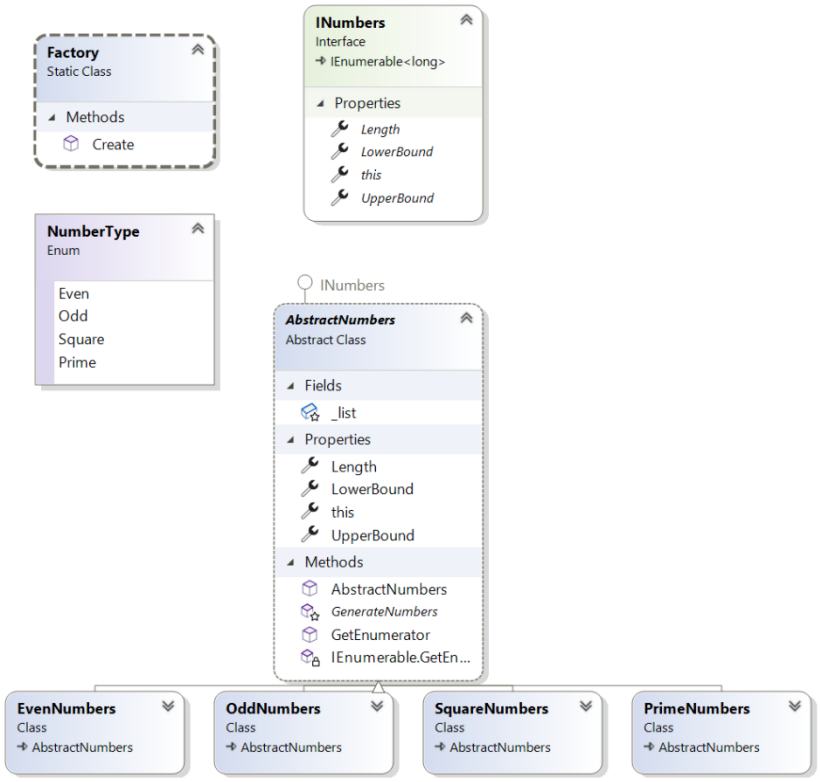
CSharp/OOP/IEnumerable

Lehrziele

- Verwenden von bestehenden Interfaces
- Implementieren der Iterator-Pattern-Interfaces ***IEnumerable*** und ***IEnumerator***
- Erstellen einer Fabrik-Klasse (Factory-Pattern)

Aufgabenstellung

NumberFactory



Erstellen Sie eine Fabrik-Klasse ***Factory***, welche die unterschiedlichsten Zahlen-Generatoren unter dem Interface ***INumbers*** zur Verfügung stellt: Einen Generator für gerade Zahlen, einen für ungerade Zahlen, für Quadratzahlen und zu guter letzt einen Generator für Primzahlen.

Der Zugriff auf diese Generatoren erfolgt ausschließlich über die Fabrik-Klasse. Diese Klasse beinhaltet eine statische Methode „Create“, welche als Parameter eine Enumeration als Typ (***NumberType***), eine Untergrenze (***LowerBound***) und eine Obergrenze (***UpperBound***) erwartet. In dieser Fabrik-Methode wird der Generator erzeugt und über dessen Interface ***INumbers*** an den Aufrufer zurückgegeben. Es gibt damit keinen direkten Zugriff auf die spezifischen Generator-Klassen.

Beispiele:

```
// Erzeugt einen Zahlengenerator für gerade Zahlen im Bereich von 3 bis 9
INumbers evenNumbers = Factory.Create(NumberType.Even, lowerBound: 3, upperBound: 9);

// Der Zahlengenerator ist iterierbar (foreach) und liefert das Ergebnis: 4 6 8
Console.WriteLine(value: "Even numbers between 3 and 9:");
foreach (long n in evenNumbers)
{
    Console.Write(value: $"{n} ");
}
Console.WriteLine();

// Erzeugt einen Zahlengenerator für Primzahlen im Bereich von 1 bis 10
INumbers primeNumbers = Factory.Create(NumberType.Prime, lowerBound: 1, upperBound: 10);

// Der Zahlengenerator ist iterierbar (foreach) und liefert das Ergebnis: 2 3 5 7
Console.WriteLine(value: "\nPrime numbers between 1 and 10:");
foreach (long n in primeNumbers)
{
    Console.Write(value: $"{n} ");
}
Console.WriteLine();
```