CS 313 Lecture 14

Smalltalk – defining classes cont'd
IntList
Linked list, functional view

\[ a = [1, 2, 3] \]

\[ b = [] \]

• List = pointer to first element
• empty list: \texttt{nil}
• pass lists as params: \texttt{length(a), isEmpty(b)}
Linked list, OO view

x := IntList new.
x isEmpty ->
x add: 3. x add: 5.
x first ->
x remove ->
x remove ->

• List is object that modifies itself
• Can't use nil for empty list
• Need separate classes for list and elements
IntListElt

- helper class only used within IntList
- Instance variables: `val`, `next`
- provide accessor methods ("setters & getters")

\[
\begin{align*}
\text{val} & : \text{anInt} \\
\text{next} & : \text{anIntListElt}
\end{align*}
\]

\[
\begin{align*}
\text{val} & \rightarrow \text{anInt} \\
\text{next} & \rightarrow \text{anIntListElt}
\end{align*}
\]

d val \rightarrow

d next val \rightarrow
IntList

• Instance variable: head

• Instance methods:
  isEmpty
    "returns whether list is empty"
  first
    "returns first integer in the list"
    "assert: list not empty"
  add: anInt
    "inserts anInt at the head of the list"
  remove
    "removes first element in the list and returns its value"
    "assert: list not empty"
IntList

• More instance methods:

  length
  "returns length of list"

  lengthR
  "returns length of list (recursive version)"

  max
  "returns maximum value in list"
  "assert: list not empty"

  maxR
  "returns maximum value in list (recursive version)"
  "assert: list not empty"

  do: aBlock
  "execute aBlock for all elements in list"