CS 2026 – Spring 2010
Assignment 2
2/5/2010
Due: 2/12/2010 11:59 PM

This assignment will exercise your knowledge of C# features such as enumerators, generics, operator overload, properties, etc.

In this assignment, you'll implement a generic Set<T> class that is parameterized by the type of object it contains. The set will be used as a basic container of objects of type T. Since this is a set, there is only one copy of any object in it. For example: {1,2,2,3} and {1,2,3} are the same set.

You should implement the following methods:

- Set()
  - The default constructor.
- Set(IEnumerable<T> e)
  - A constructor that fills the set with all the elements in some enumerable collection.
- public static Set<T> operator +(Set<T> lhs, Set<T> rhs)
  - This operator implements set union: it should return a new set that contains any item contained in either the lhs or the rhs set.
- public bool Add(T item)
  - This method adds the input element to the set. It returns true if the element is added to the set, and false if the element is already present in the set.
- public bool Remove(T item)
  - This method removes the input element from the set. It returns true if the element is removed to the set, and false if the element was not in the set to begin with.
- public bool Contains(T item)
  - This method returns true if and only if the input element is in the set.
- public Set<T> Filter(F<T> filterFunction)
  - This method takes a delegate of type bool F<T>(T elt) and returns all the elements in the set for which this function returns true.
- Sets should be enumerable by foreach.

Your Set<T> should implement the following properties:

- public int Count
  - This property returns the number of elements in the set. This property should be readable but not writable.
- public bool IsEmpty

    - This property returns true if there are no items in the set. This property should be readable but not writable.

Next, write a generic class SortedSet<T> that is a subclass of Set<T>. Just like its name indicates, the set is sorted(The difference is that, for {2, 3, 1}, enumerating Set would give you 2, 3, 1, and enumerating SortSet set would always produce 1, 2, 3), so the type T should have a constraint that T should implement interface IComparable (Find more information of IComparable on MSDN). Override Add and Remove for SortedSet<T>. Think about one question: does "+" still work for SortedSet? If not, you should implement that as well.

Hints:

- To avoid possible name collisions, define your Set<T> class in namespace CS2026
- I recommend using C#'s List<T> class to store the data in your set. So the List<T> in SortedSet should be sorted. You can find more information about it on MSDN.

Save your program in a single file called Program.cs and submit it to CMS.

Good luck!