**Все типы размещайте в пространствах имён.**

**Задание 1.** Это задание основывается на задании о диагональной матрице. По классическому определению, числовая диагональная матрица — квадратная матрица, все элементы которой, стоящие вне главной диагонали, равны нулю. В этом же задании вам необходимо создать **универсальный класс** для диагональной матрицы с элементами **любого** типа T (элементы вне главной диагонали равны значению по умолчанию для T).

1. Объект класса хранит только элементы матрицы, расположенные на диагонали. Для этого используется одномерный массив.
2. Класс располагает конструктором для создания матрицы. В конструктор передаётся размер матрицы (например, 5 – для матрицы 5х5). Если аргумент отрицательный – генерируется исключение ArgumentException.
3. Объект класса имеет свойство только для чтения Size – размер матрицы (например, для матрицы 5х5 свойство Size возвращает 5).
4. Для удобства работы класс предлагает индексатор **с двумя индексами** i **и** j. Если значения индексов меньше нуля или больше или равны размеру матрицы – генерируется исключение IndexOutOfRangeException. Если i не равно j: при чтении возвращается значение по умолчанию для типа T, а при записи – ничего не происходит.
5. Класс матрицы содержит событие ElementChanged, которое происходит **после** изменения элемента матрицы, и только если новое значение элемента отличается от старого значения. В событие в качестве дополнительной информации передаются индексы элемента, старое значение элемента, новое значение элемента. **Внимание: используйте стандартные практики работы с событиями!**
6. Создайте метод расширения для диагональной матрицы, выполняющий сложение двух диагональных матрицы. Одним из параметров метода должен быть экземпляр делегата, описывающий, как выполнить сложение двух элементов типа T (это функция с двумя параметрами). Протестируйте работу метода расширения.
7. Реализуйте класс MatrixTracker, который получает диагональную матрицу в качестве параметра конструктора, подписывается на её событие ElementChanged и имеет открытый метод Undo(). При вызове этого метода последнее изменение элемента, сделанное в матрице, откатывается (т.е. отменяется).

**Задание 2.** Создайте **запечатанный неизменяемый** класс для представления *рационального числа*, то есть числа вида n/m, где n – целое число, а m – натуральное число.

1. Рациональное число должно храниться в виде несократимой дроби. Например, число 2/6 должно храниться как Числитель = 1, Знаменатель = 3.
2. Определите в классе конструктор, принимающий в качестве параметров Числитель и Знаменатель рационального числа. Если Знаменатель = 0, генерируется исключение.
3. Переопределите в классе методы Equals(), GetHashCode(), ToString().
4. Реализуйте в классе интерфейс IComparable<T>.
5. Переопределите в классе арифметические операции +, -, \* и /.
6. Переопределите в классе метод явного приведения рационального числа к типу double и метод неявного приведения значения int к рациональному числу.

**Опциональное задание 3 (необязательное к выполнению).** В задании 2 вы создали неизменяемый класс для представления рационального числа. Переделайте этот класс в readonly-структуру. (Не забывайте, что у любой структуры есть конструктор без параметров.) Сравните код вашего класса и структуры. Он стал проще или сложнее? *Правильно работающая структура даст вам два бонусных балла к общей оценке.*