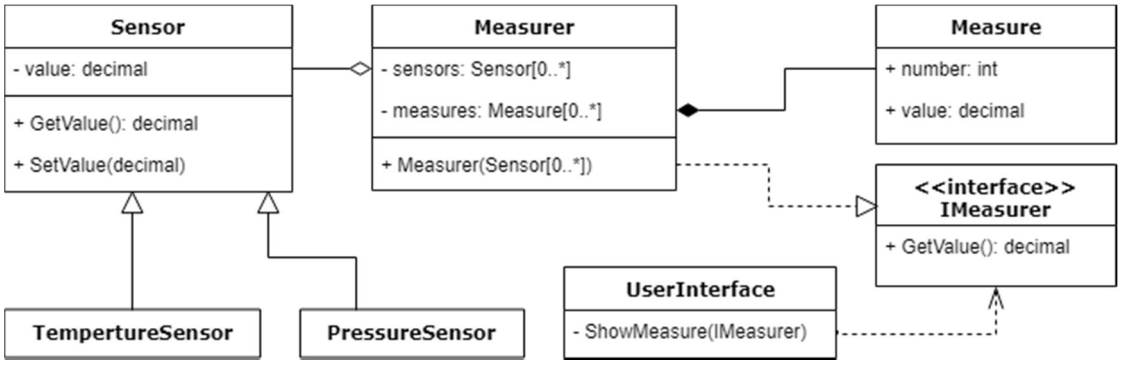
ДЕНЬ 4С. Спецификации

4С.1. Создать декларацию классов C# согласно заданной UML-диаграмме классов

Создать код декларации классов (без их реализации) на языке C# по двум заданным UML-диаграммам классов

Пример

Диаграмма классов:



Код декларации классов:

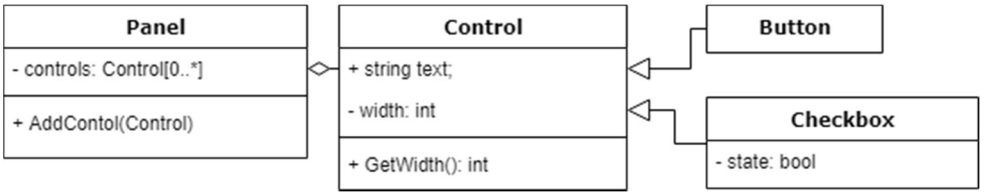
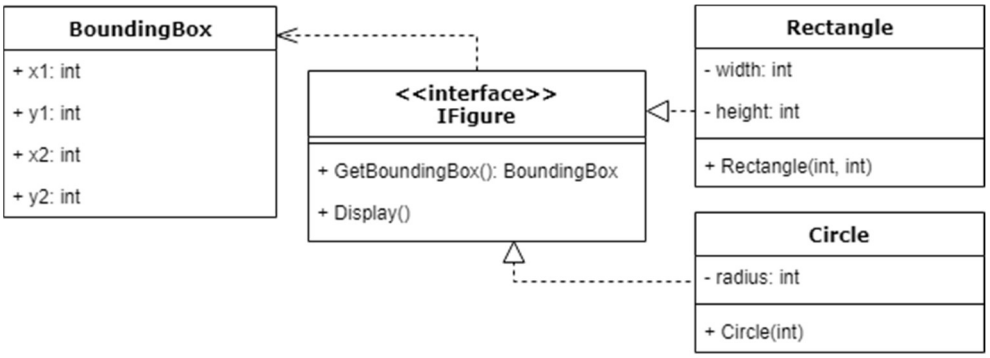
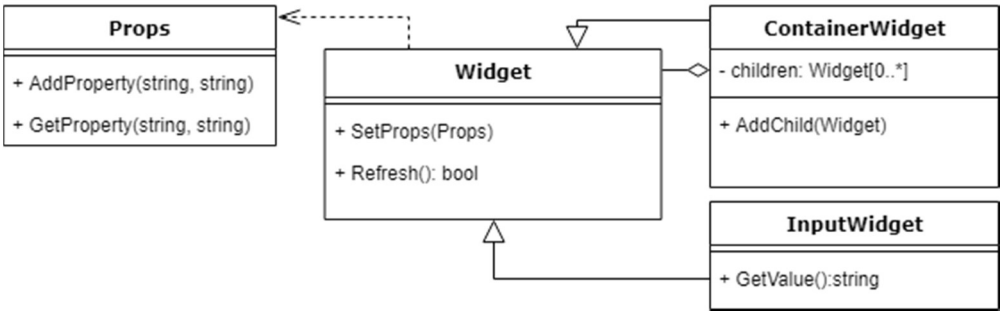
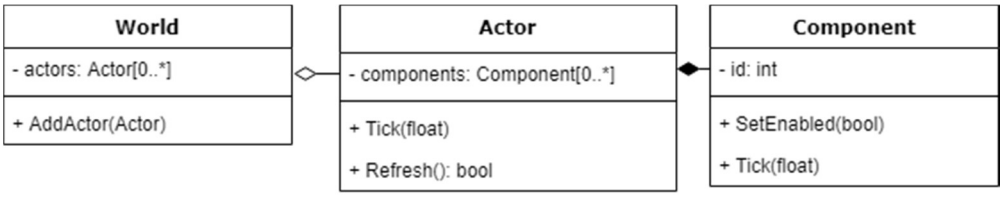
```
public class Sensor
{
    private decimal value;

    public decimal GetValue()
    {
        throw new NotImplementedException();
    }

    public void SetValue(decimal val)
    {
        throw new NotImplementedException();
    }
}

public class TempertureSensor: Sensor
{

}

public class PressureSensor: Sensor
{

}

public class Measure
{
    public int number;
    public decimal value;
}
```

```
public interface IMeasurer
{
    decimal GetValue();
}

public class Measurer: IMeasurer
{
    private List<Sensor> sensors;
    private List<Measure> measures;

    public Measurer(List<Sensor> sens)
    {
        throw new NotImplementedException();
    }

    public decimal GetValue()
    {
        throw new NotImplementedException();
    }
}

public class UserInterface
{
    private void ShowMeasure(IMeasurer)
    {
        throw new NotImplementedException();
    }
}
```

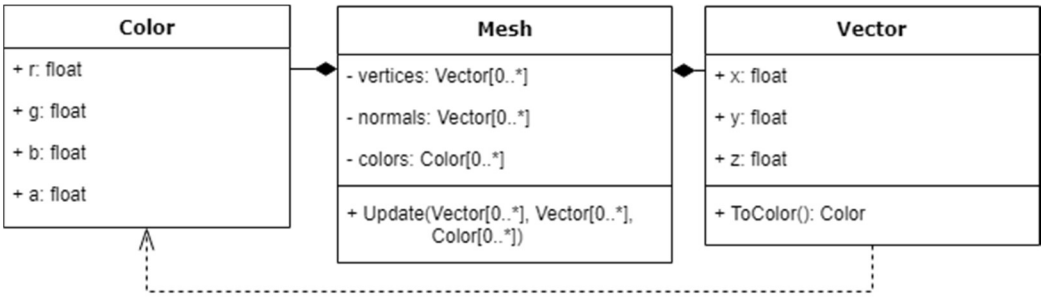
Варианты

№	Диаграммы		№	Диаграммы		№	Диаграммы		№	Диаграммы	
	1	2		1	2		1	2		1	2
1	Д8	Д14	**6**	Д9	Д12	**11**	Д5	Д11	**16**	Д3	Д4
2	Д1	Д3	**7**	Д5	Д8	**12**	Д2	Д5	**17**	Д3	Д11
3	Д2	Д14	**8**	Д2	Д7	**13**	Д3	Д12	**18**	Д6	Д13
4	Д6	Д7	**9**	Д9	Д13	**14**	Д5	Д12	**19**	Д2	Д4
5	Д9	Д14	**10**	Д4	Д8	**15**	Д6	Д8	**20**	Д12	Д13

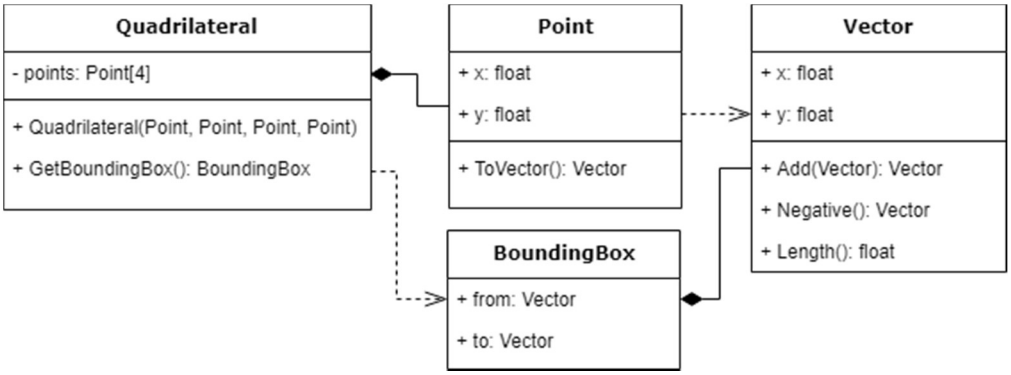
Диаграммы

№	Диаграмма
Д1	
Д2	
Д3	
Д4	

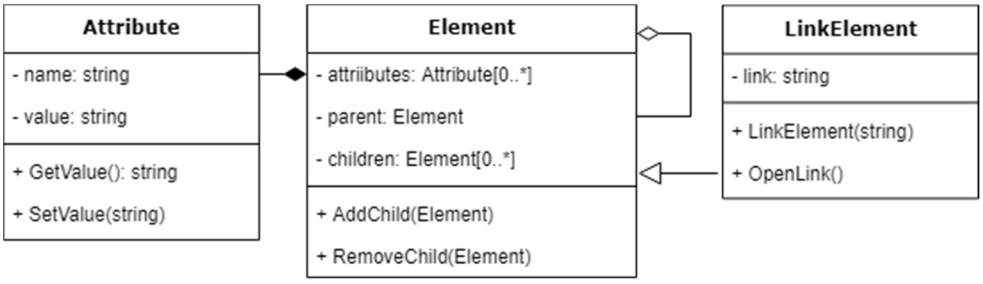
Д5


Color
+ r: float
+ g: float
+ b: float
+ a: float
Mesh
- vertices: Vector[0..*]
- normals: Vector[0..*]
- colors: Color[0..*]
+ Update(Vector[0..*], Vector[0..*], Color[0..*])
Vector
+ x: float
+ y: float
+ z: float
+ ToColor(): Color

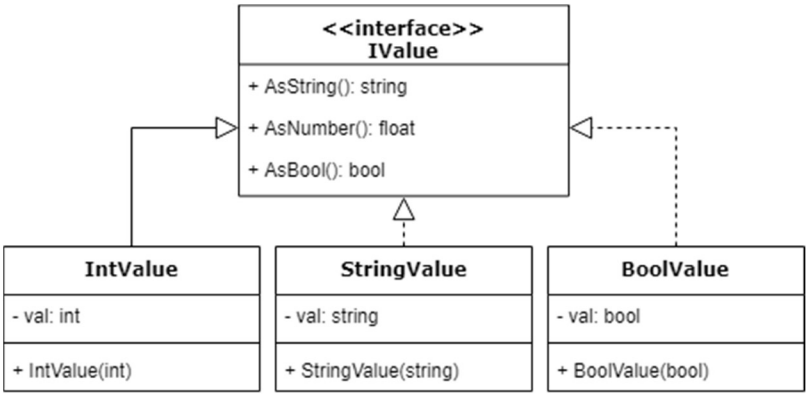
Д6


Quadrilateral
- points: Point[4]
+ Quadrilateral(Point, Point, Point, Point)
+ GetBoundingBox(): BoundingBox
Point
+ x: float
+ y: float
+ ToVector(): Vector
Vector
+ x: float
+ y: float
+ Add(Vector): Vector
+ Negative(): Vector
+ Length(): float
BoundingBox
+ from: Vector
+ to: Vector

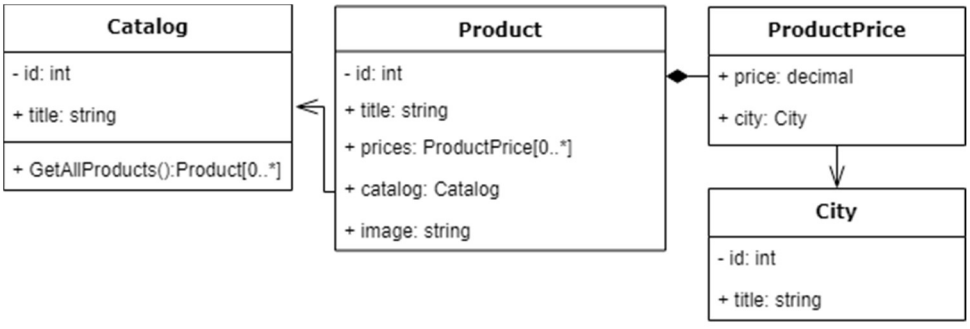
Д7


Attribute
- name: string
- value: string
+ GetValue(): string
+ SetValue(string)
Element
- attriibutes: Attribute[0..*]
- parent: Element
- children: Element[0..*]
+ AddChild(Element)
+ RemoveChild(Element)
LinkElement
- link: string
+ LinkElement(string)
+ OpenLink()

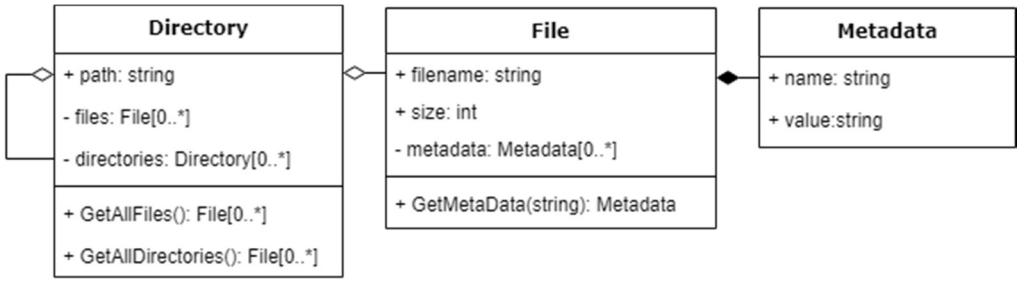
Д8


<<interface>>
IValue
+ AsString(): string
+ AsNumber(): float
+ AsBool(): bool
IntValue
- val: int
+ IntValue(int)
StringValue
- val: string
+ StringValue(string)
BoolValue
- val: bool
+ BoolValue(bool)

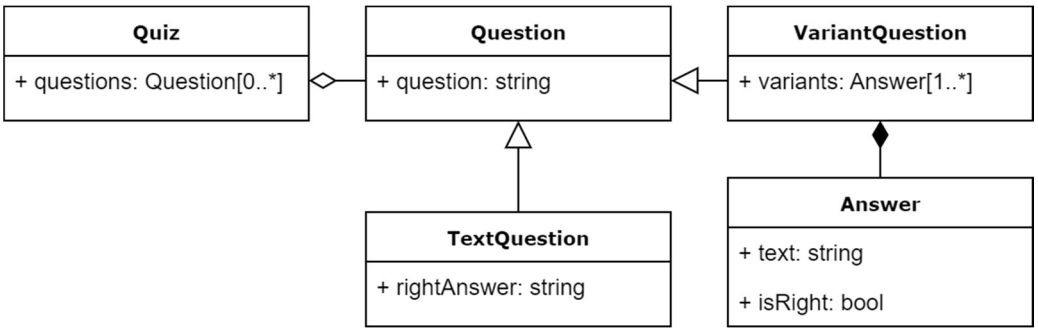
Д9


Catalog
- id: int
+ title: string
+ GetAllProducts():Product[0..*]
Product
- id: int
+ title: string
+ prices: ProductPrice[0..*]
+ catalog: Catalog
+ image: string
ProductPrice
+ price: decimal
+ city: City
City
- id: int
+ title: string

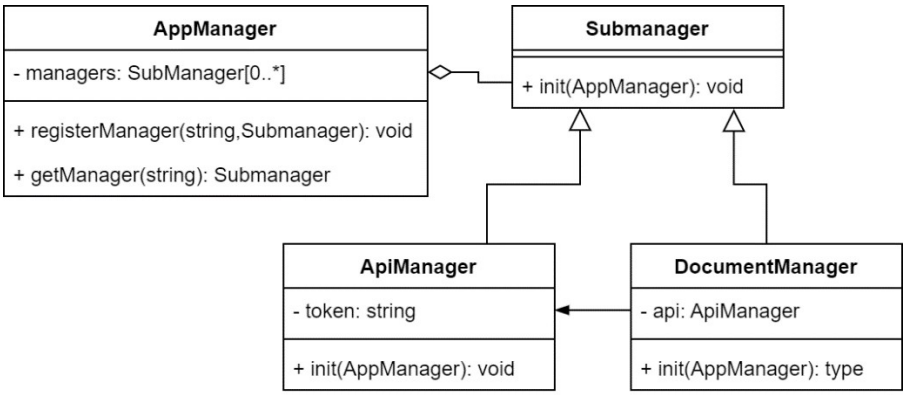
Д10



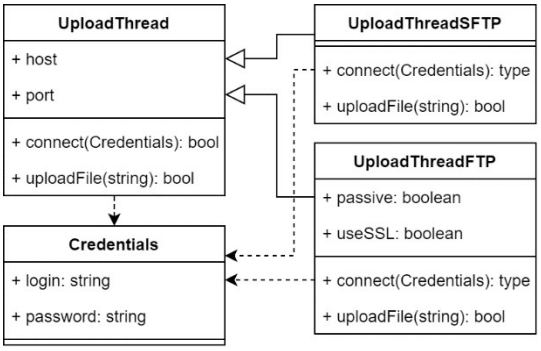
Д11



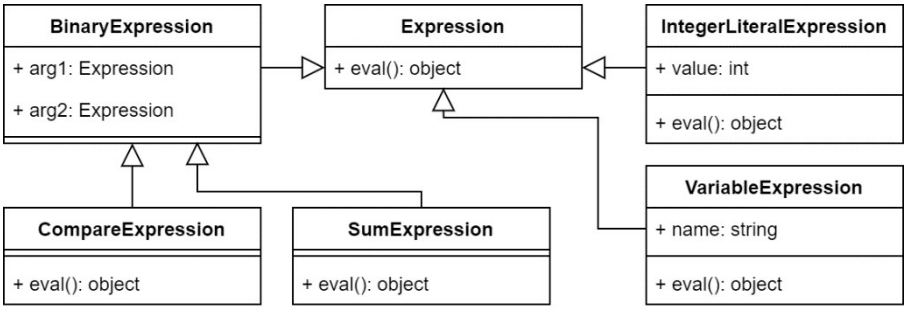
Д12



Д13



Д14



4С.2. Расширить функционал заданного класса, используя наследование

Вам дан код класса, выполняющий преобразование текста

```
class TextTransformer
{
    protected string[] lines;
    public TextTransformer(string text)
    {
        lines = text.Split('\n');
    }

    virtual public string TransformWord(string word, int lineIndex, int wordIndex,
                                        int wordCount)
    {
        return System.Text.RegularExpressions.Regex.Replace(word, "[а-я]", "*");
    }

    virtual public string TransformLine(string line, int lineIndex)
    {
        string[] words = line.Split(' ');
        for (int i = 0; i < words.Length; i++)
        {
            words[i] = TransformWord(words[i], lineIndex, i, words.Length);
        }
        return string.Join(" ", words);
    }

    public string GetResult()
    {
        string[] result = new string[lines.Length];
        for (int i = 0; i < lines.Length; i++)
        {
            result[i] = TransformLine(lines[i], i);
        }
        return string.Join("\n", result);
    }
}
```

Ваша задача – поменять способ трансформации текста, создав класс, который наследуется от *TextTransformer*. В созданном подклассе для изменения поведения Вы должны переопределить методы *TransformWord* и (или) *TransformLine.* Все дополнительно созданные члены класса должны иметь спецификатор доступа private. Метод *GetResult* должен остаться без изменения.

Изменять класс *TextTransformer* недопустимо

В каждом варианте задания помимо использования нового преобразования для некоторых частей текста необходимо сохранить исходное (см. варианты).

Для демонстрации работы Вам необходимо разработать программу (консольную или оконную, на Ваш выбор) со следующей логикой:

5. пользовать вводит многострочный текст
6. создается экземпляр вашего подкласса и в конструктор передается исходный текст
7. с помощью вызова метода *GetResult* у экземпляра получается преобразованный текст
8. преобразованный текст выводится на экран

Пример

Новое преобразование: удалить каждое второе слово.

Исходное преобразование: каждая вторая строка

```
class DerivedTextTransformer: TextTransformer
{
    public DerivedTextTransformer(string text) : base(text)
    {
    }

    override public string TransformWord(string word, int lineIndex, int wordIndex,
                                         int wordCount)
    {
        if (lineIndex % 2 == 1)
        {
            return base.TransformWord(word, lineIndex, wordIndex, wordCount);
        }
        return wordIndex % 2 == 1 ? "" : word;
    }
}
```

Исходный текст:

```
В томленьях грусти безнадежной
В тревогах ШУМНОЙ суеты,
Звучал мне долго голос нежный
И снились МИЛЫЕ ЧЕРТЫ.
```

Преобразованный текст:

```
В  грусти
В ******** ШУМНОЙ *****
Звучал  долго  нежный
И ******* МИЛЫЕ ЧЕРТЫ.
```

Варианты

№	Новое преобразование	Исходное преобразование для
1	Удалить все гласные	Каждое слово с дефисом
2	Перевести первую букву каждого слова в верхний регистр	Каждое слово короче 8 символов
3	Перевести в верхний регистр все согласные	Каждое слово длиннее 10 символов
4	Удалить все слова размером больше, чем из 5 букв	Каждая строка длиннее 3 слов
5	Поменять местами слова в каждой паре слов: аб вг де жз -> вг аб жз де	Каждая строка длиннее 4 слов
6	Перевести в верхний регистр каждую вторую букву слов: абв гдеж -> аБв гДеЖ	Каждая строка короче 5 слов
7	Перевести в верхний регистр все гласные	Каждое слово короче 5 символов
8	Удалить первое и последнее слова в строках	Каждая строка, в которой присутствуют слова длиннее 6 символов
9	Перевернуть каждое слово в строке (абг деж -> гба жед)	Каждое слово, начинающееся с большой буквы
10	Удалить все слова, в которых меньше 5 согласных	Каждое слово, содержащее цифры
11	Удалить все согласные	Каждое слово длиннее 5 символов

12	Удалить все слова, в которых больше 5 согласных	Каждая строка короче 4 слов
13	Удалить знаки препинания	Каждое слово, заканчивающееся на согласную
14	Удалить все слова без гласных букв	Каждое слово, заканчивающееся на гласную
15	Перевести все буквы первого слова каждой строке в верхний регистр	Каждая строка, которая длиннее по количеству слов, чем предыдущая
16	Удалить все слова размером меньше, чем из 5 букв	Каждая строка, содержащая два одинаковых слова подряд
17	Удалить все слова без согласных букв	Каждая строка, заканчивающаяся знаком препинания
18	Удаление слов с цифрами	Каждое слово, которое совпадает с предыдущим словом в строке
19	Перевернуть строки (абг деж -> жед гба)	Каждая строка, начинающаяся с большой буквы
20	Перемешать в случайном порядке все слова в строке	Каждая строка, в которой присутствуют слова короче 5 символов

Примечание: под словами в данном задании понимаются подстроки, состоящие из любого количества любых символов (кроме пробела), разделенные одним пробелом

4С.3. Разработать модульный тест для функции проверки правильности ввода данных, возвращающей код ошибки

Необходимо разработать функцию проверки правильности ввода данных для заданной записи и модульный тест, ее проверяющий. Функция проверки правильности ввода должна принимать на вход объект записи и возвращать, либо 0, если ошибок нет, либо номер поля с ошибкой (начиная с 1)

Решение должно состоять из двух проектов:

- *Библиотека классов (.NET Framework)*, в которой должен быть класс с функцией проверки правильности ввода данных и класс, представляющий собой запись
- *Проект модульного теста (.NET Framework)*, в котором должен быть реализован модульный тест.

Модульный тест должен проверить, как положительный исход функции, так и все варианты отрицательного исхода.

Варианты

№	Поля записи и их проверка		
	Поле 1	**Поле 2**	**Поле 3**
1	Фамилия (строка): не меньше 3 букв	Число ролей (целое число): не меньше 0 и не больше 9999	Гонорар в млн. руб. (вещественное число): не меньше 0
2	Название товара (строка): не меньше 10 символов	Количество на складе (целое число): от 0 до 999 включительно	Количество зарезервированных (целое число): не меньше 0
3	Название (строка): не больше 255 символов	Число сезонов (целое число): больше или равно одному	Год выпуска первого сезона (целое число): от 2000 до текущего года включительно

4	Фамилия (строка): не меньше 3 букв	Должность (строка): только русские буквы	Оклад в руб (целое число): больше 0
5	Номер телефона (строка): только цифры, скобки, знаки тире, плюса и пробела	Имя оператора (строка): не меньше 3 символов	баланс в копейках (целое число): не меньше -9999999
6	Фамилия (строка): не меньше 3 букв	Рост (вещественное число): не больше 300	Вес (вещественное число): больше 30
7	Адрес сайта (строка): только латинские буквы, точка, тире и цифры	Число посетителей (целое число): не меньше числа уникальных посетителей	Число уникальных посетителей (целое число): не меньше 0
8	Название цеха (строка): только буквы и цифры	План выпуска деталей (целое число): не больше 1000	Фактический выпуск деталей (целое число): не меньше 0
9	Компания (строка), не меньше 10 символов	Сумма поступлений в млн. руб. (вещественное число): не меньше 0 и не больше 1000000	Сумма списаний в млн. руб. (вещественное число): не меньше 0
10	Адрес отправления (строка): не меньше 30 символов	Адрес доставки (строка): не равен адресу отправления	Вес (вещественное число): не больше 90
11	Фамилия (строка): не меньше 3 букв	Номер группы (строка): только цифры, знак дефиса и русские буквы	Номер в группе (целое число): больше 0
12	Название материала (строка): не меньше 10 символов	Объем (вещественное число): не больше 1000	Вес (вещественное число): не меньше 0
13	Тема письма (строка): не больше 1024 символов	Адресат (строка): только латинские буквы, знак «@», точка, тире и цифры	Число слов (целое число): не меньше 1
14	Производитель (строка): не меньше 10 символов	Объем выпуска (вещественное число): не меньше 0	Средняя цена (вещественное число): от 1 до 99999 включительно
15	Номер заказа (строка): только латинские буквы и цифры	Описание (строка): не меньше 200 символов	Сумма заказа (целое число): не меньше 0
16	Фамилия (строка): не меньше 3 букв	Год поступления (целое число): не меньше 1952 и не больше текущего года	Средний балл (вещественное число): от 0 до 5 включительно
17	Автомобильный номер (строка): только цифры и буквы А, В, Е, К, М, Н, О, Р, С, Т, У, Х	Год выпуска (целое число): от 1980 до текущего года включительно	Пробег в км (целое число): больше или равно 0
18	Дисциплина (строка): только русские буквы, знаки дефиса и пробела	номер курса (целое число): от 0 до 6 включительно	количество часов (целое число): от 0 до 9999 включительно
19	Фамилия (строка): не меньше 3 букв	Оценка за теорию (целое число): не меньше 3	Оценка за практику (целое число): не больше 5
20	Город (строка): не больше 255 символов	Улица (строка): только русские буквы, знак дефиса, пробела и цифры	Номер дома (число): больше 0

4С.4. Прочитать метаданные из файла заданного бинарного формата

Вам необходимо разработать программу, которая считывает заданное поле метаданных (см. варианты) из указанного бинарного формата и выводит его на экран.

Обрабатываемый файл указывается пользователем при запуске программы, которая может быть, на Ваш выбор, консольной или оконной.

Возможные варианты бинарных форматов:

Формат	**Ссылки на описание**
BMP	https://ru.wikipedia.org/wiki/BMP
PNG	https://habr.com/ru/post/130472/
JPEG	https://ru.wikipedia.org/wiki/JPEG, https://habr.com/ru/post/102521/
GIF	http://home.onego.ru/~chiezo/gif.htm, https://habr.com/ru/post/274917/
WAV	http://microsin.net/programming/pc/wav-format.html, https://audiocoding.ru/articles/2008-05-22-wav-file-structure/
AVI	https://ru.wikipedia.org/wiki/Audio_Video_Interleave

Для анализа содержимого файла вы можете воспользоваться бесплатным онлайн-ресурсом https://hexed.it/

В папке https://github.com/Nordth/istu-priklad-practic-2023/tree/main/media/day4/Task_ReadBinaryFiles приведены примеры тестовых файлов, метаданные которых приведены в следующей таблице:

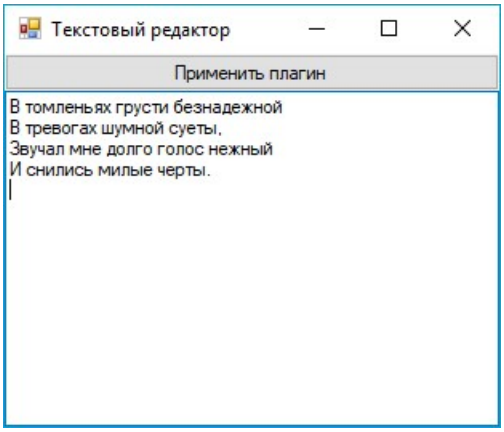
Файл	**Поле метаданных**	**Значение**
sun.bmp	ширина	64
	высота	58
	битность растра	24
bird.png	ширина	16
	высота	16
	битовая глубина цвета	8
	наличие альфа-канала	да
lenna.jpg	ширина	220
	высота	220
	количество каналов	3
cat.gif	ширина	64
	высота	67
	количество кадров	52
	наличие глобальной палитры	да
laugh.wav	число каналов	2
	частота дискретизации	48000
	количество бит в сэмпле	16
coding.avi	ширина	480
	высота	270
	число кадров в секунду	10

Ваша программа должна обрабатывать любые файлы заданного формата. В папке файлы приведены исключительно для примера

Варианты

№	Формат	Поле метаданных
1	GIF	наличие глобальной палитры
2	JPEG	ширина
3	GIF	ширина
4	BMP	ширина
5	PNG	высота
6	GIF	высота
7	JPEG	количество каналов
8	WAV	количество бит в сэмпле
9	PNG	наличие альфа-канала
10	BMP	битность растра
11	AVI	высота
12	WAV	частота дискретизации
13	GIF	количество кадров
14	PNG	ширина
15	AVI	ширина
16	JPEG	высота
17	AVI	число кадров в секунду
18	WAV	число каналов
19	PNG	битовая глубина цвета
20	BMP	высота

4С.5. Разработать плагин к текстовому редактору в виде DLL-библиотеки



Необходимо создать DLL-библиотеку, которая будет использоваться в качестве плагина к текстовому редактору. Плагин должен осуществлять заданную операцию над входным текстом и возвращать результирующий текст.

Текстовый редактор уже написан, его код и сборка лежат по адресу https://github.com/Nordth/istu-priklad-practic-2023/tree/main/media/day4/Task_TextEditor. В данном проекте изменять ничего не нужно. Вам нужно создать **новый** проект типа «*Библиотека классов (.NET Framework)*». В этом проекте вам необходимо в пространстве имен *Task_TextEditor* создать класс *Plugin* с методом *Execute*, который принимает на вход строку и возвращает изменённый текст (см. пример).

После сборки вашего проекта запустите текстовый редактор, введите текст и нажмите кнопку «Применить плагин». Откроется окно выбора файла, выберите в нем вашу скомпилированную библиотеку (DLL-файл). После выбора создастся экземпляр вашего класса *Plugin*, вызовется метод *Execute* и результат запишется в исходное текстовое поле.

Пример

Преобразование: вырезать все пробелы

Код плагина:

```
namespace Task_TextEditor
{
    public class Plugin
    {
        public string Execute(string input)
        {
            return input.Replace(" ", "");
        }
    }
}
```

Варианты

№	Преобразование	№	Преобразование
1	Перевернуть каждое слово в строке (абг деж -> гба жед)	11	Перевести в верхний регистр все согласные
2	Удалить все слова размером меньше, чем из 5 букв	12	Перемешать в случайном порядке все слова в строке
3	Удалить все согласные	13	Перевести в верхний регистр каждую вторую букву слов: абв гдеж -> аБв гДеЖ
4	Перевернуть строки (абг деж -> жед гба)	14	Перевести все буквы первого слова каждой строке в верхний регистр
5	Перевести первую букву каждого слова в верхний регистр	15	Удалить все слова размером больше, чем из 5 букв
6	Перевести в верхний регистр все гласные	16	Удаление каждого второго слова
7	Поменять местами слова в каждой паре слов:аб вг де жз -> вг аб жз де	17	Удалить все слова без гласных букв
8	Удалить все гласные	18	Удалить первое и последнее слова в строках
9	Удалить все слова без согласных букв	19	Удалить все слова, в которых меньше 5 согласных
10	Удалить все слова, в которых больше 5 согласных	20	Удалить знаки препинания