**LinkedList + Iterator Class**

**Class Node**

|  |
| --- |
| **class** Node<E>  {  **public** E value;  **public** Node next;    **public** Node(E val, Node n)  {  value = val;  next = n;  }    **public** Node(E val)  {  **this**(val, **null**);  }  } |

**Class LinkedList**

|  |
| --- |
| **Class** LinkedList<E>  {  **private** Node<E> first;  **private** Node<E> last;    **public** LinkedList()  {  first = **null**;  last = **null**;  }    **public** **void** add(E element)  {  **if**(isEmpty())  {  first = **new** Node(element);  last = first;  }  **else**  {  last.next = **new** Node(element);  last = last.next;  }  }    **public** **void** add(**int** index, E element)  {  **if**(index < 0 || index > size())  **throw** **new** IndexOutOfBoundsException();  **if**(index == 0)  {  first = **new** Node(element, first);  **if**(last == **null**)  last = first;  }  **else**  {  Node pred = first;  **for**(**int** i = 1; i <= index - 1; i++)  pred = pred.next;  pred.next = **new** Node(element, pred.next);  **if**(pred.next.next == **null**)  last = pred.next;  }  }  **public** E remove(**int** index)  {  E temp;  **if**(index < 0 || index > size())  **throw** **new** IndexOutOfBoundsException();  **else** **if**(index == 0)  {  temp = first.value;  first = first.next;  }  **else**  {  Node pred = first;  **for**(**int** i = 1; i <= index - 1; i++)  pred = pred.next;  temp = (E) pred.next.value;  pred.next = pred.next.next;  **if**(pred.next == **null**)  last = pred;  }  **return** temp;  }  **public** **boolean** remove(E element)  {  **if**(isEmpty())  **return** **false**;  **if**(first.value.equals(element))  {  first = first.next;  **if**(first == **null**)  last = **null**;  **return** **true**;  }  **else**  {  Node pred;  **for**(pred = first; pred.next != **null** && !pred.next.value.equals(element); pred = pred.next)  pred = pred.next;  **if**(pred.next == **null**)  **return** **false**;  pred.next = pred.next.next;  **if**(pred.next == **null**)  last = pred;  **return** **true**;  }  }  **public** **void** clear()  {  **while**(!isEmpty())  remove(0);  }    **public** **boolean** isEmpty()  {  **return** (first == **null**);  }    **public** **int** size()  {  **int** count = 0;  **for**(Node ref = first; ref != **null**; ref = ref.next)  count++;  **return** count;  }  **public** E get(**int** index)  {  **if**(index < 0 || index > size())  **throw** **new** IndexOutOfBoundsException();  Node pred = first;  **for**(**int** i = 0; i <= index - 1; i++, pred = pred.next)  **if**(i == index)  **break**;  **return** (E) pred.value;  }    **public** String toString()  {  StringBuilder builder = **new** StringBuilder();  **for**(Node ref = first; ref != **null**; ref = ref.next)  builder.append(ref.value + " ");  **return** builder.toString();  }  **public** Iterator<E> getIterator()  {  **return** **new** Iterator<E>(**this**);  }  } |

**Iterator Class**

|  |
| --- |
| **class** Iterator<E>  {  **private** LinkedList<E> list;  **private** **boolean** canRemove;  **private** **int** previous;    **public** Iterator(LinkedList<E> aList)  {  list = aList;  previous = -1;  canRemove = **false**;  }    **public** **boolean** hasNext()  {  **return**(previous + 1 < list.size());  }    **public** E next()  {  **if**(!hasNext())  **return** **null**;  **else**  {  previous++;  canRemove = **true**;  **return** list.get(previous);  }  }    **public** **void** remove()  {  **if**(!canRemove)  **return**;  **else**  {  list.remove(previous);  previous--;  canRemove = **false**;  }  }  } |

**The main Class**

|  |
| --- |
| **public** **class** LinkedLinkApp  {  **public** **static** **void** main(String[] args)  {  LinkedList<String> myList = **new** LinkedList<String>();  String[] names = {"One", "Two", "Three", "Four"};  **for**(String s : names) myList.add(s);  //System.out.println(myList); // One Two Three Four    myList.add(0, "Beginning"); myList.add(3, "Middle"); myList.add(6, "End");  //System.out.println(myList); // Beginning One Two Middle Three Four End  //myList.clear();  //myList.remove(myList.size() - 1); // removes the last element    myList.remove("Middle");  //myList.remove("End");  System.*out*.println(myList);    //System.out.println(myList.get(1));    Iterator<String> iterator1 = myList.getIterator();  Iterator<String> iterator2 = myList.getIterator();    **while**(iterator2.hasNext())  **if**(iterator2.next().compareTo("End") == 0)  iterator2.remove();    **while**(iterator1.hasNext())  System.*out*.print(iterator1.next() + " ");  System.*out*.println();  }  } |