```
::::::::::::::
Lab3Status.txt
::::::::::::::
Problem 1: compiles, runs correctly on all provided input

Problem 2: compiles, runs correctly on all provided input

Problem 3: compiles, runs correctly on all provided input::::::::::::::
Lab3Conclusions.txt
::::::::::::::
My biggest takeaway from this lab was the ability to create a collection without t
he usage of official collections such as ArrayList, etc. Despite this lab requirin
g an Array, the overall collection was only changed by the Array, not maintained b
y it. I also learned the importance of the compareTo method as it is clear it can
be used for a variety of things in programming.::::::::::::::
Node.java
::::::::::::::
//please note that this code is different from the textbook code, because the data
 is encapsulated!

public class Node
{
    private Object item;
    private Node next;

    public Node(Object newItem)
    {
        item = newItem;
        next = null;
    } // end constructor

    public Node(Object newItem, Node nextNode)
    {
        item = newItem;
        next = nextNode;
    } // end constructor

    public void setItem(Object newItem)
    {
        item = newItem;
    } // end setItem

    public Object getItem()
    {
        return item;
    } // end getItem

    public void setNext(Node nextNode)
    {
        next = nextNode;
    } // end setNext

    public Node getNext()
    {
        return next;
    } // end getNext

    public String toString()
    {
        return item+" ";
    } //end toString
} // end class Node::::::::::::::
ListInterface.java
::::::::::::::
// ********************************************************
// Interface ListInterface for the ADT list.
// *********************************************************
public interface ListInterface
{
    boolean isEmpty();
    int size();
    void add(int index, Object item) throws ListIndexOutOfBoundsException;
    Object get(int index) throws ListIndexOutOfBoundsException;
    Object remove(int index) throws ListIndexOutOfBoundsException;
    void removeAll();
    String toString();
}  // end ListInterface::::::::::::::
ListIndexOutOfBoundsException.java
::::::::::::::
public class ListIndexOutOfBoundsException
    extends IndexOutOfBoundsException
{
    public ListIndexOutOfBoundsException(String s)
    {
        super(s);
    }  // end constructor
}  // end ListIndexOutOfBoundsException
::::::::::::::
ListReferencedBased.java
::::::::::::::
// Please note that this code is slightly different from the textbook code
//to reflect the fact that the Node class is implemented using data encapsulation


// ****************************************************
// Reference-based implementation of ADT list.
// ****************************************************
public class ListReferencedBased implements ListInterface
{
    // reference to linked list of items
    private Node head;
    private int numItems; // number of items in list

    public ListReferencedBased()
    {
        numItems = 0;
        head = null;
    }  // end default constructor

    public boolean isEmpty()
    {
        return numItems == 0;
    }  // end isEmpty

    public int size()
    {
        return numItems;
    }  // end size

    private Node find(int index)
    {
        // --------------------------------------------------
        // Locates a specified node in a linked list.
        // Precondition: index is the number of the desired
    // node. Assumes that 0 <= index <= numItems
    // Postcondition: Returns a reference to the desired
    // node.
    // ----------------------------------------------------
    Node curr = head;
    for (int skip = 0; skip < index; skip++)
    {
        curr = curr.getNext();
    } // end for
    return curr;
} // end find

public Object get(int index)
throws ListIndexOutOfBoundsException
{
    if (index >= 0 && index < numItems)
    {
        // get reference to node, then data in node
        Node curr = find(index);
        Object dataItem = curr.getItem();
        return dataItem;
    }
    else
    {
        throw new ListIndexOutOfBoundsException(
            "List index out of bounds exception on get");
    } // end if
} // end get

public void add(int index, Object item)
throws ListIndexOutOfBoundsException
{
    if (index >= 0 && index < numItems+1)
    {
        if (index == 0)
        {
            // insert the new node containing item at
            // beginning of list
            Node newNode = new Node(item, head);
            head = newNode;
        }
        else
        {
            Node prev = find(index-1);
            // insert the new node containing item after
            // the node that prev references
            Node newNode = new Node(item, prev.getNext());
            prev.setNext(newNode);
        } // end if
        numItems++;
    }
    else
    {
        throw new ListIndexOutOfBoundsException(
            "List index out of bounds exception on add");
    } // end if
}  // end add

public Object remove(int index)
throws ListIndexOutOfBoundsException
{   Object result;
    if (index >= 0 && index < numItems)
    {
        if (index == 0)
        {
            // delete the first node from the list
            result = head.getItem();
            head = head.getNext();
        }
        else
        {
            Node prev = find(index-1);
            // delete the node after the node that prev
            // references, save reference to node
            Node curr = prev.getNext();
            result = curr.getItem();
            prev.setNext(curr.getNext());
        } // end if
        numItems--;
    } // end if
    else
    {
        throw new ListIndexOutOfBoundsException(
            "List index out of bounds exception on remove");
    } // end if
    return result;
}   // end remove

public void removeAll()
{
    // setting head to null causes list to be
    // unreachable and thus marked for garbage
    // collection
    head = null;
    numItems = 0;
} // end removeAll

} // end ListReferenceBased::::::::::::::
MyListReferenceBased.java
::::::::::::::
/**
 * Purpose: Data Structure and Algorithms Lab 3
 * Status: Complete and thoroughly tested
 * Last update: 02/06/23
 * Submitted:  02/06/23
 * Comment: test suite and sample run attached
 * Comment: I declare that this is entirely my own work
 * @author: Antonio Rosado
 * @version: 2023.02.06
 */

public class MyListReferenceBased implements ListInterface
{
    private Node head;

    public MyListReferenceBased()
    {
        head = null;
    }

    /**
     * Get item from Node
     * @param int index     index of item
     * @return Object item  item from specified index
```

```
         */
        private Node find(int index) throws ListIndexOutOfBoundsException
        {
            Node curr = head;
            for(int i = 0; i < index; i++)
            {
                curr = curr.getNext();
            }
            return curr;
        }
        /**
         * Add item to Node
         * @param int index     index of item
         * @param Object item   item Object
         */
        public void add(int index, Object item) throws ListIndexOutOfBoundsException
        {
            if(index >= 0 && index < size() + 1)
            {
                if(index == 0)
                {
                    head = new Node(item, head);
                }

                else
                {
                    Node prev = find(index - 1);
                    Node temp = new Node(item, prev.getNext());
                    prev.setNext(temp);
                }
            }

            else
            {
                throw new ListIndexOutOfBoundsException("List index out of bounds on a
dd.");
            }
        }

        /**
         * Get item from Node
         * @param int index     index of item
         * @return Object item  item from specified index
         */
        public Object get(int index) throws ListIndexOutOfBoundsException
        {
            {
                if (index >= 0 && index < size())
                {
                    // get reference to node, then data in node
                    Node curr = find(index);
                    Object dataItem = curr.getItem();
                    return dataItem;
                }
                else
                {
                    throw new ListIndexOutOfBoundsException(
                        "List index out of bounds exception on get");
                } // end if
            } // end get
        }
```

```
        /**
        * Check if Node is empty
        * @return head == null
        */
        public boolean isEmpty()
        {
            return head == null;
        }

        /**
         * Remove item from Node
         * @param int index         index of item
         * @return Object result    item removed
         */
        public Object remove(int index)
        throws ListIndexOutOfBoundsException
        {   Object result;
            if (index >= 0 && index < size())
            {
                if (index == 0)
                {
                    // delete the first node from the list
                    result = head.getItem();
                    head = head.getNext();
                }
                else
                {
                    Node prev = find(index-1);
                    // delete the node after the node that prev
                    // references, save reference to node
                    Node curr = prev.getNext();
                    result = curr.getItem();
                    prev.setNext(curr.getNext());
                } // end if
            } // end if
            else
            {
                throw new ListIndexOutOfBoundsException(
                    "List index out of bounds exception on remove");
            } // end if
            return result;
        }   // end remove

        /**
         * Remove all items from Node
         */
        public void removeAll()
        {

            head = null;
        }

        /**
         * Return size of Node
         * @return size of Node
         */
        public int size()
        {
            int size = 0;
            Node curr = head;
            while (curr != null)
            {
```

```
            size++;
            curr = curr.getNext();

        }
        return size;
    }  // end size

    /**
     * Returns a string value of Node items
     */
    public String toString()
    {
        StringBuilder sb = new StringBuilder();
        Node curr = head;
        while(curr != null)
        {
            sb.append(curr.getItem().toString() + " ");
            curr = curr.getNext();
        }
        return sb.toString();
    }

}::::::::::::::
Lab3P2Driver.java
::::::::::::::
import java.io.IOException;
import java.io.BufferedReader;
import java.io.InputStreamReader;
public class Lab3P2Driver
{
    static BufferedReader stdin = new BufferedReader(new InputStreamReader(System.
in));
    public static void main (String[] args) throws IOException
    {
        MyListReferenceBased myList = new MyListReferenceBased();
        Lab3P2Driver driver = new Lab3P2Driver();
        boolean exit = false;
        int pos = -1;
        while (!exit) {
            System.out.println("Select from the following menu: \n"
                              + "\t 0. Exit the program \n"
                              + "\t 1. Insert item into the list \n"
                              + "\t 2. Remove item from the list \n"
                              + "\t 3. Get item from the list \n"
                              + "\t 4. Clear the list \n"
                              + "\t 5. Display size and content of the list \n"
                              + "\t 6. Delete the smallest and largest item in th
e list \n"
                              + "\t 7. Reverse the list \n");

            System.out.print("Make your menu selection now: " );
            int input = Integer.parseInt(stdin.readLine());
            System.out.println(input);
            // possible cases for initial input
            switch (input) {
            case 0:
                System.out.println("Exiting program... good bye");
                exit = true;
                break;

            case 1:
                System.out.println("You are now inserting an item into the list.")
;
                System.out.print("\t Enter item: ");
                Object item = stdin.readLine();
                System.out.println(item);

                System.out.print("\t Enter the position to insert item in: ");
                pos = Integer.parseInt(stdin.readLine());
                System.out.println(pos);
                if (pos <= myList.size())
                {
                    myList.add(pos, item);
                    System.out.println("Item " + item + " inserted in position " +
 pos + " in the list.");
                }

                else
                {
                    System.out.println("Position specified is out of range!");
                }
                break;

            case 2:
                System.out.println("You are now removing an item from the list.");

                System.out.print("\t Enter position to remove item from: ");
                pos = Integer.parseInt(stdin.readLine());
                System.out.println(pos);
                if(pos > myList.size() - 1)
                {
                    System.out.println("Position specified is out of range!");
                }

                else
                {
                    System.out.println("Item " + myList.get(pos) + " removed from
position " + pos + " in the list.");
                    myList.remove(pos);
                }
                break;

            case 3:
                System.out.print("\t Enter position to retrieve item from: ");
                pos = Integer.parseInt(stdin.readLine());
                System.out.println(pos);
                if(pos > myList.size())
                {
                    System.out.println("Position specified is out of range!");
                }

                else
                {
                    System.out.println("Item " + myList.get(pos) + " retrieved fro
m position " + pos + " in the list.");
                }
                break;

            case 4:
                if(myList.isEmpty())
                {
                    System.out.println("List is empty, nothing to clear!");
                }
                else
                {
                    System.out.println("Clearing list...");
                    myList.removeAll();
                    System.out.println("List cleared.");
                }
                break;

            case 5:
                if(myList.isEmpty())
                {
                    System.out.println("List is empty.");
                }

                else
                {
                    System.out.println("\t List of size " + myList.size() + " has
 the following items: " + myList.toString());
                }
                break;

            case 6:
                if(myList.isEmpty())
                {
                    System.out.println("List is empty, nothing to delete!");
                }

                else if(myList.size() == 1)
                {
                    System.out.println(myList.toString() + "is deleted.");
                }

                else
                {
                    driver.displayAndDeleteLargeAndSmall(myList);
                }
                break;

            case 7:
                driver.reverse(myList);
                break;

            }
        }
    }

    /**
     * Find largest and smallest items from Node collection lexicographically
     * @param MyListReferenceBased tempList     list to be iterated
     * @param int[] numbers                     Array of largest and smallest valu
es
     */
    public void findIndexLargeAndSmall(MyListReferenceBased myList, int[] numbers)
    {
        int size = myList.size();
        int smallIndex = 0;
        int largeIndex = 0;
        String curr = "";
        String smallestValue = myList.get(0).toString();
        String largestValue = myList.get(0).toString();
        for (int index = 0; index < size; index++)
        {
            curr = myList.get(index).toString();
            if(curr.compareTo(smallestValue) <= 0)
            {
                smallIndex = index;
                smallestValue = myList.get(smallIndex).toString();
            }

            else if(curr.compareTo(largestValue) >= 0)
            {
                largeIndex = index;
                largestValue = myList.get(largeIndex).toString();
            }
        }
        numbers[0] = smallIndex;
        numbers[1] = largeIndex;
    }

    /**
    * Display and delete largest and smallest items from Node collection lexicogra
phically
    * @param MyListReferenceBased myList     list to be iterated
    */
    public void displayAndDeleteLargeAndSmall(MyListReferenceBased myList)
    {

        int numbers[] = new int[2];
        findIndexLargeAndSmall(myList, numbers);
        System.out.println("Smallest item " + myList.get(numbers[0]) + " deleted."
);
        System.out.println("Largest item " + myList.get(numbers[1]) + " deleted.")
;
        try
        {
            myList.remove(numbers[0]);
            myList.remove(numbers[1]);
        }
        catch(ListIndexOutOfBoundsException e)
        {
            System.out.println("Index out of bounds.");
        }
    }

    /**
    * Display Node objects in reverse
    * @param MyListReferenceBased myList     list to be reversed
    */
    public MyListReferenceBased reverse(MyListReferenceBased myList)
    {
        MyListReferenceBased reversed = new MyListReferenceBased();
        if(!(myList.isEmpty()))
        {
            int size = myList.size();
            for(int index = 0; index < size; index++)
            {
                reversed.add(index, myList.get(size - index - 1));
            }
            System.out.println("\t Here is the content: " + reversed.toString());
        }

        else
        {
            System.out.println("List is empty... nothing to reverse!");
        }
        return reversed;
    }
}



::::::::::::::
compareTo.java
::::::::::::::
import java.io.IOException;
import java.io.BufferedReader;
import java.io.InputStreamReader;
/**
 * String class' compareTo method returns:
 * The value 0 if the argument string is equal to this string;
 * a value less than 0 if this string is lexicographically less
 * than the string argument; and a value greater than 0 if this
 * string is lexicographically greater than the string argument.
**/

public class compareTo
{
    static BufferedReader stdin = new BufferedReader(new InputStreamReader(System.
in));
    public static void main(String[] args) throws IOException
    {
        System.out.println("Some examples of compared strings are: ");
        String s1 = "a";
        String s2 = "A";
        String s3 = "t";
        String s4 = "T";
        String s5 = "z";
        String s6 = "l";
        String s7 = "1";
        String s8 = "9";
        String s9 = "18";
        String s10 = "21";
        String s11 = "81";
        String s12 = "to";
        String s13 = "top";
        String s14 = "%";
        String s15 = "0";

        /**
         * compareTo() compares strings lexicographically, with the returned value
 representing
         * each ASCII value's distance from each other.
         * A returned value is either positive or negative depending on whether th
e first string's
         * ASCII value comes before or after the second string's value.
         * 'a' has an ASCII value of 97, whereas 'A' has a value of 65, hence the
returned value of 32.
         **/
        System.out.println("'a' compared to 'A' is : " + s1.compareTo(s2));
        System.out.println("'t' compared to 'T' is : " + s3.compareTo(s4));
        System.out.println("'a' compared to 'z' is : " + s1.compareTo(s5));
        System.out.println("'a' compared to 'l' is : " + s1.compareTo(s6));
        System.out.println("'l' compared to 'A' is : " + s6.compareTo(s2));
        System.out.println("'1' compared to '9' is : " + s7.compareTo(s8));
        System.out.println("'1' compared to '18' is : " + s7.compareTo(s9));
        System.out.println("'21' compared to '81' is : " + s10.compareTo(s11));
        System.out.println("'to' compared to 'top' is : " + s12.compareTo(s13));
        System.out.println("'%' compared to '0' is : " + s14.compareTo(s15));

        //'a' compared to 'A' is : 32      'a'  is greater than 'A'.
        //'t' compared to 'T' is : 32      't'  is greater than 'T'.
        //'a' compared to 'z' is : -25     'a'  is smaller than 'z'.
        //'a' compared to 'l' is : -11     'a'  is smaller than 'l'.
        //'l' compared to 'A' is : 43      'A'  is greater than 'l'.
        //'1' compared to '9' is : -8      '9'  is smaller than '1'.
        //'1' compared to '18' is : -1     '18' is smaller than '1'.
        //'21' compared to '81' is : -6    '81' is smaller than '21'.
        //'to' compared to 'top' is : -1   'to' is smaller than 'top'.
        //'%' compared to '0' is : -11     '%'  is smaller than '-11'.

        boolean exit = false;
        String input = "";
        while (!exit)
        {
            System.out.println("Select from the following menu: \n"
                                   + "\t 0. Exit the program \n"
                                   + "\t 1. Compare two strings");

            input = stdin.readLine();
            System.out.println(input);
            switch (input) {
            case "0":
                System.out.println("Exiting program... good bye");
                exit = true;
                break;

            case "1":
                System.out.println("You are now comparing two strings...");
                System.out.print("\t Enter string 1: ");
                String string1 = stdin.readLine();
                System.out.println(string1);

                System.out.print("\t Enter string 2: ");
                String string2 = stdin.readLine();
                System.out.println(string2);

                System.out.println("'" + string1 + "'" + " compared to " + "'" + s
tring2 + "' " + "is: "  + string1.compareTo(string2));
                break;
            }

        }
    }

}::::::::::::::
Lab3P2Sampleruns.txt
::::::::::::::
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 5
```

```
List is empty.
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 6
List is empty, nothing to delete!
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 7
List is empty... nothing to reverse!
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 1
You are now inserting an item into the list.
         Enter item: Data
         Enter the position to insert item in: 0
Item Data inserted in position 0 in the list.
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 5
         List of size 1 has the following items: Data
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list
Make your menu selection now: 7
         Here is the content: Data
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 1
You are now inserting an item into the list.
         Enter item: Beverly
         Enter the position to insert item in: 0
Item Beverly inserted in position 0 in the list.
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 5
         List of size 2 has the following items: Beverly Data
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 1
You are now inserting an item into the list.
         Enter item: Jean-Luc
         Enter the position to insert item in: 5
Position specified is out of range!
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 5
         List of size 2 has the following items: Beverly Data
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 1
You are now inserting an item into the list.
         Enter item: Jean-Luc
         Enter the position to insert item in: 2
Item Jean-Luc inserted in position 2 in the list.
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 1
You are now inserting an item into the list.
         Enter item: Geordi
         Enter the position to insert item in: 2
Item Geordi inserted in position 2 in the list.
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 1
You are now inserting an item into the list.
         Enter item: Worf
         Enter the position to insert item in: 3
Item Worf inserted in position 3 in the list.
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 5
         List of size 5 has the following items: Beverly Data Geordi Worf Jean-Luc

Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 7
         Here is the content: Jean-Luc Worf Geordi Data Beverly
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 7
         Here is the content: Jean-Luc Worf Geordi Data Beverly
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 6
Smallest item Beverly deleted.
Largest item Worf deleted.
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 5
         List of size 3 has the following items: Data Geordi Worf
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 7
         Here is the content: Worf Geordi Data
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 7
```

```
         Here is the content: Worf Geordi Data
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 2
You are now removing an item from the list.
         Enter position to remove item from: 9
Position specified is out of range!
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 2
You are now removing an item from the list.
         Enter position to remove item from: 2
Item Worf removed from position 2 in the list.
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 5
         List of size 2 has the following items: Data Geordi
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 2
You are now removing an item from the list.
         Enter position to remove item from: 0
Item Data removed from position 0 in the list.
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 1
You are now inserting an item into the list.
         Enter item: Will
         Enter the position to insert item in: 0
Item Will inserted in position 0 in the list.
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 5
         List of size 2 has the following items: Will Geordi
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 3
         Enter position to retrieve item from: 1
Item Geordi retrieved from position 1 in the list.
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 3
         Enter position to retrieve item from: 0
Item Will retrieved from position 0 in the list.
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 3
         Enter position to retrieve item from: 8
Position specified is out of range!
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 5
         List of size 2 has the following items: Will Geordi
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 6
Smallest item Geordi deleted.
Largest item Will deleted.
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 5
List is empty.
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 7
List is empty... nothing to reverse!
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 1
You are now inserting an item into the list.
         Enter item: Velcro
         Enter the position to insert item in: 0
Item Velcro inserted in position 0 in the list.
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 5
         List of size 1 has the following items: Velcro
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 7
         Here is the content: Velcro
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 6
Velcro is deleted.
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 5
         List of size 1 has the following items: Velcro
Select from the following menu:
         0. Exit the program
         1. Insert item into the list
         2. Remove item from the list
         3. Get item from the list
         4. Clear the list
         5. Display size and content of the list
         6. Delete the smallest and largest item in the list
         7. Reverse the list

Make your menu selection now: 0
Exiting program... good bye
::::::::::::::
Lab3P3Sampleruns.txt
::::::::::::::
```

```
Some examples of compared strings are:
'a' compared to 'A' is : 32
't' compared to 'T' is : 32
'a' compared to 'z' is : -25
'a' compared to 'l' is : -11
'l' compared to 'A' is : 43
'1' compared to '9' is : -8
'1' compared to '18' is : -1
'21' compared to '81' is : -6
'to' compared to 'top' is : -1
'%' compared to '0' is : -11
Select from the following menu:
         0. Exit the program
         1. Compare two strings
1
You are now comparing two strings...
         Enter string 1: Zebra
         Enter string 2: zebra
'Zebra' compared to 'zebra' is: -32
Select from the following menu:
         0. Exit the program
         1. Compare two strings
1
You are now comparing two strings...
         Enter string 1: Animal
         Enter string 2: *&%
'Animal' compared to '*&%' is: 23
Select from the following menu:
         0. Exit the program
         1. Compare two strings
1
You are now comparing two strings...
         Enter string 1: Chocolate
         Enter string 2: charcoal
'Chocolate' compared to 'charcoal' is: -32
Select from the following menu:
         0. Exit the program
         1. Compare two strings
1
You are now comparing two strings...
         Enter string 1: zero
         Enter string 2: zero
'zero' compared to 'zero' is: 0
Select from the following menu:
         0. Exit the program
         1. Compare two strings
1
You are now comparing two strings...
         Enter string 1: two
         Enter string 2: too
'two' compared to 'too' is: 8
Select from the following menu:
         0. Exit the program
         1. Compare two strings
1
You are now comparing two strings...
         Enter string 1: Sesquipedalianism
         Enter string 2: I
'Sesquipedalianism' compared to 'I' is: 10
Select from the following menu:
         0. Exit the program
         1. Compare two strings
0
Exiting program... good bye
```