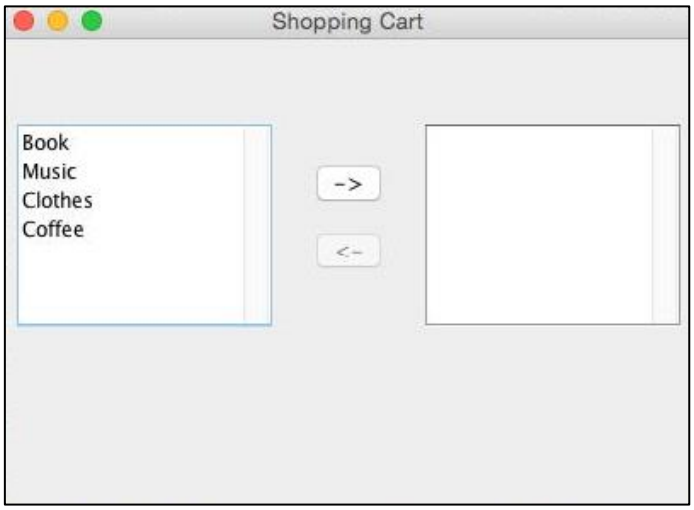
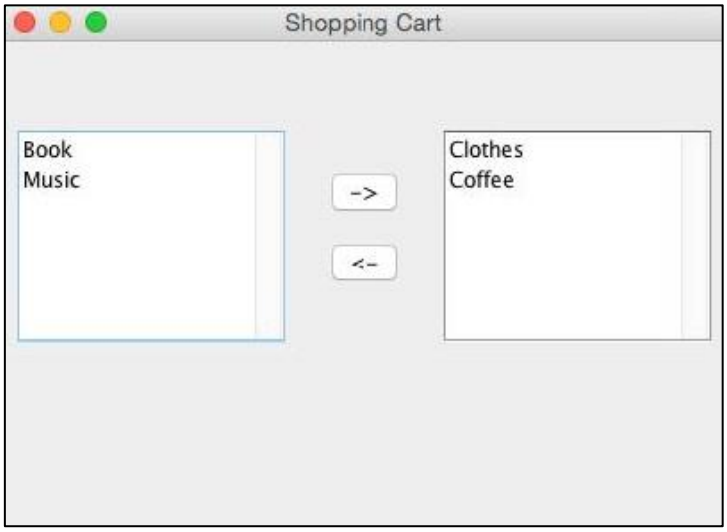
COMP503/ENSE502/ENSE602: Week 11 – Exercises

➢ Exercise 1:

Use the MVC template and design a GUI with two lists of items. The user can select one or more items and click the button to move the item to the other list.
Your GUI may look as follows:



Note that the button is disabled if there are no items in the list to move.



COMP503/ENSE502/ENSE602: Week 11 – Exercises

➢ Exercise 2: The Bank Transfer GUI

Design your own application that features a GUI to carry out money transfers between a customer's varying bank accounts. For example, a customer may wish to transfer $100 from their Savings account to their Freedom account.

There are three possible account types: Savings, Current and Freedom.

The GUI has the following functionality requirements:

- Displays customer's name and address
- Lists the customer's bank accounts: their types and balance in each.
- Has a text field to input the amount of money to transfer
- Has a button that carries out the transfer.
- Maintains and displays a list of transfers completed. The most recent transfer is placed at the top of the list.

Your application must satisfy MVC architecture: use the templates.

- Develop model classes for your application (Which classes do you need: Customer, BankAccount, Transfer?)
- Consider the kinds of components you will need in your view. Draw a diagram outlining your GUI.
- Try to display some of the data in the view.
- Add JComponent's one at a time and get them working with your model.

COMP503/ENSE502/ENSE602: Week 11 – Exercises

➢ Exercise 3: Recursion exercises

In the **Recursion** class, complete the following methods using recursion :

1. **`String plusString(String in)`** takes as input a string and adds a + symbol between identical adjacent characters.

   `plusString("Hello Bunny Rabbit!!!!")`
   returns **`Hel+lo Bun+ny Rab+bit!+!+!+!`**

2. **`multiply(int a,int b)`** takes as input two position numbers and returns the product of a and b (Look at the power example from the lecture notes). Would recursion really be an appropriate choice for implementing this method?

3. **`String reverse(String in)`** takes as input a string and returns it's reverse. Is recursion a good implementation choice?

4. **`Boolean isPal(String in)`** takes as input a string and returns true if it is a palindrome and false otherwise. (try invoking the reverse method…)

5. **`Boolean contains(String in,char ch)`** returns true if the String in contains the character ch and false otherwise. Is recursion a good implementation choice?

6. **Boolean alphaContains(String in,char ch)** returns true if the String in contains the character ch and false otherwise, assuming all characters in the String "in" are in alphabetical order. For example: alphaContains("abcde",'c')=true and alphaContains("abde",'c')=false. Is recursion a good implementation choice?