Homework: Common Type System

Problem 1. Student class

- Define a class `Student`, which contains data about a student – first, middle and last name, SSN, permanent address, mobile phone e-mail, course, specialty, university, faculty. Use an enumeration for the specialties, universities and faculties.
- Override the standard methods, inherited by `System.Object`: `Equals()`, `ToString()`, `GetHashCode()` and operators `==` and `!=`.

Problem 2. ICloneable

- Add implementations of the `ICloneable` interface. The `Clone()` method should deeply copy all object's fields into a new object of type `Student`.

Problem 3. IComparable

- Implement the `IComparable<Student>` interface to compare students by names (as first criteria, in lexicographic order) and by social security number (as second criteria, in increasing order).

Problem 4. Person class

- Create a class `Person` with two fields – name and age. Age can be left unspecified (may contain `null` value. Override `ToString()` to display the information of a person and if age is not specified – to say so.
- Write a program to test this functionality.

Problem 5. 64 Bit array

- Define a class `BitArray64` to hold `64` bit values inside an `ulong` value.
- Implement `IEnumerable<int>` and `Equals(…)`, `GetHashCode()`, `[]`, `==` and `!=`.

Problem 6.* Binary search tree

- Define the data structure binary search tree with operations for "adding new element", "searching element" and "deleting elements". It is not necessary to keep the tree balanced.
- Implement the standard methods from `System.Object` – `ToString()`, `Equals(…)`, `GetHashCode()` and the operators for comparison `==` and `!=`.
- Add and implement the `ICloneable` interface for deep copy of the tree.

Remark: Use two types – structure `BinarySearchTree` (for the tree) and class `TreeNode` (for the tree elements). Implement `IEnumerable<T>` to traverse the tree.