Nested Array Flatten

# Summary:

The Nested Array Flatten program will take a text representation of a nested int array and output its flattened value. For a proof of concept, text inputs will be first parsed into a nested array type, which will then be flattened into a single-dimensional int array before finally being shown as a text output.

By: Hyuk Jin Kwon

Hyukjink@sfu.ca

# Methodology and Implementation:

The Nested Array Flatten program is a simple WPF application and is written in C# and Visual Studio Community 2015. All libraries and tools required for the program are available through the .NET Framework.

The program will only take valid text representations of a nested in array. Elements are separated using a single whitespace character, and only brackets (‘[‘ and ‘]’), digits (0-9), and the negative sign (‘-‘) are considered other valid characters.

|  |
| --- |
|  |
| *Fig. 1: A valid input of array elements* |

|  |
| --- |
|  |
| *Fig. 2: An invalid input of array elements* |

Once a valid nested array input is accepted, the array is then converted into a nested Array object. For more details, see Line 179 of TreeViewModel.cs (Found under the “ViewModels” folder of the solution).

*Note: This conversion to a nested array structure is purely for demonstration purposes, and could have been skipped if just a string representation of a nested array suffices.*

|  |
| --- |
|  |
| *Fig. 3: Code snippet of input string being parsed into a nested array structure* |

Once a nested array structure is obtained, it is then ready for flattening. To do so, the nested array structure is then converted into a tree where leaf nodes contain integer values and inner nodes contain arrays. From there, a depth-first search traversal will order the elements and output them into a one-dimensional int array. For more details, see Line 260 of TreeViewModel.cs (Found under the “ViewModels” folder of the solution) and the TreeNode.cs file (Found under the “Models” folder of the solution.

|  |
| --- |
|  |
| *Fig. 4: Code snippet of nested array being converted into a tree structure* |

# Results

The efficiency of this program is based on the depth of the tree structure being created, and the depth-first search algorithm. Hence, the worst-case performance is O(|V| + |E|) in time, where:

1. V is the sum of integer elements and inner arrays
2. E is the sum of all nestings of ints in arrays

Ex:

[1 [2] 3] nests “1” once, “2” twice, “3” once

Hence E = 4.