

Chapter 17 Programming

- No changes were made to Node.java

Method: insertEnd

```
123
124     /**
125      * Insert at the end
126      * @param x the item to insert.
127      */
128     public void insertEnd(AnyType _x)
129     {
130         // Use previous to keep the Last node
131         Node<AnyType> previous = this.list;
132         // This is the start of the chain
133         Node<AnyType> current = this.list.getNextNode();
134
135         // Iterate though the nodes
136         while (current != null)
137         {
138             previous = current;
139             current = current.getNextNode();
140         }
141
142         // Attach the new node to the last node's pointer
143         Node<AnyType> last = new Node<AnyType>(_x);
144         previous.setNextNode(last);
145     }
```

Method: removeLast

```
147
148     /**
149      * Remove the Last occurrence of an item.
150      * @param x the item to remove.
151      */
152     public void removeLast(AnyType _x)
153     {
154         Node<AnyType> matchBefore = null;
155         Node<AnyType> matchAfter = null;
156
157         // Use previous to keep the Last node
158         Node<AnyType> previous = this.list;
159         // This is the start of the chain
160         Node<AnyType> current = this.list.getNextNode();
161
162         // Iterate though the all the nodes
163         while (current != null)
164         {
165             // Check the node for the value
166             if (current.getItem() == _x)
167             {
168                 // Retain the previous and next nodes
169                 matchBefore = previous;
170                 matchAfter = current;
171             }
172             previous = current;
173             current = current.getNextNode();
174         }
175
176         // Remove the node between the Last previous and next nodes
177         if (matchBefore != null && matchAfter != null)
178         {
179             matchBefore.setNextNode(matchAfter.getNextNode());
180         }
```

Chapter 17 Programming

Method: removeAll

```
182  /**
183   * Remove all occurrences of an item.
184   * @param x the item to remove.
185   */
186  public void removeAll(AnyType _x)
187  {
188      // Use previous to keep the last node
189      Node<AnyType> previous = this.list;
190      // This is the start of the chain
191      Node<AnyType> current = this.list.getNextNode();
192
193      // Iterate though the all the nodes
194      while (current != null)
195      {
196          // Remove every node that matches the value
197          if (current.getItem() == _x)
198          {
199              previous.setNextNode(current.getNextNode());
200          }
201          previous = current;
202          current = current.getNextNode();
203      }
204  }
```

Method: previous

```
206  /**
207   * Return Node just before the first node containing an item.
208   * @param x the item to search for.
209   * @return a Node; node is not valid if item is not found, or is the
210   * first item in the list.
211   */
212
213  public Node<AnyType> previous(AnyType _x)
214  {
215      // Use previous to keep the last node
216      Node<AnyType> previous = null;
217      // This is the start of the chain
218      Node<AnyType> current = this.list.getNextNode();
219
220      // Iterate though the nodes until one matches the value
221      while (current != null)
222      {
223          if (current.getItem() == _x)
224          {
225              break;
226          }
227          previous = current;
228          current = current.getNextNode();
229      }
230
231      return previous;
232  }
```

Chapter 17 Programming

Method: findAndMoveToFront

```
239  /**
240   * Return Node corresponding to the first node containing an item.
241   * @param x the item to search for.
242   * @return a Node; node is not valid if item is not found.
243   *
244   * As a by product, the found node is moved to the front of the list
245   * according to the move to front heuristic of 17.17
246   */
247  public Node<AnyType> findAndMoveToFront(AnyType _x)
248  {
249      // First use the find method to get the node with the value
250      Node<AnyType> found = this.find(_x);
251      // If we found a match use the remove method, then the insertFront method
252      if (found != null)
253      {
254          this.remove(_x);
255          this.insertFront(_x);
256      }
257
258      return found;
259  }
```

Testing Output

```
C:\ c:\Users\wes\github-repos\cs2420_summer2023\Chapter17\Programming - VS Code Console
Testing LinkedList methods
Finished LinkedList testing
Press any key to continue . . . _
```