Lab 11: ListMaker

IT 1090C Computer Programming I

IT 6090C Java Programming

Prof. Jason Geanuracos

Spring 2025

# Lab 11: ListMaker

20 Points

## Learning Goals

* Work with the **ArrayList** to create a dynamic list
* Develop additional one-off static private support methods for this program
* Use the **SafeInput** library we created for the input

# Mini-lecture:

ArrayLists are similar to arrays but are dynamic and flexible. ArrayLists only hold objects, so you have to use one of the java object wrapper classes for an Arraylist of primitives (Integer, Boolean, Double, Character, etc...) For this assignment, our ArrayList will be of type String.

**import java.util.ArrayList**; // required for using

Create a new ArrayList:

**ArrayList**<String> **myArrList** = new **ArrayList**<>(); // note the diamond notation on the type parameter <> so you don’t have to repeat the type

Add and element to the end

**myArrlist.add("Some String");**

Read an element at location m which is an index just like an array

**String var = myArrList.get(m);** // again m is the index which is zero based as in the arrays

Insert an element at location m

**myArrlist.add(m, "Some String");** // again m is an index like an array

See how the add method has two forms? We call this overloading.

How many items does the **arrayList** hold:

**myArrList.size();**

Overwrite or replace an item at index m:

**myArrList.set(m, "New String Value");**

So unlike sparse arrays where we have unused space. ArrayLists always are filled so the size() is the actual number of elements.

# Lab

## Directions

You will code a simple line editor that lets users create on screen lists. A list is just a series of lines which are Strings of text. We will store the list in an ArryList<String> Use the SafeInput Library for all inputs.

1. Use IntelliJ to create a java project called **Lab\_11\_Listmaker** and create the GitHub repo for your code.
2. Start by creating a menu driven loop:

a) The Loop awaits user input until the Quit command is issued   
 Command Options:

A – Add an item to the list  
D – Delete an item from the list  
I – Insert an item into the list  
P – Print (i.e. display) the list  
Q – Quit the program (This should do an are you sure? type query before exiting.)

b) The program gets one of these commands from the user and executes that function  
- Add an item always puts it at the end of the list  
- Delete an item user has to specify which one using the item number from the display  
- Insert an item user has to indicate where using an location number  
- Print the list just displays the list  
- Quit asks the user if they are sure and then terminates the program.  
  
3. Initially stub out the functions so you have a program that you can run almost immediately as you develop it as per the Agile Software Dev approach. You will develop each of the menu options as a separate java method.  
You also will need some private static utility functions:

* You will want to display the current list along with the menu of options so the user can see what they are doing.
* You need to display a numbered version of the list to allow users to pick list elements for deletion. Here the user looks at the display and then indicates the item to delete by the number.
* You should use your SafeInput library to bulletproof all input.  
  For instance, use your getRegExString method to get the menu choice from the user. Here, a regEx pattern like this [AaDdIiPpQq] creates a set where a match will be any one of these characters which of course are the menu choices that the user will make. Note that we have to include both the upper and lower case.
* Use your getRangedInt method to get the item number to delete, etc.
* Use your getYNConfirm method for the quit prompt etc.

Document your work:  
Create a sequence of screen shots that show that you coded each of the menu functions. Do this in a series of operations. i.e. build a list, insert an item, delete an item view the list. Several examples of each operation.  
Paste these here, please caption each image with the operation.

# Post-Lab: Double-check Your Work

Now that you have completed the lab tasks, build constructive professional work habits by carefully rechecking your work, and abiding by the naming format and submission instructions.

## Rename the file:

Use the **YourLastname\_YourFirstname\_Lab11.docx** format; remember to replace "**YourLastname**" and "**YourFirstname**"withyour actual names.

The naming format will help us efficiently track and organize your files.

## GitHub Link

Paste the link to your GitHub repo here.

## Submit

Submit to Canvas using the assignments mechanism to upload your document. **Don’t submit a link to your document.**

A screenshot of a computer program

AI-generated content may be incorrect.Inserting Line and Quitting Program

A screenshot of a computer

AI-generated content may be incorrect.

Adding and Deleting Line

A screenshot of a computer

AI-generated content may be incorrect.

Inserting and Printing the list