Вариант 1

В библиотеке классов CWLibrary описать обобщённый класс «двойка» **Pair<T, U>**, реализующий интерфейс **IComparable**:

- **T item1**, **U item2** – поля класса, представляющие элементы двойки.
- Конструктор с двумя параметрами типов **T** и **U** – связывает поля с объектами.
- Метод **CompareTo()** сравнивает двойки по значению элемента типа **T**.
- Переопределенный метод **ToString().**

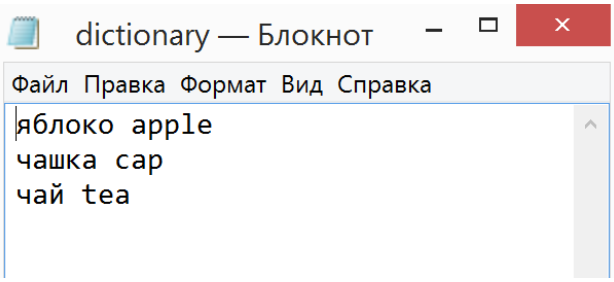
Описать класс **Dictionary**, хранящий список русских и английских слов:

- **int locale** – поле класса, определяющее тип словаря: русско-английский (0) или англо-русский (1).
- Закрытый список **words** хранит элементы типа **Pair<string, string>** (русское слово, английское слово).
- Конструкторы класса (по умолчанию, принимающий список типа **Pair<string, string>**). Локаль генерируется случайно в конструкторе.
- Метод **Add (Pair<string, string>)** для добавления нового элемента в список words. А также перегруженный метод **Add (string, string).**
- Итератор, позволяющий перебирать с помощью цикла foreach все пары словаря в алфавитном порядке локали (т.е. если локаль 0, то перебираем в алфавитном порядке по русским словам, если локаль 1, то перебираем в алфавитном порядке по английским словам);
- Именованный итератор, позволяющий перебирать с помощью цикла foreach все пары словаря в алфавитном порядке локали (т.е. если локаль 0, то перебираем в алфавитном порядке по русским словам, если локаль 1, то перебираем в алфавитном порядке по английским словам), начинающиеся с определенной буквы (задается в качестве параметра);
- Метод void **MySerialize(string path)** для выполнения бинарной сериализации текущего объекта класса **Dictionary**, path – путь к файлу, лежащему в одной папке с *.exe.
- Статический метод Dictionary **MyDeserialize(string path)** для выполнения бинарной десериализации объекта.

В основной программе

- Создать программным путём в каталоге вашего решения текстовый файл, каждая строка которого содержит два слова (первое слово – слово на русском языке, второе слово – слово на английском языке). Число строк в файле **N** задается пользователем с клавиатуры. Имя создаваемого файла – **dictionary.txt.**
  Hint: Помните про декомпозицию

  Пример созданного файла из 3 строк.



- Создать объект dict типа Dictionary. С помощью метода Add добавить в список данные, загруженные из файла. Порядок элементов в списке должен полностью совпадать с порядком, определенном в файле.
- Заполненный объект dict сериализовать в файл out.bin, распложенный в папке с exe-модулем проекта. Затем прочитать ранее сериализованный файл out.bin в новый объект dict2, вывести информацию о данных объекта на экран, используя оператор foreach. Затем (также с помощью оператора foreach) вывести на экран слова, начинающиеся с буквы <*symbol*> (сгенерировать случайно в зависимости от локали).

  Цикл повторения решения не предусматривать, необходимо произвести обработку исключений и проверку корректности ввода данных. Слово – это последовательность букв без пробелов и иных символов.

Вариант 2

В библиотеке классов CWLibrary описать обобщённый класс «двойка» **Pair<T, U>**, реализующий интерфейс **IComparable**:

- **T item1**, **U item2** – поля класса, представляющие элементы двойки.
- Конструктор с двумя параметрами типов **T** и **U** – связывает поля с объектами.
- Метод **CompareTo()** сравнивает двойки по значению элемента типа **T**.
- Переопределенный метод **ToString().**

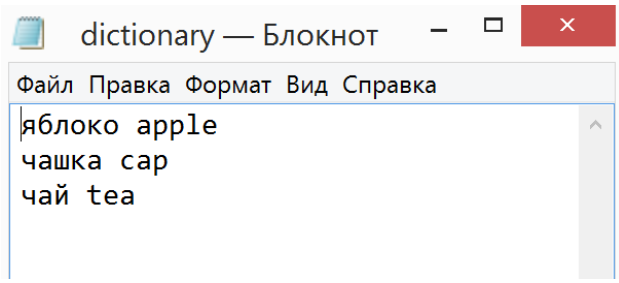
Описать класс **Dictionary**, хранящий список русских и английских слов:

- **int locale** – поле класса, определяющее тип словаря: русско-английский (0) или англо-русский (1).
- Закрытый список **words** хранит элементы типа **Pair<string, string>** (русское слово, английское слово).
- Конструкторы класса (по умолчанию, принимающий список типа **Pair<string, string>**). Локаль генерируется случайно в конструкторе.
- Метод **Add (Pair<string, string>)** для добавления нового элемента в список words. А также перегруженный метод **Add (string, string).**
- Итератор, позволяющий перебирать с помощью цикла foreach все пары словаря в алфавитном порядке локали (т.е. если локаль 0, то перебираем в алфавитном порядке по русским словам, если локаль 1, то перебираем в алфавитном порядке по английским словам);
- Именнованный итератор, позволяющий перебирать с помощью цикла foreach все пары словаря в алфавитном порядке локали (т.е. если локаль 0, то перебираем в алфавитном порядке по русским словам, если локаль 1, то перебираем в алфавитном порядке по английским словам), имеющие заданную длину (задается в качестве параметра). Длину смотрим у слова, чья локаль задана. Если, например, локаль 0, то смотрим по длине русского слова;
- Метод void **MySerialize(string path)** для выполнения бинарной сериализации текущего объекта класса **Dictionary**, path – путь к файлу, лежащему в одной папке с *.exe.
- Статический метод Dictionary **MyDeserialize(string path)** для выполнения бинарной десериализации объекта.

В основной программе

- Создать программным путём в каталоге вашего решения текстовый файл, каждая строка которого содержит два слова (первое слово – слово на русском языке, второе слово – слово на английском языке). Число строк в файле **N** задается пользователем с клавиатуры. Имя создаваемого файла – **dictionary.txt.**
  Hint: Помните про декомпозицию

  Пример созданного файла из 3 строк.



- Создать объект dict типа Dictionary. С помощью метода Add добавить в список данные, загруженные из файла. Порядок элементов в списке должен полностью совпадать с порядком, определенном в файле.
- Заполненный объект dict сериализовать в файл out.bin, распложенный в папке с exe-модулем проекта. Затем прочитать ранее сериализованный файл out.bin в новый объект dict2, вывести информацию о данных объекта на экран, используя оператор foreach. Затем (также с помощью оператора foreach) вывести на экран слова, имеющие длину в диапазоне <d>, где <d> - случайное число в диапазоне [2; 10].

  Цикл повторения решения не предусматривать, необходимо произвести обработку исключений и проверку корректности ввода данных. Слово – это последовательность букв без пробелов и иных символов.