1.  В классе **Task1** создайте структуру **Point**, представляющую точку в двумерном пространстве с полями для координат X и Y и свойствами для их чтения **X** и **Y**. Переопределите метод **ToString()** для вывода координат точки на консоль в формате “x = 1, y = 2”, где 1 и 2 - значения полей x и y. Напишите метод **Length** для определения расстояния между точкой и переданной в метод как параметр другой точкой (округлять до 3-го знака после запятой). Напишите статический метод, который получает на вход 2 точки и возвращает в качестве строки информацию об их координатах и расстоянии между ними построчно. В конструктор передавать массив из 2х элементов. В классе **Task1** создайте приватное поле для массива точек **Point** и свойство **Points** для чтения этого массива. Переопределите метод **ToString()** для вывода всех элементов массива **Points** на консоль построчно. Напишите метод **Sorting()** для сортировки точек в массиве **Points** по возрастанию расстояния от начала координат (чем выше скорость сортировки, тем больше баллов за выполнение).

2. В класс **Task2** скопируйте код из класса **Task1**. Создайте в нем абстрактный класс **Fourangle**, представляющую четырехугольник с массивом из 4х точек **Point** и абстрактными методами **Length()** и **Area()**. В конструктор передавать массив точек Point. Если их число не равно 4-м, создавать пустой четырехугольник (все точки - нули). Переопределите метод **ToString()** для вывода фигуры на консоль в формате “a with P = b, S = c”, где a - тип фигуры, b - периметр, c - площадь. Создайте от **Fourangle** наследников: **Square**, **Rectangle**. Реализуйте методы для вычисления периметра и площади каждой из этих фигур используя координаты точек. В классе **Task2** создайте массив из фигур и свойство **Fourangles** для его чтения. Напишите метод **Sorting()** для сортировки фигур по уменьшению площади. Переопределите метод **ToString()** для вывода фигур на консоль.

3. В классе **Task3** создайте класс **Grep** с полями для входной строки и выходной строки. Сделайте публичные свойства для чтения этих полей **Input** и **Output**. В конструктор должен передаваться текст и сохраняться как входная строка. После этого нужно найти самую часто встречающуюся букву в тексте и удалить из текста все слова с этой буквой. Сохранить как выходную строку. Переопределите метод **ToString(),** чтобы он возвращал выходную строку.

4. В классе **Task4** создайте интерфейс **IDataSerializer** с методами **Write** и **Read** для записи объекта и чтения объекта по переданному пути. Реализовать интерфейс в классе **DataSerializer** для работы с классом **Grep** (или любым типом) для хранения экземпляра **Grep** в формате JSON. В классе **Task4** создайте интерфейс **ICreator** с 4 методами: **CreateFolder** и **CreateFile**. 2 метода получают путь, где создать папку/файл и название папки/файла. 2 других метода получают путь, где создать папки/файлы и массив названий папок/файлов. Реализовать интерфейс **ICreator** в классе **DataSerializer**. Создайте приватное поле для хранения экземпляра Grep и свойство для чтения этого экземпляра.