Programming the .Net Framework using C#

# Sets

Object Oriented programming in C#

## Instructions

1. Write a class that will be used for creating sets of "int" numbers in the range of 0…1000. Implement the sets with a Boolean array. Define in the class the following operations:

* A constructor without parameters, which initialize the set as an empty set.
* A constructor that gets an undefined numbers of int's and creates a set that contains those values (use "*params*").
* Union – a union operation that receives a set and unites it with the original set. i.e. the set the operation was being done on will represent the union. For Example: given sets s1, s2: s1.union.s2() will cause s1 to be the result of the union.
* Intersect – intersecting operation, that receives a set and performs set intersecting. The set the operation was being done on will represent the intersection.
* Subset – an operation that receives a set and checks whether it represents a subset of the set.
* isMember – an operation that receives a member and checks whether it belongs to the set.
* Insert – an operation that receives a member and inserts it to the set.
* Delete – an operation that receives a member and deletes it from the set
* Override the method **ToString** that inherit from **Object** in the following way:
* ToString – will return a string representation of the set.
* Equals – will check if another set has the same elements.

1. Write a program that checks the sets class in the following way:

* Create 2 sets by drawing 12 random numbers in the range of 0…1000 for each set, and display the sets contents.
* Perform **intersection** and **union** of the two sets and display the received sets.
* Get from the user 3 numbers and create a third set. Check whether this set is a **sub-set** of one of the sets and display the result.
* Get from the user a number and check whether it **is a member** of each set and display the result.
* Get from the user a number and **insert** it to each set, then display them after the addition.
* Get from the user another number, **delete** it from the two sets and display them after the deletion.

## Detailed Instructions

## Features in C# that may be used in this project:

* Classes, Object Based programming, encapsulation

## Important Notes

* Build your classes based on the principles we learned in class.
* Define private and public as needed.
* As always, make sure your code is readable, using meaningful names, indentation and documentation.
* Avoid code duplication
* Do not use Console methods in the set class – make sure your class can be added in the future to an application with Graphical User Interface without changes.
* Make your class interface (public methods) reliable – make sure you define the correct state of the class in your design, bring the class object to that state using the ctor, and carefully return the object to that state in the different methods. Check ranges if needed.