Programming and Data Structures
Active Learning Activity 8: Data Structures

Activity Objectives

At the end of this activity, students should be able to:

1. Use the generic data structure **`Stack`** from the Java Collection Framework to evaluate postfix expressions using a stack
2. Implement generic classes for the data structures **`ArrayList`** and **`LinkedList`**
3. Test the two data structures using a search method

Activity

Write a program named "**`DataStructures.java`**" to perform the following:

1. Create an instance of the generic class **`Stack`** (**`java.util.Stack`**) for the type **`Integer`** and name it **`postfixStack`**.
2. Prompt the user to enter a postfix expression with operands and operators separated by a space. Extract the tokens from the input postfix expression using the method **`split()`** from class **`String`**.
3. Evaluate the postfix expression using the following algorithm:
    a. Consider one token from the input expression.
    b. If the token is an operand, push it in the stack **`postfixStack`**.
    c. If the token is an operator, pop two values from the stack **`postfixStack`**, perform the operation, and push the result back into the stack **`postfixStack`**.
    d. Repeat from **a.** until all the tokens have been processed
    e. Pop the result of the postfix expression from the stack
    f. If the stack is empty, display the popped value as the result, otherwise display "*`postfix expression malformed.`*"
4. Test your program for the following postfix expressions.

```
12 25 5 1 / / * 8 7 + –
8 2 + 3 * 16 4 /
70 14 4 5 15 3 / * – – / 6 +
```

6. Implement the generic classes **`ArrayBasedList`** and **`LinkedList`** as seen in class. Add the method **`searchIterations()`** to both classes to search for an item in each data structure and return the number of iterations performed by the method to find the item or not. The header of the method is shown below:
   `public int searchIterations(E item)`
7. Analyze the time complexity of the method **`searchIterations()`** in both classes and include it in the code as a comment.
8. In the main method, instantiate **`ArrayBasedList`** and **`LinkedList`** for the type **`String`** and name the instances **`animalArrayList`** and **`animalLinkedList`** respectively.
9. Read the file **`animals.txt`** and add each animal name from the file to the two data structures **`animalArrayList`** and **`animalLinkedList`**.
10. Generate ten (10) random integers with value from **0** to **`animalArrayList.size()`** and search for the random animal names in both **`animalArrayList`** and **`animalLinkedList`** by calling the method **`searchIterations`**. Display the results returned by the method for each animal and for each data structure. Display the average number of iterations for each data structure.
11. Discuss the numbers returned by **`searchIterations`** against the time complexity you performed in step 7.

Important note: The method **`pop()`**, from class **`Stack`**, throws an **`EmptyStackException`** if the stack is empty. Whenever you call **`pop()`**, use a try-catch block to catch such exception. In the catch block, display the message "*`postfix expression malformed.`*".

Submit the following files on coursesite:

`ArrayBasedList.java`,

`LinkedList.java`, and

`DataStructures.java`.

A sample run of the program is shown below. Note that the list of animal names is generated randomly. Therefore, you may get a different list of names every time you run your program.

```
Enter a postfix expression:
8 2 + 3 * 16 4 /
Malformed expression.
Do you want to evaluate another postfix expression? (yes/no):
yes
Enter a postfix expression:
70 14 4 5 15 3 / * - - / 6 +
Result = 8
Do you want to evaluate another postfix expression? (yes/no):
no

Comparing Array List and Linked List

Animal name                 Iterations(ArrayList)   Iterations(LinkedList)
Armadillo                   338                     338
Narwhal                     196                     196
Wombat                      278                     278
Salamander                  304                     304
Ground shark                350                     350
Sheep                       300                     300
Eagle                       288                     288
Siamese fighting fish       61                      61
Hoverfly                    436                     436
Wallaby                     150                     150
-------------------------------------------------------------------
Average # iterations        270                     270
```