# IN710 OOSD 2016 Practical 3.2 – Unit Testing

Due Tuesday at 9.00 am

In this practical you are going to build a Stack class[[1]](#footnote-1), and use a unit testing approach to ensure that it works correctly. It will be run against a battery of black box tests to evaluate its robustness.

## Stack

A stack is a collection ADT. It is a LIFO (last in first out) data structure. That is, elements are added to the stack for storage one at a time. When an element is requested from the stack, it returns ***the* *most recently added*** item. Stacks are enormously valuable for implementing a large class of processing algorithms that use backtracking (i.e. work toward a solution until you reach a dead end, then return to the previous choice point and make another choice). In this practical you will build a Stack that holds ***strings***.

## Methods:

Your Stack class must have the following methods (with exactly these prototypes):

* Stack(): Constructor
* void Push(string newString): Adds newString to the stack
* String Pop(): Returns the most recently added string, and removes it from the stack
* String Peek(): Returns the most recently added string, but does not remove it from the stack
* int Count(): Returns the number of strings currently held in the stack
* bool IsEmpty(): Returns true if the stack contains zero elements and false otherwise.

## Decide What to Test:

Fill in the table on the next page with all the situations you need to test to ensure that your stack works correctly. ***Do this before designing or writing the code for your stack***. I have put in a few tests to help you get started. As a guideline, my solution has 28 tests. If you have trouble generating test ideas, try focusing on a single method at a time and thinking of all the different scenarios where that method might occur, and what the resulting system state or action should be. Please include a copy (cell phone image is fine) of this table in your GitHub repo files for this practical.

NB: Some of your tests will look for the throwing of a particular Exception. Use a credible reference manual or website to discover the syntax for this in C#.

|  |  |
| --- | --- |
| **Situation** | **Result** |
| IsEmpty called on a newly created stack | Should return true. |
| IsEmpty called on a stack that used to have items, but from which all the items have been popped | Should return true. |
| IsEmpty called on a stack with items | Should return false. |
| Peek is called on an empty stack | Should throw an exception ***with an appropriate error message.*** |
| Push several items onto stack, then call Peek | Should return last pushed item, but not delete it from the stack. |
| Push n items onto a stack then call Count | Should return n. |
| Count called on empty stack | Should return 0. |
| Count called on a stack that used to have items, but from which all the items have been popped | Should return 0. |
| Peek called on a stack that used to have items but from which all the items have been popped | Should throw an exception. |
| Pop I called on an empty stack | Should throw an exception. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

## Stack Class:

Start a new VS Project and build your stack class. Please name your class Stack and use the method signatures ***exactly*** as described above[[2]](#footnote-2). Think carefully about the appropriate data structure(s) for this class. (**NB: You may not use the .NET Stack class.)**

## Test Project:

Add your test project, and write all of your tests. Work with your class code until all tests pass.

## Verification:

Have your solution in your GitHub repo by the due date. I will run your stack class through my test battery and send you the results.

1. .NET has a built-in Stack class. That is not relevant to this practical. Don't use it. [↑](#footnote-ref-1)
2. Write only return stubs first if you have decided to use the “write the tests first and make sure they all fail” approach. [↑](#footnote-ref-2)