Беляков А.Ю.

Разработка продукционной Экспертной системы

Традиционно под экспертной системой понимают компьютерную программу, предназначенную для выдачи ответов по ограниченному ряду вопросов в рамках определенной предметной области знаний. Работа экспертной системы основана на модели знаний, как правило, продукционной или фреймовой. Структурно экспертная система может быть представлена тремя самостоятельными блоками:

– базой знаний,

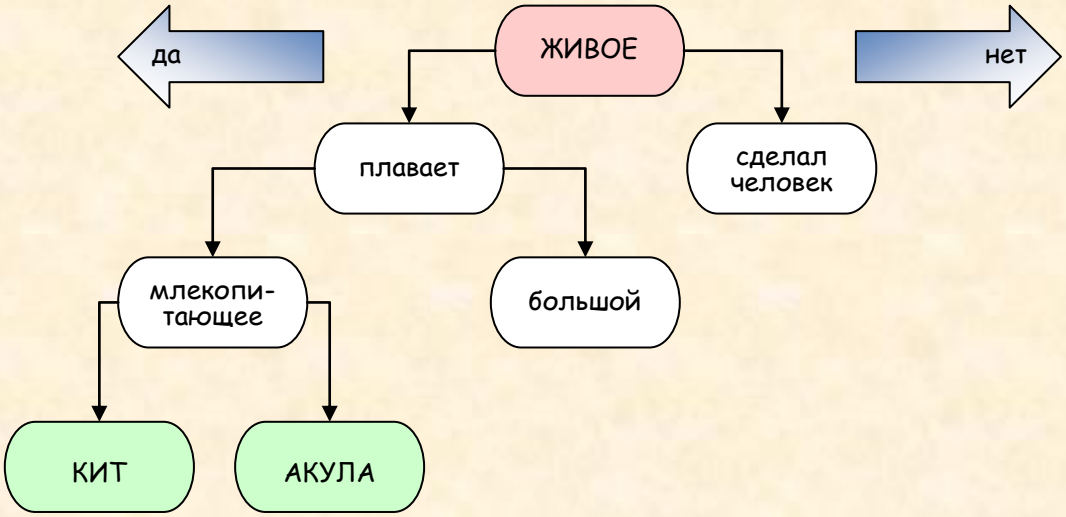
– машиной вывода,

– интерфейсом пользователя.

Наименее технологичным является этап создания ***базы знаний*** ввиду сложности формализации знаний. Знания являются слабо структурированной или вовсе неформализованной информацией. Для того чтобы представить знания в продукционной форме, знания нужно сначала хотя бы частично формализовать. На этом этапе часто приходится идти на компромисс, в чём-то упрощая и снижая достоверность модели предметной области. В нашем случае для простоты рассмотрения будем использовать модель знаний о некоторых объектах живой и неживой природы:

```
живое
    плавает
        млекопитающее
            _кит
            _акула
        большой
            косолапый
                _медведь
                длинная шея
                    _жираф
                    _слон
            _крыса
    сделал человек
        есть колеса
            два колеса
                есть двигатель
                    _мотоцикл
                    _велосипед
                _автомобиль
            работает, если включить в розетку
                _телевизор
                _кубик Рубика
        _камень
```

Данная модель знаний интерпретирует бинарное дерево, в корне которого располагается первый и ключевой вопрос, делящий дерево на два поддерева, в свою очередь являющиеся двоичными деревьями поиска:



Модель знаний можно хранить в любом формате (XML, Excel, txt), но в нашем случае, чтобы обойтись малыми средствами и иметь возможность простого редактирования базы данных ограничимся текстовым файлом. Чтобы заменить графические обозначения на текстовые аналоги оговорим некоторые условности:

– первая строка текстового файла содержит корень дерева и находится на нулевом уровне погружения;

– дерево бинарное, поэтому у вершины не может быть более двух ветвей;

– каждая ветвь оканчивается вершиной одного из двух видов: узел или листок;

– у узла могут быть ветви, у листка – нет (это и есть искомое решение);

– каждая новая ветвь добавляет уровень погружения, для которого в текстовом файле применяется специальный символ, например, табуляция;

– для обозначения листка в текстовом файле применяется специальный символ, например, символ подчёркивания ‘_’.

Следующий структурный элемент экспертной системы – машина вывода. Сформируем её в виде библиотеки классов, откомпилированной в динамическую библиотеку **`es.dll`** (экспертная система).

В качестве минимального необходимого содержимого библиотеки обозначим класс Esb (экспертная система бинарная), со следующим содержимым:

– поля, для хранения:

массива строк из текстового файла (это вся БД),

текущей позиции по строкам в БД,

позиции по уровню погружения в дерево,

символа отступа по уровню,

символа обозначения листка дерева

логической переменной (достигли листка дерева?).

– конструкторы (с параметрами и без);

– методы:

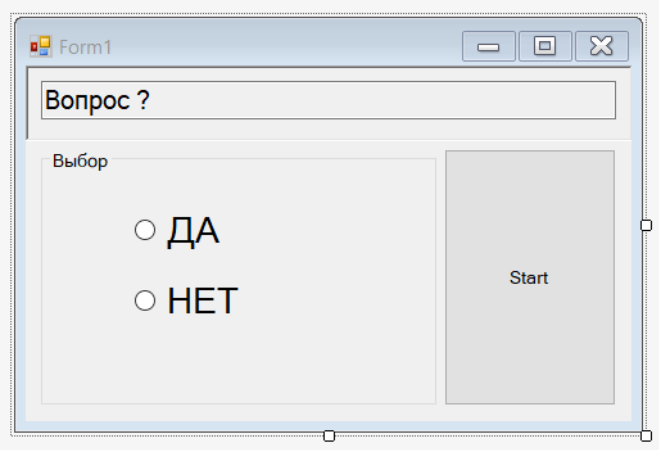
выдать вершину дерева (при старте программы),

выдать следующий узел, в ответ на выбор пользователя (да/нет).

```
    public class Esb
    {
// поля
        string[] lines; // массив строк - для хранения всей БД
        int posLines; // позиция по строкам
        int ur; // позиция по уровню погружения
        char sTab; // символ отступа по уровню
        char sEnd; // символ обозначения листка дерева
        public bool finish = false; // достигли листка дерева?
// конструкторы
        public Esb(string FileName) : this(FileName, '\t', '_')
        { }
        public Esb(string FileName, char stab, char send)
        {
            lines = File.ReadAllLines(FileName, Encoding.Default);
            sTab = stab;
            sEnd = send;
        }
// методы
        public string getStart()
        {
            posLines = 0; ur = 0;
            return lines[posLines] + " ?";
        }
        void findPos()
        {
            int countTab; // кол-во отступов в текущей строке
            do
            {
                countTab = lines[++posLines].LastIndexOf(sTab);
            } while (countTab != ur); // пока не найдем нужный уровень
        }
        public string getNext(int otv)
        {
            string result = "";
```

```
            posLines++;
            if (otv > 0) { findPos(); } // НЕТ
            result = lines[posLines].Substring(++ur);
            if (result[0] == sEnd)
            {
                finish = true;
                return result.Substring(1) + " .";
            }
            else
            {
                return result + " ?";
            }
        }
    }
```

В качестве интерфейса пользователя будем использовать приложение типа Wndows Forms с минимально достаточным количеством элементов управления:




```
public partial class Form1 : Form
{
    Esb db; // декларируем переменную для объекта БД ЭС

    public Form1()
    {
        InitializeComponent();
        string FileName = Directory.GetCurrentDirectory() + "/db.txt";
        db = new Esb(FileName); // создаём объект
    }

    private void button1_Click(object sender, EventArgs e)
    {
        if (button1.Text == "Start")
        {
            groupBox1.Enabled = true;
            button1.Text = "NEXT";
            textBox1.Text = db.getStart();
        }
        else
        {
            int otv; // выбор пользователя - ДА/НЕТ
            otv = (radioButton1.Checked == true) ? 0 : 1;
            textBox1.Text = db.getNext(otv);
            if (db.finish)
            {
                groupBox1.Enabled = false;
                button1.Text = "Start";
            }
        }
        radioButton1.Checked = false;
        radioButton2.Checked = false;
        button1.Focus(); // textBox1.TabStop = false;
    }
}
```

Возможные доработки:

– добавить класс для поддержки мультидерева;

– добавить в интерфейс пользователя функционал по добавлению ветвей и узлов в дерево поиска.