**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 числовой матрицы.

1. Создать класс SparseMatrix для представления разреженной числовой матрицы (элементы матрицы имеют тип long). 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 стандартную коллекцию на ваш выбор. Желательно обеспечить приемлемое быстродействие при доступе к элементам матрицы и разумный расход памяти.
2. Для удобства отладки переопределить в классе матрицы метод ToString().
3. Реализовать в классе SparseMatrix интерфейс IEnumerable<long>. Должны возвращаться **все** элементы матрицы, включая нулевые (делается построчный обход).
4. Описать в классе SparseMatrix метод GetNonzeroElements(). Тип возвращаемого значения – IEnumerable<(int, int, long)>. 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 аргумент 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 книги* (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).
3. Реализуйте при помощи LINQ to Objects следующие методы работы с *Каталогом*:
4. Получить из каталога набор названий книг, отсортированный по алфавиту.
5. Получить из каталога набор книг (объектов соответствующего класса) по указанному имени автора. Книги должны быть отсортированы по дате публикации.
6. Получить из каталога набор кортежей вида «имя автора – количество его книг в каталоге» для всех авторов.