```java
import java.util.*;
// reversing a singly linked list

class Node{

    private int data;
    private Node nextNodePtr;


    public Node(){}

    public void setData(int d){
            data = d;
    }

    public  int getData(){
            return data;
    }

    public  void setNextNodePtr(Node nodePtr){
            nextNodePtr = nodePtr;
    }

    public  Node getNextNodePtr(){
        return nextNodePtr;
    }

}

class List{

    private Node headPtr;


    public  List(){
        headPtr = new Node();
        headPtr.setNextNodePtr(null);
    }


    public Node getHeadPtr(){
        return headPtr;
    }

    public  boolean isEmpty(){

        if (headPtr.getNextNodePtr() == null)
            return true;

        return false;
    }


    public  void insert(int data){

            Node currentNodePtr = headPtr.getNextNodePtr();
            Node prevNodePtr = headPtr;

            while (currentNodePtr != null){
                prevNodePtr = currentNodePtr;
                currentNodePtr = currentNodePtr.getNextNodePtr();
            }

            Node newNodePtr = new Node();
```

```
65              newNodePtr.setData(data);
66              newNodePtr.setNextNodePtr(null);
67              prevNodePtr.setNextNodePtr(newNodePtr);
68
69      }
70
71      public  void insertAtIndex(int insertIndex, int data){
72
73              Node currentNodePtr = headPtr.getNextNodePtr();
74              Node prevNodePtr = headPtr;
75
76              int index = 0;
77
78              while (currentNodePtr != null){
79
80                  if (index == insertIndex)
81                      break;
82
83                  prevNodePtr = currentNodePtr;
84                  currentNodePtr = currentNodePtr.getNextNodePtr();
85                  index++;
86              }
87
88              Node newNodePtr = new Node();
89              newNodePtr.setData(data);
90              newNodePtr.setNextNodePtr(currentNodePtr);
91              prevNodePtr.setNextNodePtr(newNodePtr);
92
93      }
94
95
96
97      public  void IterativePrint(){
98
99              Node currentNodePtr = headPtr.getNextNodePtr();
100
101             while (currentNodePtr != null){
102                 System.out.print(currentNodePtr.getData()+" ");
103                 currentNodePtr = currentNodePtr.getNextNodePtr();
104             }
105
106             System.out.println();
107
108     }
109
110
111     public  void reverseList(){
112
113             Node currentNodePtr = headPtr.getNextNodePtr();
114             Node prevNodePtr = null;
115             Node nextNodePtr = currentNodePtr;
116
117             while (currentNodePtr != null){
118
119                 nextNodePtr = currentNodePtr.getNextNodePtr(); // Step 1
120                 currentNodePtr.setNextNodePtr(prevNodePtr); // Step 2
121                 prevNodePtr = currentNodePtr;  // Step 3
122                 currentNodePtr = nextNodePtr;  // Step 4
123
124             }
125
126             headPtr.setNextNodePtr(prevNodePtr);
127
128         }
```

```
129
130
131
132  }
133
134  class ReverseSinglyLinkedList{
135
136      public static void main(String[] args){
137
138      Scanner input = new Scanner(System.in);
139
140      int listSize;
141      System.out.print("Enter the number of elements you want to insert: ");
142      listSize = input.nextInt();
143
144      List integerList = new List(); // Create an empty list
145
146      int maxValue;
147      System.out.print("Enter the maximum value for an element: ");
148      maxValue = input.nextInt();
149
150      Random randGen = new Random(System.currentTimeMillis());
151
152      for (int i = 0; i < listSize; i++){
153
154          int value = randGen.nextInt(maxValue);
155
156          integerList.insertAtIndex(i, value);
157      }
158
159      System.out.print("Contents of the List (before reversal): ");
160      integerList.IterativePrint();
161
162      integerList.reverseList();
163
164      System.out.print("Contents of the List (after reversal): ");
165      integerList.IterativePrint();
166
167      }
168
169  }
```

```
Enter the number of elements you want to insert: 10
Enter the maximum value for an element: 50
Contents of the List (before reversal): 24 41 1 11 13 11 46 8 37 31
Contents of the List (after reversal): 31 37 8 46 11 13 11 1 41 24
```