WRA202 : Practical 1
Week 02 : 29/31 July or 4 August 2014

Sections Covered

Chapter 5: Stacks (pp 111 – 115)

Objectives

Demonstrate the use of stacks to solve programming problems

Preparation

Chapter 5: Stacks (pp 111 - 115)
Activities 1 – 3 of Week 02

Compulsory Practical Tasks

Any number of delegates may register to attend a particular conference. The information recorded for a conference is name of conference, location (name of a city), maximum number of presentation slots and a cost for the conference, as well as a reference to a list of conference delegates[1]. For each delegate the following details are recorded: unique delegate identifier (numeric), delegate name, area of expertise, amount still due for the conference (on registration of a delegate, the amount due is set to the cost for the conference) and an indication of whether the delegate is a presenter at the conference (true) or not (false).

Copy to your working folder and open up the partial solution for **Activity 4** provided under \\CS2\Courses\WRA202\Week02\Act04.

Compulsory Tasks

Review the code given for methods **availSlots** and **registerDelegate**. Ensure that you fully understand what each of these methods does. Ask your tutor/group members questions until you totally comprehend these methods since you will require this understanding to generate the code required below.

You are required to implement appropriate methods ***in the correct class*** for each of:

- Method **displayDelegates**: Recursively display the delegate identifiers and names for all delegates in the list of delegates, with the one at the top being displayed first. ***Note***: The order of delegates in the list of delegates must be the same after the method has been processed as before processing.

[1] Implemented as a stack.

- Method **makePayment**: For a specified delegate (on delegate identifier), modify the relevant property to indicate that a given payment has been made. ***Note***: The order of messages in the list of delegates must be the same after the method has been processed as before processing.
- Method **duplicateList**: Return an exact duplicate of the list of delegates so that each list and its contents are entirely independent from the other list (and its related contents). ***Note***: The order of delegates in the list of delegates must be the same after the method has been processed as before processing.
- Method **reverseList**: Reverse the list of delegates.
- Method **noStillOwing**: Compute and return a count of the delegates who still owe conference costs. ***Note***: The order of delegates in the list of delegates must be the same after the method has been processed as before processing.

Optional but Recommended Tasks

You are required to implement appropriate methods ***in the correct class*** for each of:

- Method **deleteDelegate**: Remove a specified delegate (on delegate identifier) from the list of delegates. If the delegate is in the list of delegates, it is removed from the list of delegates, a warning is displayed if the delegate is also a presenter and a boolean value of **True** is returned. If the specified delegate is not in the list of delegates, a boolean value of **False** is returned. ***Note***: The list of delegates must retain the ordering of the remaining delegates prior to a successful delegate removal.
- Method **totalDue**: Compute and return the total amount still owing by delegates. ***Note***: The order of delegates in the list of delegates must be the same after the method has been processed as before processing.