**Module: Stack**

Stacks is a data structure that arrange its elements in a linear order. Common functions of List are insert an element into a list, delete an element from a list, search if an element is in a list, display all elements in a list and return the number of elements in a list. List can be implemented using array and pointer.

Learning outcome:

* To design and develop array-based Stack based on the given ADT
* To design and develop pointer-based Stack based on the given ADT
* To apply Stacks for solving problem

Exercise 1: Array-based Stack

1. Use the previous definitions for the class StaffArray, to implement Stack data structure for objects of class Staff (as described in the previous Tutorial Module) to include functions that perform push and pop operations.
2. **Write** StaffArray class using C++ programming language. Include error handling if necessary.
3. **Write and execute** a program that adds the following Staff into the StaffArray object using Push function.

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Staff number** | **Faculty/Division** | **Age** |
| Zulfakar | 149 | Law | 50 |
| Ooi Ji Woo | 637 | Agriculture | 42 |
| Abdullah | 721 | Economics | 26 |

1. **Display** the names and faculty/division of all elements in StaffArray.

Exercise 2: Stack application

You may use either the array or linked list implementation to form the Post-fix expression as discussed in your class.

Write a program that takes an Infix expression and produce a Postfix expression, by referring to the algorithm in your Stacks slides.

Check if your program produces the correct output for the infix expressions given in the examples.