**Name:** Donark Patel (DP663)

**Class:** IS114-451

**Professor:** Maura Deek

**Date:** 11/1/2019

**1. Formulating the Problem**

**1.1 Problem Description**

Create program with a GUI for users to enter a value and index number. Value is added to a LinkedList at given index number. User can also search the value, and delete the value.

**1.2 Verbalization**

*What is the goal?*

Create program with a GUI for users to enter a value and index number. Value is added to a LinkedList at given index number. User can also search the value, and delete the value.

*What are the givens?*

User’s inputs the index number and value.

*What are the unknowns?*

User can search index number with input of value.

1.3 **Information Elicitation**

*Goal*: Create program with a GUI for users to enter a value and index number. Value is added to a LinkedList at given index number. User can also search the value, and delete the value.

*Givens*: User’s inputs the index number and value.

*Unknowns*: User can search index number with input of value.

*Conditions*:

Program can only user LinkedList. Inputted value has to be linked to inputted index number.

**2. Planning the Solution**

**2.1 Solution Strategy**

Search button: Search button are allowed to insert only value field. Once the user input the value. Program crates a LinkedList iterator. With the iterator’s hasNext and Next method Program goes through each node of the link list. One it find the value. It output the value and index number.

Insert button: Insert button takes two fields value and index number. Once the user enters the index number. A validation method validates that inputted value of a index number must be a number. Once the validation is done. Program adds the value field to the index number of list.

Delete button: Delete button takes one input value, the index number. Once the user input the value. Program crates a LinkedList iterator. With the iterator’s hasNext and Next method Program goes through each node of the link list. One it find the value. Using the Set method. It set the index value to null.

**)2.2 Goal Decomposition**

*Sub-goal 1*: Create the GUI panel.

*Sub-goal 2*: Validate the input.

*Sub-goal 3*: Create a LinkedList object

*Sub-goal 4*: Insert the value.

*Sub-goal 5*: Search the value.

*Sub-goal 6*: Delete the value.

**2.3 Resources**

*Relevant formulas*

**None**

**2.4 Data Organization and Description**

Input:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Description | Origin | Used in Sub-goal # |
| frame | GUi Frame | User | 1 |
| lblIndexNumber | Label for index Number | User | 1 |
| lblValue | Label for value | User | 1 |
| textFieldIndex | TextBox for index number | User | 1, 2, 4, 5, 6 |
| textFieldValue | TextBox for value | User | 1, 4, 5 |
| btnSearch | Button for search | User | 1, 2, 5 |
| btnInsert | Button for insert | User | 1, 2, 4 |
| btnDelete | Button for delete | User | 1, 2, 6 |
| list | LinkedList | Program | 4, 5, 6 |

Output:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Description | Origin | Used in Sub-goal # |
| textArea | TextArea to Output String | Screen | 4, 5, 6 |

**3. Designing the Solution**

**)3.1 Structure Chart**

*First Level Decomposition*

*Goal Refinement*

**Sub-goal 1**

Create the GUI Panel

**Sub-Goal 1.1**

From main class initialize the GUI panel class

**Sub-goal 2**

Validate the input

**Sub-goal 2.1**

Check the length of the input using .length() method.

**Sub-goal 2.2**

If the length is less, then 1. Then output error.

**Sub-goal 3**

Create LinkedList object.

**Sub-goal 4**

Insert the value.

**Sub-goal 4.1**

Add the value at given index number

**Sub-goal 4.2**

If the index value grater then size of LinkedList. Then add null values till the size is equal to inputted index number.

**Sub-goal 5**

Search the value.

**Sub-goal 5.1**

Create a LinkedList Iterator.

**Sub-goal 5.2**

Loop though each element of the list.

**Sub-goal 5.3**

If the value is found, Output the value and index number.

**Sub-goal 6**

Delete the value.

**Sub-goal 5.1**

Using the Set method. It set the index value to null.

*Second Level Decomposition*

**()3.2 Module and Data Specifications**

**Name**: initialize – handles the frame elements.

**Input**: frame, lblIndexNumber, lblValue, textFieldIndex, textFieldValue, btnSearch, btnInsert, btnDelete.

**Output**: None

**Logic**: None

**Name**: btnDelete.addActionListener – Delete button ActionListener

**Input**: String

**Output**: String

**Logic**: Delete button takes one input value, the index number. Once the user input the value. Program crates a LinkedList iterator. With the iterator’s hasNext and Next method Program goes through each node of the link list. One it find the value. Using the Set method. It set the index value to null.

**Name**: Main – Class Frame class.

**Input**: None

**Output**: GUI panel

**Name**: btnInsert.addActionListener – Insert button ActionListener

**Input**: String

**Output**: String

**Logic**: Insert button takes two fields value and index number. Once the user enters the index number. A validation method validates that inputted value of a index number must be a number. Once the validation is done. Program adds the value field to the index number of list.

**Name**: btnSearch.addActionListener – Search button ActionListener

**Input**: String

**Output**: String

**Logic**: Search button are allowed to insert only value field. Once the user input the value. Program crates a LinkedList iterator. With the iterator’s hasNext and Next method Program goes through each node of the link list. One it find the value. It output the value and index number.

**Data:**

|  |  |  |
| --- | --- | --- |
| Name | Type | Structure |
| frame | JFrame | Object |
| lblIndexNumber | JLabel | Object |
| lblValue | JLabel | Object |
| textFieldIndex | JTextField | Object |
| textFieldValue | JTextField | Object |
| btnSearch | JButton | Object |
| btnInsert | JButton | Object |
| btnDelete | JButton | Object |
| list | String | Object |
| value | String | Variable |
| it | ListIterator | Object |
| tempindex | integer | Variable |
| temp | String | Variable |

**3.3 Algorithm**

*Logic*

1.0: Create the GUI Panel

1.1: From main class initialize the GUI panel class.

2.0: Validate the input

2.1: Check the length of the input using .length() method.

2.2: If the length is less, then 1. Then output error.

3.0: Create LinkedList object.

4.0: Insert the value.

4.1 Add the value at given index number

4.2 If the index value grater then size of LinkedList. Then add null values till the size is equal to inputted index number.

5.0: Search the value.

5.1: Create a LinkedList Iterator.

5.2: Loop though each element of the list.

5.3: If the value is found, Output the value and index number.

6.0: Delete the value.

6.1: Using the Set method. It set the index value to null.

*Algorithm Description*

The program displays the GUI, where user can input the string. Insert button takes two fields value and index number. Once the user enters the index number. A validation method validates that inputted value of a index number must be a number. Once the validation is done. Program adds the value field to the index number of list. Search button are allowed to insert only value field. Once the user input the value. Program crates a LinkedList iterator. With the iterator’s hasNext and Next method Program goes through each node of the link list. One it find the value. It output the value and index number. Delete button takes one input value, the index number. Once the user input the value. Program crates a LinkedList iterator. With the iterator’s hasNext and Next method Program goes through each node of the link list. One it find the value. Using the Set method. It set the index value to null.

**4. Translation**

**4.1** **Source Code**

/\*

Name: Donark Patel (DP663)

Class: IS114-451

Professor: Maura Deek

Date: 11/12/2019

Information: This program create's a GUI for users to enter a value and index number. Value is added to a linkedList.

\*/

**import** java.awt.EventQueue;

**import** javax.swing.JFrame;

**import** javax.swing.JLabel;

**import** javax.swing.JOptionPane;

**import** java.awt.Font;

**import** java.util.LinkedList;

**import** java.util.ListIterator;

**import** javax.swing.JTextField;

**import** javax.swing.JButton;

**import** java.awt.event.ActionListener;

**import** java.awt.event.ActionEvent;

**import** javax.swing.JTextArea;

**import** javax.swing.JScrollPane;

**public** **class** Frame {

**private** JFrame frame;

**private** JTextField textFieldIndex;

**private** JTextField textFieldValue;

**private** JTextArea textArea;

**public** **int** indexNumber;

**public** String value;

LinkedList<String> list = **new** LinkedList<String>();

/\*\*

\* Launch the application.

\*/

**public** **static** **void** main(String[] args) {

EventQueue.*invokeLater*(**new** Runnable() {

**public** **void** run() {

**try** {

Frame window = **new** Frame();

window.frame.setVisible(**true**);

} **catch** (Exception e) {

e.printStackTrace();

}

}

});

}

/\*\*

\* Create the application.

\*/

**public** Frame() {

initialize();

}

/\*\*

\* Initialize the contents of the frame.

\*/

**private** **void** initialize() {

frame = **new** JFrame();

frame.setBounds(100, 100, 452, 434);

frame.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

frame.getContentPane().setLayout(**null**);

JLabel lblIndexNumber = **new** JLabel("Index number:");

lblIndexNumber.setFont(**new** Font("Tahoma", Font.***PLAIN***, 16));

lblIndexNumber.setBounds(35, 53, 106, 20);

frame.getContentPane().add(lblIndexNumber);

JLabel lblValue = **new** JLabel("Value:");

lblValue.setFont(**new** Font("Tahoma", Font.***PLAIN***, 16));

lblValue.setBounds(96, 84, 45, 20);

frame.getContentPane().add(lblValue);

textFieldIndex = **new** JTextField();

textFieldIndex.setBounds(147, 55, 86, 20);

frame.getContentPane().add(textFieldIndex);

textFieldIndex.setColumns(10);

textFieldValue = **new** JTextField();

textFieldValue.setColumns(10);

textFieldValue.setBounds(147, 86, 86, 20);

frame.getContentPane().add(textFieldValue);

JScrollPane scrollPane = **new** JScrollPane();

scrollPane.setBounds(35, 211, 354, 173);

frame.getContentPane().add(scrollPane);

JTextArea textArea = **new** JTextArea();

scrollPane.setViewportView(textArea);

textArea.setText("Log: \n");

JButton btnSearch = **new** JButton("Search");

btnSearch.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

//textArea.setText("");

value = textFieldValue.getText();

ListIterator it= list.listIterator();

**boolean** found = **false**;

**while**(it.hasNext()) {

String temp = (String) it.next();

**int** tempindex = it.nextIndex();

**if**(!value.equals(temp))

{

textArea.append("\nValue " + value + " is not found at index " + (tempindex - 1));

found = **false**;

}

**else** **if**(value.equals(temp)) {

JOptionPane.*showMessageDialog*(**null**,"Found " + value + " at index " + (tempindex - 1));

textArea.append("\nFound " + value + " at index " + (tempindex - 1));

found = **true**;

**return**;

}

}

**if**(found == **false**)

{

JOptionPane.*showMessageDialog*(**null**,value + " not found in the list");

}

textFieldValue.setText("");

textFieldIndex.setText("");

}

});

btnSearch.setBounds(151, 136, 106, 64);

frame.getContentPane().add(btnSearch);

JButton btnInsert = **new** JButton("Insert");

btnInsert.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

String temp = textFieldIndex.getText();

**try**

{

indexNumber = Integer.*parseInt*(temp);

}

**catch** (Exception E)

{

JOptionPane.*showMessageDialog*(**null**,"Invalid input: Please, enter a number in the index field");

**return**;

}

value = textFieldValue.getText();

**int** size = list.size();

**if** (indexNumber > size)

{

**while** (indexNumber > size)

{

String temp1 = **null**;

list.add(size, temp1);

size++;

}

list.add(indexNumber, value);

textArea.append("\nAdd value \nIndex: " + indexNumber + "\nValue: " + value + "\nLinkedList Size: " + list.size()+ "\n---------------------------\n");

}

**else** **if** (indexNumber == size)

{

list.add(indexNumber, value);

textArea.append("\nAdd value \nIndex: " + indexNumber + "\nValue: " + value + "\nLinkedList Size: " + list.size()+"\n---------------------------\n");

}

**else**

{

list.set(indexNumber, value);

textArea.append("\nAdd value \nIndex: " + indexNumber + "\nValue: " + value + "\nLinkedList Size: " + list.size()+"\n---------------------------\n");

}

textFieldValue.setText("");

textFieldIndex.setText("");

}

});

btnInsert.setBounds(35, 136, 106, 64);

frame.getContentPane().add(btnInsert);

JButton btnDelete = **new** JButton("Delete");

btnDelete.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

**boolean** found = **false**;

String temp = textFieldIndex.getText();

**try**

{

indexNumber = Integer.*parseInt*(temp);

}

**catch** (Exception E)

{

JOptionPane.*showMessageDialog*(**null**,"Invalid input: Please, enter a number in the index field");

**return**;

}

String temp1 = **null**;

**if**(indexNumber < list.size())

{

list.set(indexNumber, temp1);

JOptionPane.*showMessageDialog*(**null**,"Value at index " + indexNumber + " has been deleted");

textArea.append("\nValue at index " + indexNumber + " has been deleted\n");

}

**else**

{

JOptionPane.*showMessageDialog*(**null**,"Index number " + indexNumber + " is greater than size of the list. So, the value cannot be deleted.");

}

textFieldValue.setText("");

textFieldIndex.setText("");

}

});

btnDelete.setBounds(267, 136, 106, 64);

frame.getContentPane().add(btnDelete);

JButton btnClear = **new** JButton("Clear");

btnClear.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

textFieldValue.setText("");

textFieldIndex.setText("");

textArea.setText("");

list.clear();

JOptionPane.*showMessageDialog*(**null**,"Note: LinkedList data has been erased");

}

});

btnClear.setBounds(267, 31, 108, 35);

frame.getContentPane().add(btnClear);

JButton btnExit = **new** JButton("Exit");

btnExit.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

System.*exit*(0);

}

});

btnExit.setBounds(267, 79, 108, 35);

frame.getContentPane().add(btnExit);

}

}

**4.2 Program and Module Description**

Frame1

Build the GUI frame.

btnDelete.addActionListener

Delete button ActionListener

btnInsert.addActionListener

Insert button ActionListener

btnSearch.addActionListener

Create a temporary char Array

Main

Initialize the frame.

**5. Solution Testing**

Test Case:

|  |  |  |  |
| --- | --- | --- | --- |
| Insert Button | | | |
| Component | Entry | Status of input | Reason |
| Index number | Null | Invalid | Index number cannot be empty |
|  |  |  |  |
| Insert Button | | | |
| Component | Entry | Status of input | Reason |
| Index number | 3 | Valid |  |
| Value | A3 | Valid | Add value: Index: 3 Value: A3 LinkedList Size: 4 |
| Index number | 0 | Valid |  |
| Value | A0 | Valid | Add value: Index: 0 Value: A0 LinkedList Size: 4 |
| Index number | 1 | Valid |  |
| Value | A1 | Valid | Add value: Index: 1 Value: A1 LinkedList Size: 4 |
| Index number | 2 | Valid |  |
| Value | A2 | Valid | Add value: Index: 2 Value: A2 LinkedList Size: 4 |
|  |  |  |  |
| Search Button | | | |
| Component | Entry | Status of input | Valid Output |
| Value | A3 | Valid | Found A3 at index 3 |
| Value | A1 | Valid | Found A1 at index 1 |
| Value | A2 | Valid | Found A2 at index 2 |
| Value | A0 | Valid | Found A0 at index 0 |
| Value | A5 | Valid | A5 is not |
|  |  |  |  |
|  |  |  |  |
| Delete Button | | | |
| Component | Entry | Status of input | Valid Output |
| Index | 3 | Valid | Value at index 3 has been deleted |
| Index | 6 | Invalid | Index number 6 is greater than size of the list. So, the value cannot be deleted. |

**(This is just a sample what you need to submit are actual screenshots of I/O or files)**











