# Activity 4

## Activity Overview

In this activity, you will create another data structure. Unlike the queue (or the stack), this data structure is more restricted in regard to adding or removing objects.

## Activity Instructions

1. This activity requires that you completely implement public generic class ***ArrayStore<T>*** under the namespace ***CSharp.Activity.Datastore****.*

2. Class ***ArrayStore<T>*** must inherit from base class ***AbstractArrayStore<T>*** (add to the project ***AbstractArrayStore.cs*** from **Week 1 Activity Code Files** folder)*.*

Note: ***ArrayStore<T>*** has access to the member methods and variables of its parent class ***AbstractArrayStore<T>*** that are marked either ***public*** or ***protected.***

3. Define appropriate ***ArrayStore*** constructors and ensure proper call of the base class constructors.

4. The test class is provided in the file ***Activity4_Tests.cs***.

5. The following methods must be implemented to complete the ***ArrayStore<T>*** class:
    a. ***public override int Add(T argToAdd)*** – Adds the object 'argToAdd' to the end of the structure on the condition that the structure isn't full. It returns the index if the add is successful or ***NOT_IN_ STRUCTURE*** if the conditions for adding are not met. If ***T*** is reference type, trying to add a ***null*** object to the structure should throw an ***ArgumentNullException****.*

    b. ***public override void RemoveAt(int removeObjectIndex) –*** Removes the object at the specified index. Make sure to compress the array so that no holes remain in the structure. If the index is out of range [0..Count) the method should throw an ***ArgumentOutOfRangeException*****.**

    c. ***public override void Remove(T argToRemove)*** – Removes the object that is equal to ***argToRemove*** from the structure. Make sure to compress the data inside so that there are no

 

holes in the array after a successful remove. (You can make use of the existing base class method ***IndexOf()*** and/or the overrided method ***RemoveAt()***). Throw an ***InvalidOperationException*** if removal is not possible. Throw an ***ArgumentNullException*** if a ***null*** is passed and ***T*** is a reference type.

d. ***public override int Insert(T argToInsert, int indexToInsert) –*** This inserts the object ***argToInsert*** at the specified index. As with ***Add(T argToAdd)*** method, the structure must not be full. Trying to add a ***null*** object should throw an ***ArgumentNullException***, or if inserting with an invalid index (outside range [0..Count)) then should throw an ***ArgumentOutOfRangeException***. Return index if the insert is successful and ***NOT_IN_ STRUCTURE*** if the conditions for insertion are not met.

6. Inform the facilitator upon completion by committing solution **Activity4** to the TFS with comment “Finished Activity 4”.

 