# Exercise: Common Type System

This document defines an in-class exercise from the ["OOP" Course @ Software University](https://softuni.bg/courses/oop/).

## Country

Create a class Country containing **name**, **population**, **area** and a list of **cities** (as strings). Override the virtual methods **GetHashCode()** and **Equals()**, implement the **ICloneable** and **IComparable** interfaces, override the **operators** **"=="** and **"!=".**

### Step 1. Create the Class

Each country should have a **name** (**non-empty string**), **population** (**non-negative 64-bit integer**), **area** (**non-negative real number**) and a list of **cities** (**non-repeating strings**). Create the appropriate fields and properties, **validate** the data. All fields are mandatory except the cities.

Example:

|  |
| --- |
| **Example** |
| Country bg = new Country("Bulgaria", 7100000, 111000, "Sofia", "Plovdiv", "Varna");  Country usa = new Country("USA", 300000000, 1200000, "New York", "Los Angeles", "San Francisco");  Country bg2 = new Country("Bulgaria", 8000000, 10);  Country bg3 = new Country("Bulgaria", 8000000, 111000);  Country hr = new Country("Croatia", 8000000, 111000); |

### Step 2. Override GetHashCode() and Equals()

Two countries should be considered equal if they have the same **name**. Override the **Equals()** method to reflect this. Use the country name's hash code to override **GetHashCode()**; you may use other properties as well (like population and area).

### Step 3. Override Operators "==" and "!="

Using the overridden method **Equals()**, override the **operators "==" and "!=".**

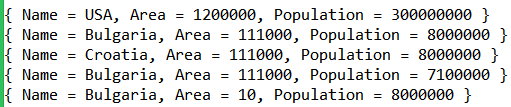
|  |
| --- |
| **Example Result** |
| Console.WriteLine(bg == bg2); // True  Console.WriteLine(bg == usa); // False  Console.WriteLine(bg != bg2); // False  Console.WriteLine(bg != usa); // True |

### Step 4. Implement IComparable<Country>

Countries should be compared by **area** (in **descending** order), then by **population** (in **descending** order) and finally by **name** (**alphabetically** in **ascending** order).

|  |
| --- |
| **Sample Code** |
| var countries = new List<Country> { bg, usa, bg2, bg3, hr };  countries.Sort();  Console.WriteLine(  string.Join(Environment.NewLine, countries  .Select(c => new { c.Name, c.Area, c.Population }))); |

Expected result:



### Step 5. Implement ICloneable

Implement the **Clone()** method to return a **deep copy** of a country. All value fields should be copied by value, while reference types should be copied manually.

|  |
| --- |
| **Sample Code** |
| var bgCopy = bg.Clone() as Country;  bg.Cities.Add("Kaspichan");  Console.WriteLine(string.Join(", ", bg.Cities));  Console.WriteLine(string.Join(", ", bgCopy.Cities)); |

Expected result:

