**Задание 1.** *Разреженные числовые матрицы* – это матрицы, в которых только относительно малая часть элементов не равна нулю (<https://en.wikipedia.org/wiki/Sparse_matrix>). Например, если в матрице размером 100x100 только угловые элементы ненулевые (их 4 штуки), то мы определённо можем назвать такую матрицу разреженной. Очевидно, что в целях экономии памяти нулевые элементы разреженной матрицы желательно не хранить. Вам необходимо создать и протестировать класс для представления разреженной матрицы.

1. Создать класс SparseMatrix для представления разреженной целочисленной матрицы (элементы матрицы имеют тип int). Обязательное условие – класс должен иметь конструктор, позволяющий задать размеры матрицы (количество строк и количество столбцов, строго больше нуля), и индексатор с двумя индексами для доступа к элементам матрицы. В качестве внутреннего хранилища вы можете использовать любую коллекцию на ваш выбор. Желательно обеспечить приемлемое быстродействие и разумный расход памяти.
2. Для удобства отладки переопределить в классе матрицы метод ToString().
3. Реализовать в классе SparseMatrix интерфейс IEnumerable<int>. Должны возвращаться **все** элементы матрицы, включая нулевые (делается построчный обход).
4. Описать в классе SparseMatrix метод GetNonzeroElements(). Тип возвращаемого значения – IEnumerable<(int, int, int)>. Метод возвращает набор кортежей вида (*индекс\_i*, *индекс\_j*, *значение*) для всех ненулевых элементов. Индексы должны быть упорядочены по столбцам, затем по строкам – то есть сначала ненулевые элементы из первого столбца, затем – ненулевые элементы из второго столбца, и т. д.
5. Снабдить класс SparseMatrix методом GetCount(x). Он должен возвращать, сколько раз элемент x встречается в матрице. При реализации метода GetCount(x) учтите, что x может быть равен 0.

**Задание 2.**

1. Создайте простой класс для представления *книги*. Сохраните в классе книги *название книги* (строка, не пустая, не null), *дату публикации* (возможно равная null) и *набор авторов книги* (коллекция неповторяющихся строк, возможно пустая).
2. *13-значный ISBN книги* – это строка, имеющая формат XXX-X-XX-XXXXXX-X или XXXXXXXXXXXXX, где X – это цифра 0..9. Создайте свой класс *Каталог* – это коллекция, **подобная словарю**, в которой хранятся книги. Предусмотрите доступ к книге по ключу-строке – по ISBN книги. Учтите важный нюанс – если книгу поместили в каталог, применив ключ 123-4-56-789012-3, то её можно извлечь и по ключу 123-4-56-789012-3, и по ключу 1234567890123. Корректность самого ISBN в этой задаче можете не проверять (не надо проверять контрольные цифры, корректность самого формата проверить желательно – например, при помощи регулярных выражений).

**Задание 3.** Имеется информация о том, как сотрудники некой организации на протяжении 2020 года брали отпуска. Эти данные представлены в виде набора объектов. Каждый объект хранит **имя сотрудника**, **дату первого дня отпуска**, **дату последнего дня отпуска** (в наборе данных имена сотрудников могут повторяться – за исследуемый год сотрудник мог сходить в отпуск несколько раз).

1. Создайте (или используйте существующие) типы для представления отдельной записи об отпуске и набора таких записей.

Следующие пункты реализуйте в виде отдельных методов. Постарайтесь использовать LINQ to Objects.

1. Найдите среднюю продолжительность отпуска в организации.
2. Найдите среднюю продолжительность отпуска для каждого сотрудника (выведите кортежи «имя сотрудника – средняя продолжительность его отпуска»).
3. Сформируйте и выведите набор кортежей «номер месяца года – к-во сотрудников в отпуске в этот месяц». Считать, что сотрудник был в отпуске в определённый месяц, если он в этот месяц провёл в отпуске 1 и более дней.
4. Укажите даты 2020 года, в которые сотрудники не брали отпуска.
5. Проверьте исходные данные на корректность: нет ли в них таких записей, в которых имена сотрудника совпадают, а даты двух отпусков пересекаются.