPosition($l' \circ$ (goBack(l')), p)$=p$

$$\begin{array}{cll} & \text{getPosition}(sn \circ l' \circ (\text{goBack}(sn \circ l')), p) & \\ = & \text{getPosition}(l' \circ (\text{goBack}(sn \circ l')), \text{move}(n,p)) & \text{by def of getPos} \\ = & \text{getPosition}(l' \circ (\text{goback}(l') \circ s{-}n), \text{move}(n,p)) & \text{by def of goback} \\ = & \text{getPosition}((l' \circ \text{goback}(l')) \circ s{-}n, \text{move}(n,p)) & \text{assoiciativty} \\ = & \text{getPosition}(s{-}n, \text{getPosition}((l' \circ \text{goback}(l')), \text{move}(n,p))) & \text{by 3.6} \\ = & \text{getPosition}(s{-}n, \text{move}(n,p)) & \text{IH} \\ = & \text{p} & \text{back forth} \end{array}$$

Inductive case 12 l =$R \circ l'$

to show getPosition($R \circ l' \circ$ (goBack($R \circ l'$)), p)$=p$

IH getPosition($l' \circ$ (goBack(l')), p)$=p$

$$\begin{array}{cll} & \text{getPosition}(R \circ l' \circ (\text{goBack}(R \circ l')), p) & \\ = & \text{getPosition}(l' \circ (\text{goBack}(R \circ l')), \text{turnRight}(p)) & \text{by def of getPos} \\ = & \text{getPosition}(l' \circ (\text{goback}(l') \circ [R,R,R]), \text{turnRight}(p)) & \text{by def of goback} \\ = & \text{getPosition}((l' \circ \text{goback}(l')) \circ [R,R,R], \text{turnRight}(p)) & \text{assoiciativty} \\ = & \text{getPosition}([R,R,R], \text{getPosition}((l' \circ \text{goback}(l')), \text{turnRight}(p))) & \text{by 3.6} \\ = & \text{getPosition}([R,R,R],\ \text{turnRight}(p)) & \text{IH} \\ = & \text{p} & \text{by 360} \end{array}$$

□