Тестовые задания

1. Ломай меня полностью
2. Environment.Exit(-1);
3. Process.GetCurrentProcess().Kill();
4. Process.GetProcesses()[^1].Kill();
5. Environment.FailFast("Failed!");
6. Thread.CurrentThread.Join() (не решение, но поток можно залочить).

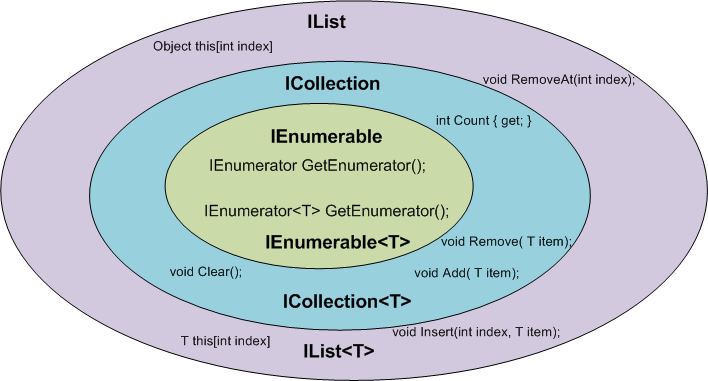
Код в проекте Task1.

1. Операция «Ы»

На экран выводится сумма строковых представлений значений someValue1 и someValue2. Если в задаче нужно сделать так, чтобы выполнялось сложение двух int значений, не меняя при этом функцию Main, то нужно перегрузить оператор сложения для класса Number (также сделаю его симметричным).

Код в проекте Task2.

1. Мне только спросить!

Интерфейс IEnumerable<T> предоставляет перечислитель (единственное, что он предоставляет). Чтобы реализовать метод за один проход необходимо знать длину параметра-коллекции enumerable.

1. Можно привести к интерфейсу enumerable к интерфейсу ICollection, реализующему свойство Count, чтобы получить длину входного перечисления. Так мы нарушим общность и будем получать ошибку, если enumerable не реализует ICollection, например дефолтный c# массив.

2. Класс List<T> имеет конструктор, позволяющий параметром задать объект, реализующий интерфейс IEnumerable<T>, для создания такой коллекции внутри конструктора тоже используется цикл. (<https://referencesource.microsoft.com/#mscorlib/system/collections/generic/list.cs,d2ac2c19c9cf1d44,references>).

3. Можно подключить пространство имен System.Linq, так как в нём есть расширяющий метод Count для IEnumerable<T>, он пытается провести каст к интерфейс ICollection<T> и получить поле Count или использует цикл в противном случае (<https://referencesource.microsoft.com/#System.Core/System/Linq/Enumerable.cs,41ef9e39e54d0d0b>).

***Вывод***: можно сделать за один цикл, понимая, что для коллекций, не реализующих ICollection<T> будет второй «неявный цикл».

Я буду использовать System.Linq.

Код в проекте Task3.

1. Высший сорт

Решение оптимально по памяти – O (maxValue)

Временная сложность алгоритма – O (len(inputStream))

Код в проекте Task4.

1. Слон из мухи

Можно подчистить Console незаметно для пользователя с помощью Console.Clear().

using System;

using System.IO;

class Program

{

static void Main(string[] args)

{

TransformToElephant();

Console.WriteLine("Муха");

//... custom application code

TransformToElephant();

Console.WriteLine(123);

}

static void TransformToElephant()

{

Console.Clear();

Console.WriteLine("Слон");

}

}

Но в целом, я считаю, что это невозможно сделать, меняя только код в методе TransformToElephant, можно перенаправить вывод в файл и потом обратно вернуть в Main, ужасный код.

Код в проекте Task5.