**IT 114 - ADVANCED PROGRAMMING FOR INFORMATION TECHNOLOGY**

**Case Studies**

**1. Formulating the Problem**

**1.1 Problem Description**

We must create a program that gives the user the option to search, insert, or delete from a linked list. To search the user must input the value they are looking for in the list and the system will reply with the index of where its located. If the user selects Insert, the program takes the user’s desired value and desired index location. If the index is out of bounds of the list, the program will create a popup window saying its out of range. If the user selects delete the system takes the user’s desired index and deletes it from the Linked List.

If invalid data is entered, an error message appears in a popup window.

**1.2 Verbalization**

*What is the goal?*

Create a program that lets the user search for a value in a Linked List, Insert a value into a specific index in a Linked List, or Delete a Value from the Linked List

*What are the givens?*   The default values in the LinkedList

*What are the unknowns?*   What will be added to or deleted from the linked list

1.3 **Information Elicitation**

*Goal*

Create a program that – lets you choose between searching for the index of a value in a linked list --- inserts a value into a specific index --- deletes a specified index from the linked list --- gives a popup for error messages --- gives a popup for search results.

*Givens*   The default values in the linked list

*Unknowns*   What will be added to or removed from the linked list, or what their locations are

*Conditions*   The user must stay in range of the linked list

**2. Planning the Solution**

**2.1 Solution Strategy**

Create an already populated linked list so that the user can choose ones from the get go. Allow the user to search for, insert, or delete from the linked list view buttons on the bottom of the GUI. If search is selected, take the user’s value input and search the linked list for the given value and return its index. Display the results in a popup window. If insert is selected, take the user’s value input and index input, and insert the value to the specified index in the linked list. If the index is out of range, create a popup window saying the chosen index is out of range. If delete is selected, take the user’s specified index and delete it from the linked list. If the index is out of range of the linked list, create an error message saying its out of range. To make sure that the user knows what values are in the list, create a list view that shows what is in there.

**)**

**2.2 Goal Decomposition**

*Sub-goal 1*

Create a Linked List

*Sub-goal 2*

Create Buttons Search, Insert, Delete

*Sub-goal 3*

Take user value input

*Sub-goal 4*

Take user index input

*Sub-goal 5*

Create List view to display what is in the Linked List

*Sub-goal 6*

Create a method for creating a second window

*Sub-goal 7*

Create TextFields for value and index to take the user inputs

*Sub-goal 8*

Create error messages if the user is out of range of the index

*Sub-goal 9*

Create error message if the value isn’t in the linked list

**2.3 Resources**

*Relevant Information*

**Linked List Default Values:** thanksgiving, pie, turkey

**3. Designing the Solution**

**)3.1 Structure Chart**

*First Level Decomposition*



*Goal Refinement*

**Sub-goal 1**

Create a linked list

**Sub-goal 2**

Create a gui with 3 buttons; search, insert, delete

**Sub-goal 3**

Create 2 text fields in the GUI for index input and value input

**Sub-goal 4**

If search is selected, take the user’s input from the value field and search the linked list for it.

**Sub-goal 4.1**

Create value variable to hold the user’s input

**Sub-goal 4.2**

Use a Boolean to see if it was found or not

**Sub-goal 4.3**

If found or not, display results in a popup window.

**Sub-goal 5**

If insert is selected, take the user’s input from the value and index fields

**Sub-goal 5.1**

Create index and value variables to hold the user’s input

**Sub-goal 5.2**

insert the value to user’s the desired index in the linked list.

**Sub-goal 5.3**

If index out of range, return error message in popup window

**Sub-goal 6**

If delete is selected, take the user’s input from the index field

**Sub-goal 6.1**

Create index variable to hold the user’s value from the index textfield

**Sub-goal 6.2**

Remove the index form the linked list

**Sub-goal 6.3**

If index out of range, return error message in popup window

*Second Level Decomposition*



The second level decomposition shows operations between the User and the program. The process starts with the user having the option between searching for a value in the linked list, inserting a value to the linked list, or deleting an value from the linked list. If search is selected, the program takes the user’s value input and searches the linked list for the given value and returns its index. It then displays the results in a popup window. If insert is selected, take the user’s value input and index input, and inserts the value to the specified index in the linked list. If the index is out of range, it creates a popup window saying the chosen index is out of range. If delete is selected, the program takes the user’s specified index and deletes it from the linked list. If the index is out of range of the linked list, the program creates an error message saying it’s out of range. To make sure that the user knows what values are in the list, the program creates a list view that shows what is in there and is updated whenever something is inserted or removed from the linked list.

**3.2 Module and Data Specifications**

**Name**: Choice – user selects search

**Input**: String value

**Output**: popup window saying the location of the value in the linked list, or

error message saying not found

**Logic**: action on button so when pressed takes the input form the value input field and sets it to variable value. Create a Boolean variable retval. Takes the variable value and searches for it in the linked list if found retval is true, if not retval is false. If retval is true, the index is retrieved and output into a popup window. If not, error message in popup window saying not found.

**Name**: Choice – user selects delete

**Input**: String index

**Output**: index value is deleted form linked list

**Logic**: action on button so when pressed takes the input form the index input field and sets it to variable index. If index is within the range of the linked list it removes the chosen index value from the linked list. If not, error message in popup window saying the chosen index is out of range of the linked list. The list view is updated for the removal from the LinkedList.

**Name**: Choice – user selects insert

**Input**: String index, String value

**Output**: value is inserted into linked list at the desired index location

**Logic**: action on button so when pressed takes the input form the index input field and the value input field then sets them to variables index and value. If index is within the range of the linked list it adds the chosen value to the chosen index location to the linked list. If not, error message in popup window saying the chosen index is out of range of the linked list. The list view is updated with the new addition to the LinkedList

**3.3 Logic**

*Logic*

1. Display choices to the user
2. User Inputs values to the value input field/index input field
3. User selects search

3.1 Create value variable to hold the user’s value input

3.2 Use a Boolean to see if it was found or not

3.3 If found or not, display results in a popup window.

4.0 User selects delete

4.1 Create index variable to hold the user’s value from the index textfield

4.2 check to see if the index is in range of the linked list

4.2.1 in range: remove the index form the linked list

4.3 out of range: return error message in popup window

5.0 User selects insert

5.1 create value and index variables to hold the user’s inputs

5.2 check to see if the index is in range of the linked list

5.2.1 in range: value is added to the chosen index location

5.2.2 out of range: popup window with error message saying

index is out of range.

*Algorithm Description*

The process of doing this : Data is stored in LinkedList called list1. The system gives the User an option to search the linked list for a value, insert a value to the list with a given index location, or delete a value with a specified index location. If search is selected, the program takes the user’s value input and searches the linked list for the given value and returns its index. It then displays the results in a popup window. If insert is selected, take the user’s value input and index input, and inserts the value to the specified index in the linked list, it then adds the item to the ListView so that the user can see it in the menu. If the index is out of range, it creates a popup window saying the chosen index is out of range. If delete is selected, the program takes the user’s specified index and deletes it from the linked list, it then removes the index value from the ListView so that the user can see it removed in the GUI. If the index is out of range of the linked list, the program creates a popup window with an error message saying the chosen index is out of range.

**4. Translation**

**4.1** **Source Code**

I removed the indents to try to make it easier to read in this document. With the indents it was hard to look at.

//===================================================   
// Name : Emad Tirmizi  
// SID : 31400222  
// Course : IT114   
// Section :   
// Instructor : Maura Deek  
// T.A :   
//===================================================   
//===================================================   
// Assignment # : 3  
// Date : 11/2/2018  
//===================================================   
//===================================================   
// Description: This program will check to see if the  
// if the starting point chosen by the user can be  
// solved through the use of a recursive method. It  
// informs the user whether or not it is able to be  
// solved using the starting point provided by the  
// user  
//===================================================   
  
  
//======================================================   
// Name : Emad Tirmizi  
// SID : 31400222  
// Course : IT114   
// Section :   
// Instructor : Maura Deek  
// T.A :   
//======================================================   
//======================================================   
// Assignment # : 4  
// Date : 11/14/2018  
//======================================================   
//======================================================   
// Description: We must create a program that gives  
// the user the option to search, insert, or delete  
// from a linked list. To search the user must input   
// the value they are looking for in the list and the  
// system will reply with the index of where its   
// located. If the user selects Insert, the program   
// takes the user’s desired value and desired index  
// location. If the index is out of bounds of the list,  
// the program will create a popup window saying its   
// out of range. If the user selects delete the system   
// takes the user’s desired index and deletes it from   
// the Linked List. If invalid data is entered, an error  
// message will appear in a popup window.   
//======================================================   
  
**import** javafx.scene.layout.GridPane;  
**import** javafx.application.Application;  
**import** javafx.event.ActionEvent;  
**import** javafx.event.EventHandler;  
**import** javafx.stage.Stage;  
**import** javafx.scene.Scene;  
**import** javafx.geometry.Insets;  
**import** javafx.scene.layout.VBox;  
**import** javafx.scene.layout.BorderPane;  
**import** javafx.scene.control.Label;  
**import** javafx.scene.control.Button;  
**import** javafx.scene.control.TextField;  
**import** javafx.scene.control.TextArea;  
**import** javafx.scene.control.ScrollBar;  
**import** javafx.scene.layout.ColumnConstraints;  
**import** javafx.geometry.Orientation;  
**import** javafx.geometry.Pos;  
**import** javafx.geometry.VPos;  
**import** java.util.**\***;  
**import** javafx.collections.ObservableList;  
**import** javafx.scene.control.**\***;  
**import** javafx.collections.FXCollections;  
  
//Create Main Class  
**public** **class** Main **extends** Application{   
//set variables  
Stage window;  
Scene scene;  
   
Label lb1;  
Label lb2;  
   
Button b1;  
Button b2;  
Button b3;

ListView**<**String**>** listView;  
   
//main method to launch  
**public** **static** **void** main(String[] args) {  
launch(args);   
}  
  
**@**Override  
**public** **void** start(Stage primaryStage) **throws** Exception{  
window **=** primaryStage;  
window.setTitle(**"Linked List"**);  
   
//I used grid the get the layout I wanted  
GridPane grid **=** **new** GridPane();  
//The padding is to essentially set the margins for which my nodes won't go past  
grid.setPadding(**new** Insets(10,10,10,10));  
grid.setVgap(10);  
grid.setHgap(15);  
   
//I set default values for the linked list so that the user can already choose from the list and see what is in there  
String[] values **=** {**"thanksgiving"**, **"pie"**, **"turkey"**};  
List**<**String**>** list1 **=** **new** LinkedList**<**String**>**();  
**for**(String x**:** values){  
list1.add(x);  
}//end for loop  
   
//label for Value Input TextField  
Label lb1 **=** **new** Label(**"Value Input"**);  
GridPane.setConstraints(lb1, 0, 0);  
   
//value TextField  
TextField valueInput **=** **new** TextField();  
//I set a constraint as to how wide it should be because it was messing up the buttons look  
valueInput.setMaxWidth(200);  
GridPane.setConstraints(valueInput, 1, 0);  
   
//label for index input TextField  
Label lb2 **=** **new** Label(**"Index Input"**);  
GridPane.setConstraints(lb2, 0, 2);  
   
//index TextField  
TextField indexInput **=** **new** TextField();  
//I set a constraint as to how wide it should be because it was messing up the buttons look

indexInput.setMaxWidth(200);  
GridPane.setConstraints(indexInput, 1, 2);  
   
//label for button1  
//Button 1 searches the program for the desired value  
Button b1 **=** **new** Button(**"Search"**);  
GridPane.setConstraints(b1, 1, 4);  
//This button will take the input from the value textfield and search the linked list for it  
b1.setOnAction(e **->** {

String output **=** **""**;  
String value **=** **""**;  
value **=** valueInput.getText();  
   
//use boolean to check if its in the linked list  
**boolean** retval **=** list1.contains(value);  
//value is in the linked list  
**if**(retval **==** **true**){  
//create a string that can be displayed to the user with the results  
output **=** **"the value: "** **+** value **+** **" was found at index: "** **+** list1.indexOf(value)**+"\n"**;  
}//end if  
//if not in the list  
**else**{  
//create a string that can be displayed to the user with the results  
output **=** **"The value doesn't exist in the Linked List"**;  
}//end else  
//open popup window and display the output for the user  
display(**"Search"**, output);   
});//end button action  
   
//label for button1  
//Button 2 inserts a value at a given index  
Button b2 **=** **new** Button(**"Insert"**);  
GridPane.setConstraints(b2, 2, 4);  
//This button takes the value and index inputs from the text fields and inserts it into the list  
b2.setOnAction(e **->** {  
**int** index **=** 0;  
String value **=** **""**;  
String output **=** **""**;  
   
index **=** Integer.parseInt(indexInput.getText());   
value **=** valueInput.getText();  
   
//checking to see if the index is in range of the list  
**if**(index **<=** list1.size()){  
list1.add(index, value);  
//update the listview with the value added  
listView.getItems().add(value);  
}//end if  
//if index isn't in range of the list it creates this popup window  
**else**{  
//popup window displays this output  
output **=** **"Sorry, that index is out of range of the Linked List"**;  
display(**"Error"**, output);  
}//end else  
   
});//end button action  
   
//label for button3  
//Button 3 deletes an a value from the LinkedList by taking in an index  
Button b3 **=** **new** Button(**"Delete"**);  
GridPane.setConstraints(b3, 3, 4);  
//This button takes the index from the index input text field and deletes it from thelist  
b3.setOnAction(e **->** {  
**int** index **=** 0;  
index **=** Integer.parseInt(indexInput.getText());  
String output **=** **""**;  
//checking to see if the index is in range of the list  
**if**(index **<=** list1.size()){  
list1.remove(index);   
//update the listview with the value removed  
listView.getItems().remove(index);  
}//end if  
//if not in range, popup window shows with error message  
**else**{  
//error message to be shown to the user in popup window  
output **=** **"Sorry, that index range doesn't exist in the Linked List"**;

display(**"Error"**, output);  
}//end else  
});//end button action  
   
//listview to show the current list in the GUI  
listView **=** **new** ListView**<>**();  
listView.getItems().addAll(list1);  
 listView.setMaxWidth(150);  
 GridPane.setConstraints(listView, 1, 5);  
   
grid.getChildren().addAll(valueInput, indexInput, b1, b2, b3, lb1, lb2, listView);  
  
//set scene parameter to grid as the layout profile   
scene **=** **new** Scene(grid, 550, 300);  
//set scene  
window.setScene(scene);  
//show display the stage  
window.show();  
}  
//create a method that makes a generic popup window that takes a title and input text  
**public** **static** **void** display(String title, String text){  
Stage window **=** **new** Stage();  
   
window.setTitle(title);  
window.setWidth(300);  
   
TextArea Out **=** **new** TextArea();  
//I set a constraint as to how wide it should be because it was messing up the buttons look  
Out.setMaxWidth(250);  
Out.setText(text);  
Button closeButton **=** **new** Button(**"Exit Window"**);  
closeButton.setOnAction(e **->** window.close());  
   
VBox layout **=** **new** VBox(10);  
layout.getChildren().addAll(Out, closeButton);  
//center position everything  
layout.setAlignment(Pos.CENTER);  
   
Scene scene **=** **new** Scene(layout);  
window.setScene(scene);  
window.show();   
}  
}

**)4.2 Program and Module Description**

Main

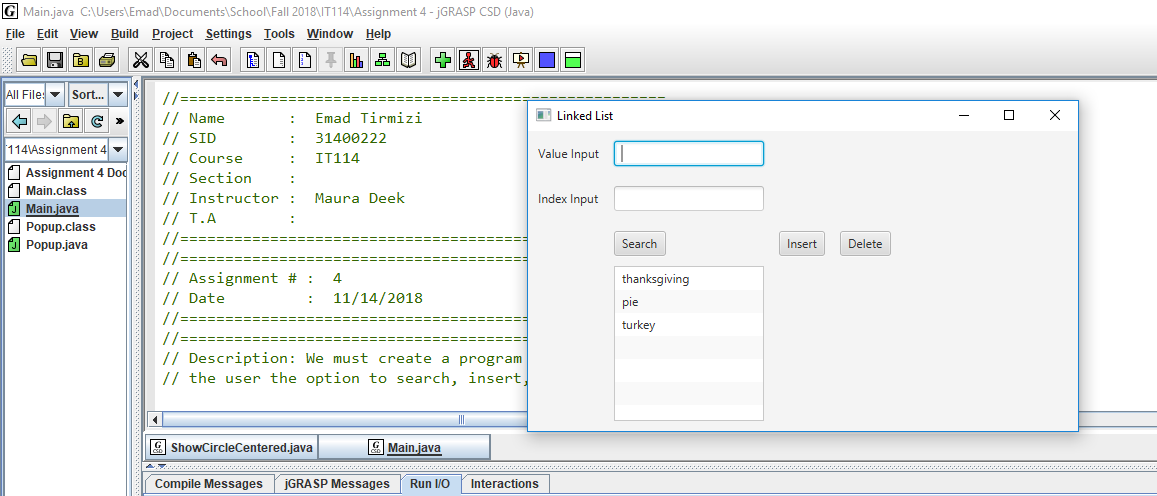
The main function creates the program GUI when started.

**5. Solution Testing**

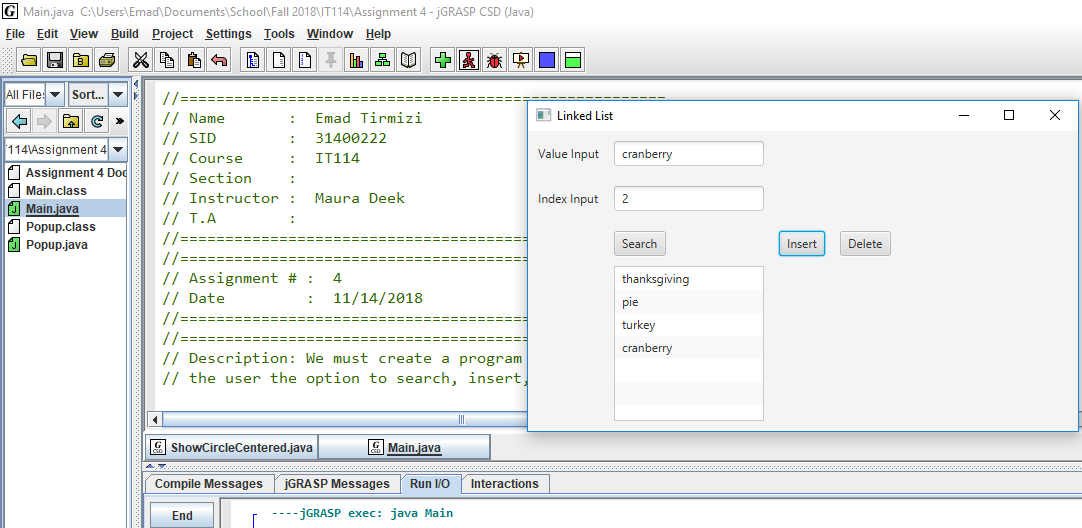
Test the program with following data domain:

thanksgiving, pie, turkey

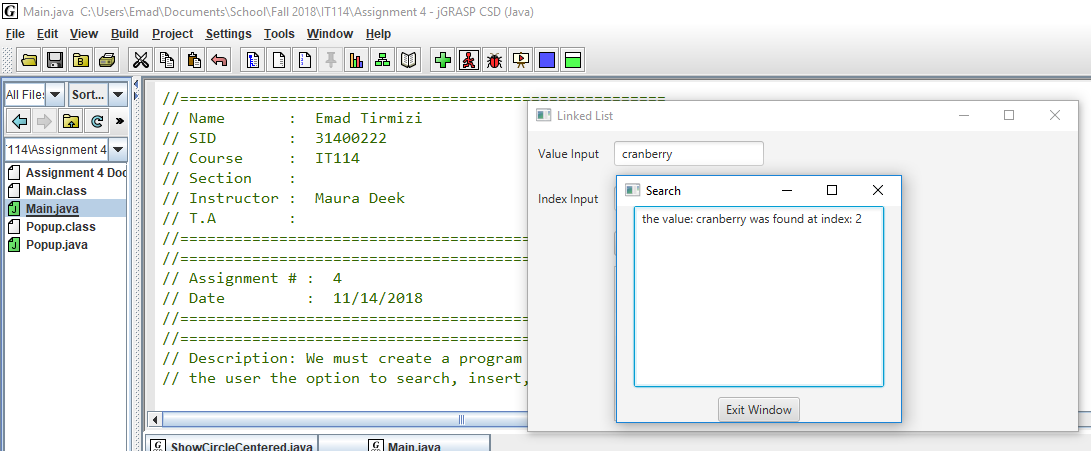
Main Menu



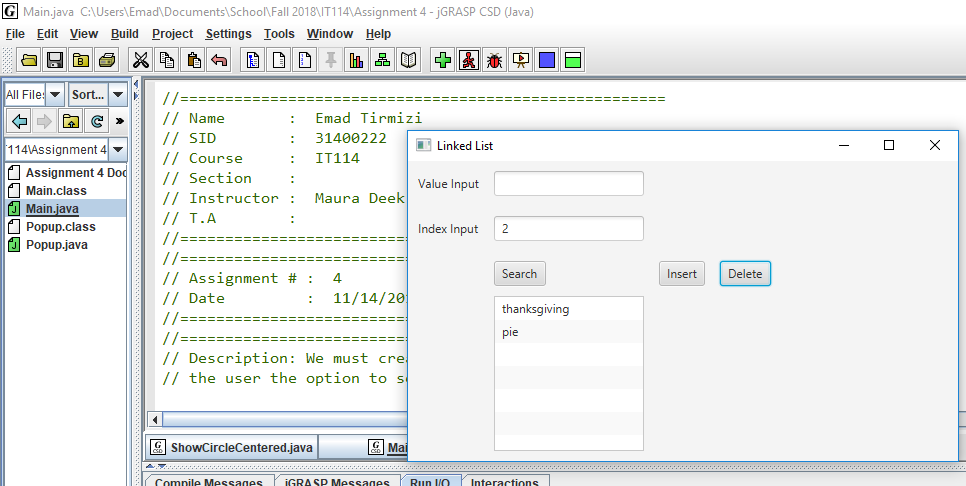
Inserting Cranberry to index 2



Checking location of cranberry is in index 2 with search

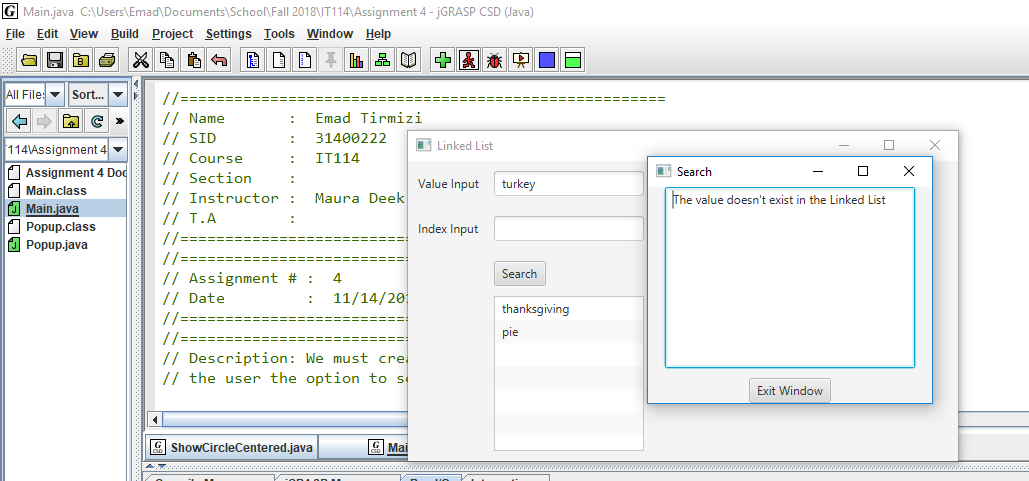


From fresh load, deleting index 2 from the list

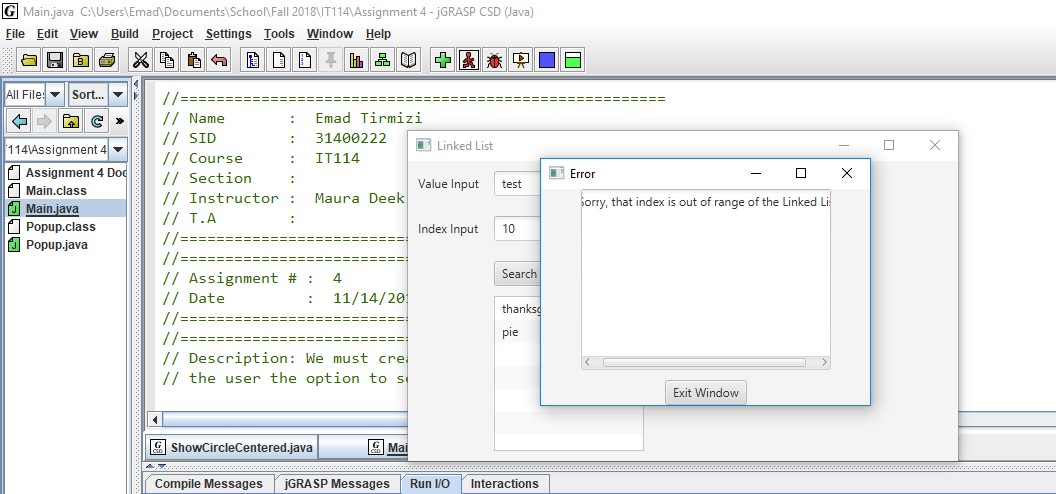


Checking to see if Turkey, which was index 2, was deleted from the list

Popup showing it was deleted and doesn’t exist anymore



Popup showing that the index for inserting test is out of range of the linked list



Popup showing that the index for deleting is out of range for the linked list

