Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

Факультет «Информатика и системы управления»
Кафедра «Компьютерные системы и сети»

Е.К. Пугачев

Методические указания по выполнению лабораторных работ

по дисциплине «Интеллектуальные технологии и системы»

Часть 1

Исследование инструментальных средств и технологий разработки интеллектуальных систем

Москва 2019

Оглавление

Введение ... 3
Логическое программирование на языке Турбо- Prolog ... 3
Теоретические сведения ... 3
Основные понятия языка Turbo-Prolog ... 4
Структура программы и типы доменов языка Prolog ... 5
Предикаты и утверждения разных арностей ... 8
Переменные в языке Prolog ... 8
Использование правил ... 8
Организация циклов ... 9
Использование динамических баз данных ... 12
Использование списков ... 14
Преобразование данных в языке Prolog ... 15
Логические возможности языка Prolog ... 15
Реализация вспомогательных функций на языке Prolog ... 16
Краткий список встроенных предикатов ... 18
Порядок выполнения работы (Prolog) ... 21
Требования к отчету (Prolog) ... 21
Контрольные вопросы (Prolog) ... 22
Задания к 1-й части лабораторных работ (Prolog) ... 22
Смешанное программирование на языкеVisual Prolog ... 24
Особенности версии Visual Prolog 7 ... 24
Создание консольного приложения. ... 25
Создание базы фактов в отдельном файле ... 27
Режимы детерминизма предикатов ... 32
Факт-переменная ... 33
Организация циклов ... 34
Предикаты базы фактов ... 35
Работа с файлами ... 36
Рекурсия ... 37
Работа с атрибутами текста ... 38
Создание модулей ... 39
Списки ... 41
Анонимные предикаты ... 44
Порядок выполнения работы(Visual Prolog) ... 47
Требования к отчету(Visual Prolog) ... 47
Контрольные вопросы (Visual Prolog) ... 47
Задания к 1-й части лабораторных работ (Visual Prolog) ... 48

Введение

Целью проведения лабораторных работ является приобретение навыков проектирования и реализации основных элементов систем искусственного интеллекта.

Методические указания состоят из двух частей. Первая часть посвящена инструментальному средству создания интеллектуальных систем на основе продукционной модели, а вторая часть освоению и реализации основных методов, применяемых при создании экспертных систем.

Основы программирования предлагается изучать на основе использования языка Turbo-Prolog (Borland). Обусловлено это тем, что базовые концепции данного языка легли в основу других версий, например: PDC Prolog и Visual Prolog. Конкретно это касается синтаксиса, встроенных предикатов и механизмов работы. С другой стороны, Turbo-Prolog сам по себе предоставляет большие возможности создания различных обрабатывающих компонент интеллектуальных систем, например, обработка фраз естественного языка.

Приобретение навыков программирования с использованием современной технологии предполагается на основе современной версии языка *Visual* Prolog, Она предоставляет возможность сочетать логическое, функциональное и объектно-ориентированное *программирование*. В настоящее время язык *Visual* Prolog используется для создания систем управления аэропортов, обработки текстов на естественном языке, экспертных систем и др.

Логическое программирование на языке Турбо- Prolog

Теоретические сведения

Prolog (**Pro**gramming in **Log**ic) был разработан и впервые реализован в 1973 г. *Алэном Колмероэ* и другими членами "группы искусственного интеллекта" Марсельского университета. Главной задачей группы было создание системы *для обработки естественного языка*. Версий языка Prolog существует много, но неофициальным стандартом стал Шотландский вариант C&M Prolog (авторы William **C**locksin Christopher **M**ellish**)**. Отличие языка Turbo-Prolog от других версий в том, что в нем используется строгая типизация данных, и он построен на основе компиляции. Сделанные отступления позволили значительно увеличить скорость трансляции и счета программ.

Такие языки как *Pascal* и *C* относятся к разряду императивных. В программах, написанных на данных языках, используется линейное следование.

Turbo-Prolog является *декларативным языком*. Программа на декларативном языке представляет собой набор логических взаимосвязанных описаний, определяющих цель, ради которой она написана. Обозначения, используемые в языке Turbo-Prolog для выражений логических взаимосвязей, унаследованы из логики предикатов. Программист освобождается

от составления программы в виде последовательности действий. Язык Turbo-Prolog базируется на естественных для человека логических принципах и в нем отсутствуют такие явные управляющие структуры, как DO WHILE, IF THEN и т.п.

Основные понятия языка Turbo-Prolog

Далее вместо Turbo- Prolog используется одно слово Prolog .

Рассмотрим основные определения языка Prolog.

Факт - это некоторое утверждение, определяющее отношение между объектами или описывающее свойства объекта. Общая форма записи факта имеет следующий вид:

<имя отношения>(*имя_объекта_1, имя_объекта_2, ... , имя_объекта_N*).

Необходимо соблюдать следующие правила:

- имена всех отношений и объектов должны начинаться со строчной буквы;
- сначала записывается имя отношения, затем через запятую записываются имена объектов , а весь список имен объектов заключается в круглые скобки;
- каждый факт должен заканчиваться точкой;
- имена объектов в скобках могут перечисляться произвольно, но по одному произвольному порядку.

Например, факт с двумя объектами может быть описан так:

likes(tom,computer).

На естественном языке вышеприведенный факт означает: "Тому нравится компьютер".

Атом - имя, число без знака или символ.

Аргумент - имя объекта в круглых скобках.

Объект - название отдельного элемента в конструкции.

Домен - диапазон и тип значения, определенные для базисного типа данных.

Предикат - утверждение о наличии связи между объектами посредством задания имени отношения перед круглыми скобками и доменов его аргументов.

Функтор - имя составного объекта (объявляются в разделе программы predicates).

База данных - представляет собой совокупность фактов (утверждений).

Унификация - процесс, выполняющий попытки сопоставить цель и утверждение. Он обычно включает поиск, сопоставление и означивание.

Правило - утверждение о связи некоторого факта с другими фактами.

Общая форма записи правила имеет вид:

<заголовок правила>:- <тело правила>.

Заголовок представляет собой предикат. Тело состоит из термов, которые могут быть связаны между собой "," или ";". (","- означает И,";"- означает ИЛИ).

Между телом и заголовком стоит символ ":-", который означает ЕСЛИ.

Например, правило, состоящее из двух термов может описано следующим образом:

likes(tom,kathy) :- likes(kathy,computer), likes(kathy,apples).

На естественном языке это означает : "Тому нравится Кэти, если Кэти нравится компьютер и яблоки."

Структура программы и типы доменов языка Prolog

Программа может состоять из следующих разделов:

1. CONSTANS - раздел описания констант. Имена констант должны быть описаны строчными буквами. В качестве констант могут быть целые или вещественные числа, символы, строки. Например, col=17, x=3.62, c='W', st="Выход" .

2. DOMAINS - данный раздел содержит определения доменов, которые описывают различные классы объектов используемых в программе (определение типов данных).

Например, имеется факт

likes(mary,apples).

Здесь mary и appless являются объектами предиката likes . Prolog требует указания типов объектов для каждого предиката программы.

Для указания типа объекта Prolog имеет шесть следующих типов данных (типов доменов):

* symbol (символические имена) - это последовательность букв, цифр и знаков подчеркивания, которая начинается со строчной буквы или заключена в кавычки, например: flower, pay_check,"Prolog" и т.п..

* string (строки) - любая последовательность символов, которая заключена в кавычки (не более 250). Например: "today","123","ПРИВЕТ".

Ограничение 250 накладывается, если какой-либо переменной присваивается конкретное значение непосредственно в тексте программы. Например, пусть переменным S1 и S2 конкретизированы значениями

S1="111...1", (250 символов)

S2="222...2" (250 символов),

Далее, если соединить эти строки , то ничего не потеряется

concat(S1,S2,S)

В итоге имеем переменную S, в которой 500 символов. Длина может достигать строки в таких случаях может достигать 64 Кбайт.

* char (символы) - отдельный символ, заключенный в апострофы, например: 'A','3','a','\13'.
* integer (целые числа) - можно задавать в диапазоне от -32768 до +32767.
* real (действительные числа) - диапазон от +1E-307 до +1E308 .
* file (файлы) - допустимое имя файла.

Вернемся к факту lakes(mary,apples). С предикатом likes могут быть еще факты, например:

likes(tom,computer).

likes(kathy,computer).

Во всех этих фактах с предикатом lakes на первом месте стоят имена объектов, имеющие смысл "тот, кто любит", а на втором месте имена объектов - "вещь".

В данном примере эти два вида имен объектов имеют один и тот же тип symbol. Прежде чем показать, как выглядит раздел DOMAINS для нашего примера, рассмотрим раздел, который называется PREDICATES.

3. PREDICATES - данный раздел служит для описания используемых программой предикатов.

В нашем примере предикат likes не является встроенным предикатом, поэтому его необходимо объявить в разделе PREDICATES.

Например:

```
PREDICATES
    lakes(symbol,symbol)
```

Это описание означает, что оба объекта относятся к типу symbol. Такое описание предиката likes освобождает от необходимости использования раздела DOMAINS. Наш пример является очень простым, поэтому мы можем себе такое позволить. Реальные программы могут содержать несколько десятков предикатов, причем с различным количеством объектов. Поэтому, чтобы программа хорошо читалась - видам объектов присваивают имена, но при этом уже необходим раздел DOMAINS. Для нашего примера это будет выглядеть так:

```
DOMAINS
        person,thing = symbol
PREDICATES
        likes(person,thing)
```

где person - "тот кто любит" ,thing - "вещь".

4. CLAUSES - в данный раздел заносятся факты и правила. О содержимом этого раздела можно говорить как о данных, необходимых для работы программы. Для нашего примера этот раздел включает три факта :

CLAUSES

likes(mary,apples). likes(tom,computer). likes(kathy,computer).

5. GOAL - данный раздел может располагаться перед разделом CLAUSES или после него. В этом разделе определяется цель. Цель может состоять из нескольких подцелей. Если программа предназначена для работы в пакетном режиме, раздел GOAL не может быть опущен.

В программе могут присутствовать еще два раздела, обеспечивающие определение глобальных доменов и предикатов.

6. GLOBAL DOMAINS - располагается после раздела DOMAINS.

7. GLOBAL PREDICATES - следует после раздела PREDICATES.

Определение типов данных и предикатов в этих разделах позволяет обеспечить межмодульный интерфейс.

8. DATABASE - следует перед разделом PREDICATES. Он содержит определения предикатов динамической базы данных. Если программа такой базы данных не требует, то этот раздел может быть опущен.

Если необходимо вставить в программу комментарии, то можно использовать два способа:

- для выделения произвольного блока комментариев используют символы ‘/’ и ‘*’ в двух сочетаниях: /* и */, т.е. помечаем начало и конец соответственно.
- для пометки строки или ее части в качестве комментария используется символ ‘%’.

В итоге программа в законченном виде для нашего примера будет выглядеть так :

Пример 1.

```
/* НАЧАЛО */
Domains
        person,thing = symbol
Predicates
        likes(person,thing)
Goal
        likes(mary,apples),nl, write("Мэри любит яблоки").
Clauses
          likes(mary,apples).
          likes(tom,computer).
          likes(kathy,computer).
% КОНЕЦ
```

Предикаты и утверждения разных арностей

Термин «арность» обозначает число объектов утверждения. Количество объектов в одном предикате может быть не более 50.

Ниже приведен пример предикатов и утверждений различных арностей.

```
Domains        s=string
Predicates     % раздел предикатов
        start          %  арность 0
        woman(s)       %  арность 1
        father(s,s)    %  арность 2
Clauses        % раздел утверждений
        start.         %  арность 0
        woman("Маша")       %  арность 1
        father("Петр Иванович","Маша")   %  арность 2
```

Переменные в языке Prolog

Любое имя может быть переменной, если начинается с прописной буквы. Переменная позволяет отвечать на вопросы. Например, если необходимо узнать, кому нравятся яблоки (см. пример 1), то в разделе GOAL должны написать следующее

likes(X,apples),nl, write(X," - любит яблоки").

Переменная X конкретизируется первым значением, который имеется в базе фактов с предикатом likes.

Использование правил

Правила используются, когда необходимо сказать, что некоторый факт зависит от группы других фактов.

При описании правил часто применяют переменные. Например, правило с одной переменной и одним предикатом может выглядеть так

likes(tom,X) :- likes(X,wine).

На естественном языке это означает: «Тому нравится любой, кому нравится вино».

Правило, в котором используется одна переменная и два предиката может быть описано следующим образом

likes(tom,X) :- woman(X),likes(X,wine).

На естественном языке звучит так: «Тому нравится любая женщина, которой нравится вино».

Правило, в котором используются две переменные и три предиката может выглядеть так

is_sister(X,Y) :- woman(X), parents(X,M,F), parents(Y,M,F).

X является сестрой Y, если X - женщина и имеет родителей M и F и Y имеет тех же родителей M и F.

Часто в правилах используются анонимные переменные. Анонимная переменная - это одиночный знак подчеркивания «_». Например, если нужно определить, является ли X вообще чьей-нибудь сестрой и неважно чьей, то после конкретизации X нужно записать:

... , is_sister(X,_), ...

Предыдущие примеры не учитывают факт существования нескольких вариантов, удовлетворяющих условию поиска, или просто возможность выдать содержимое всей базы.

Рассмотрим правило, позволяющее выдавать содержимое всей базы данных.

total:-woman(X),write(X),nl, fail.

В примере используется предикат fail , который осуществляет вынужденное неудачное завершение выполнения. Это один из способов организации циклов в Prologe (см. ниже), который позволяет просмотреть все факты с предикатом woman.

Ниже приведены два правила.

```
st:-consult("bd.pro"), ret.
ret:-retract(woman(_)), fail.
```

В теле первого правила с заголовком st используется два терма (предиката). С помощью встроенного предиката consult осуществляется загрузка базы данных с именем bd.pro, в которой содержатся факты с предикатом woman. Терм ret позволяет обратиться ко второму правилу с заголовком ret. Второе правило позволяет удалить все факты с предикатом woman из памяти.

Организация циклов

В языке Prolog имеется два основных способа организации циклов, при организации которых необходимо учитывать встроенные «невидимые» механизмы языка.

В первом способе используется рекурсия. Общий вид правила, выполняющего рекурсию, следующий:

```
repetitive_rule :- %правило рекурсии
                   <предикаты и правила>,
                   repetitive_rule.
```

Во втором способе используется встроенный предикат fail, который вызывает откат. Данный способ называют повторением.

Откат - это механизм, который Prolog использует для нахождения дополнительных фактов и правил, необходимых при вычислении цели, если текущая попытка вычислить цель оказалась неудачной.

Общий вид правила, выполняющего повторение с помощью встроенного механизма, следующий

```
repetitive_rule :-  %правило повторения
                    <предикаты и правила>,
                    fail.
```

Ниже приведен пример 2а, в котором для вычисления факториала используется рекурсия.

Пример 2а.

```
Domains r=real
Predicates
    start  calculate(r,r,r)
Goal start.
Clauses
    start:- write("Введите положительное целое число: "),
            readint(X),calculate(X,1,1).
    calculate(X,Y,Z) :- X>Y,Y1=Y+1,Z1=Z*Y1,calculate(X,Y1,Z1).
    calculate(X,Y,Z) :- X=Y,write(X,"!=",Z).
    calculate(X,Y,Z) :- X<Y,write(X,"!=1").
```

В данном примере используется символ «=». Данный символ применяется в двух случаях:

* чтобы конкретизировать переменную (присвоить конкретное значение);
* чтобы сравнить два значения.

Внешне эти два случая могут ничем не отличаться. Чтобы правильно понять смысл символа «=» в каком-то конкретном случае, необходимо проследить предыдущую цепь событий. В примере 2а в первом правиле с предикатом calculate вновь введенным переменным Y1 и Z1 присваиваются конкретные значения. Во втором правиле X и Y сравниваются, т.к. их значения передаются из заголовка правила (т.е. значения X и Y уже конкретизированы).

В примере 2в демонстрируется организация цикла без рекурсии с использованием предиката fail. Данный пример позволяет с помощью стандартных средств языка выбирать из каталога файлы и просматривать их содержимое.

Пример 2в.

```
Database    namefile(string) % для сохранения имени выбранного файла
Predicates  start andisk
```

```
Goal  start.
Clauses
start:-  makewindow(1,112,94," РЕЖИМ ПРОСМОТРА ТЕКСТОВЫХ ФАЙЛОВ
        (Esc-выход ...) ",0,0,25,80),    andisk.
andisk:- makewindow(2,45,47,"",1,1,4,78),dir("C:\\","*.*",Name),
        assert(namefile(Name)),removewindow,fail.
andisk:- not(namefile(_)),removewindow,exit,!.
andisk:- namefile(Name),concat("Содержимое файла ",Name,S),
        makewindow(3,73,27,S,1,1,23,78), file_str(Name,Str),
        display(Str),retractall(namefile(_)),removewindow,fail.
andisk:- andisk.
```

Ниже приведен пример 3, в котором с помощью предиката fail на экран выдается содержимое всей базы фактов с предикатом woman.

Пример 3.

```
Domains
        s=string
Predicates
        start
        woman(s)
Goal start.
Clauses
        start:- woman(Name),write(Name),nl,fail.
        % база фактов
        woman("Катя").  woman("Таня").  woman("Маша"). woman("Ира")
```

Кроме механизма отката в языке Prolog существует механизм отсечения, который можно организовать с помощью предиката cut (отсечение). В тексте программы предикат cut обозначается символом « ! ». Данный предикат устанавливает барьер, запрещающий выполнить откат ко всем альтернативным решениям текущей проблемы. Однако последующие подцели могут создать новые указатели отката и тем самым создать условия для поиска новых решений.

В примере 4 демонстрируется механизм отката. В отличии от примера 3, где на экран выдаются все факты, в данном примере, как только будет найдена женщина с именем Маша и выведено на экран, дальнейший поиск прекращается (т.е. происходит отсечение).

Пример 4.

```
Domains         s=string
Predicates      start
                woman(s)
                make_cut(s)
Goal            start.
Clauses
        start:- woman(Name),write(Name),nl,make_cut(Name),!,fail.
        make_cut(Name):-Name="Маша".
                % база фактов
woman("Катя"). woman("Таня"). woman("Маша"). woman("Ира").
```

Использование динамических баз данных

Программы баз данных на Prologe есть частный случай систем управления базами данных. Можно выделить три основные модели организации базы данных это: иерархическая модель (данные хранятся в иерархии классов), сетевая модель (данные в виде связанных агрегатов, образующих сеть) и реляционная модель (данные в виде таблиц). Prolog ориентирован на создание баз данных на основе реляционной модели. Для работы с базами данных существует достаточно большой набор встроенных предикатов. Некоторые из них продемонстрированы в примерах 5 и 6.

В Примере 5 происходит загрузка файла базы данных с именем «facts.bd», далее в случае существования объекта с именем Name все факты связанные с этим объектом уничтожаются, после чего база данных на диске обновляется.

Пример 5.

```
Domains
  i = integer  s = string
Database
        age(s,i)
        woman(s)
        man(s)
Predicates
        start
        conclusion(s)
Goal   start.
Clauses
start:- existfile("facts.bd"),consult("facts.bd"), % если база существует, то загружаем
        makewindow(1,27,57,"",0,0,25,80),  % создаем окно на весь экран
        /* где 1 - номер окна , 27 - атрибут экрана ( цвет символов )
        57 - атрибут рамки ,заголовок окна,
        0,0 -координаты верхнего левого угла окна (Y,X),
        25,80 -размеры окна по осям Y и X. */
write("Введите имя: "),readln(Name),nl, % вводим искомый объект
        conclusion(Name).   % переходим к правилу с одноименным заголовком
start.
conclusion(Name):-
        woman(Name),          % проверка существования объекта с именем Name
        retract(age(Name,_)), retract(woman(Name)),          % удаляем данные по объекту
        save("facts.bd"),         % сохраняем базу на диске
        write("Данные по объекту ",Name," удалены из базы !"),
        readchar(_).  % задержка экрана до нажатия любого символа
conclusion(Name):-
        not(woman(Name)),   %  если объект не существует, то выводим сообщение
        write("Объект ",Name," отсутствует в базе !"),readchar(_).
        /* содержимое файла "facts.bd"
age("Петя",20)   age("Таня",10)   age("Катя",18)
woman("Таня") woman("Катя")
man("Петя")
*/
```

Prolog позволяет создать несколько баз данных и дает возможность работать с ними как по отдельности, так и одновременно со всеми.

В примере 6 приведен фрагмент программы, который позволяет сохранить в базе данных на диске текущую дату с подтверждением пользователя. В данном примере используется база данных с именем da1. Обобщенный алгоритм программы такой:

* загружается файл базы данных с именем da1;
* если факты в базе отсутствуют, то в память заносится факт da("",1);
* с помощью системных предикатов считываем текущую дату;
* используя встроенный редактор Prologa пользователь подтверждает (вводит) текущую дату;
* если формат даты правильный, то база данных с именем da1 обновляется.

Пример 6.

```
Domains         i = integer  s = string
Database - da1    % описание динамической базы данных с именем da1
        da(s,i)             % 1-й объект дата,
        % 2-й объект - номер пользователя (активно не используется)
Predicates
        start   % предикат целевого утверждения
        dtt     % для ввода новой даты
        td(i,s) % для проверки правильности ввода с сохранением
Goal    start.
Clauses
start:- existfile("da.bd"),consult("da.bd",da1), % загрузка базы в память
        makewindow(1,0,0,"",0,0,25,80), dtt, removewindow, !.
start.
dtt:-not(da(_,_)),assert(da("",1)).       % если факты отсутствуют в базе, то
        % добавляем в память начальный факт
dtt:-   da(H1,_),       % считываем последнюю дату работы из базы
          % определяем текущую дату и выводим ее в заголовке окна редактора
        date(G,M,D),str_int(G1,G),str_int(M1,M),str_int(D1,D),
        concat("Введите дату (Esc - отказ) [СЕГОДНЯ:",D1,S1),
        concat(S1,".",S2),concat(S2,M1,S3),concat(S3,".",S4),
        concat(S4,G1,S5),concat(S5," г.]",S6),
        makewindow(5,48,91,S6,10,10,4,58),
          % используем редактор Prologa
        editmsg(H1,H2,"F10 - выход с сохранением","","",0,"",C),
        removewindow,td(C,H2),!.
dtt:-dtt. % используем рекурсию для повторного ввода в случае ошибки
                % если дата введена правильно, то обновляем базу данных
td(C,H2):- C=0,str_len(H2,Ld),Ld<9,
        frontstr(2,H2,H3,H4),str_int(H3,Nd),Nd>0,Nd<32,
        frontstr(1,H4,K1,K2),K1=".", frontstr(2,K2,P1,P2),
        str_int(P1,Nm),Nm<13,Nm>0, frontstr(1,P2,J1,J2),J1=".",
        frontstr(2,J2,Q1,Q2),str_int(Q1,Ng),Ng>=0,
        da(_,Bb),  retractall(da(_,_)),assert(da(H2,Bb)),
        save("da.bd",da1),!.
td(C,H2):- C=1,!. % если пользователь ничего не ввел, то база не обновляется.
```

Использование списков

Список является набором связанных объектов одного и того же доменного типа. Объектами списка могут быть целые числа, действительные числа, символы, символьные строки и структуры. Prolog позволяет выполнять со списком целый ряд операций. Их перечень включает:

* доступ к объектам списка;
* проверка на принадлежность к списку;
* разделение списка на два;
* слияние двух списков;
* сортировку элементов списка.

Основным встроенным механизмом Prologa для работы со списком является метод «разделения списка на голову и хвост». Ниже приведены примеры 7.1 и 7.2 . В одном из них показано, как можно разложить список на элементы, а в другом создать список из множества фактов и сохранить его в базе данных.

Пример 7.1. Программа демонстрации разделения списка на голову и хвост.

```
Domains         i=integer
        l=integer* %  список целых чисел
Predicates      split(l)
        sw
Goal            makewindow(1,1,7,"",0,0,25,80),sw.
Clauses
        sw:- S=[1,2,3,4,5],     % соэдаем список
        split(S).         % обращаемся к правилу разделения списка
                % разделение списка с помощью рекурсивного цикла
        split([N|S]):-write(N," ",S),nl,readchar(_),fail.
        split([]):-write("Список пуст").
        split([N|W]):-split(W).         % основная идея разделения списка
```

Пример 7.2. Программа собирает в список все объекты фактов с предикатом bn

```
Domains         i=integer
                l=integer*      % список целых чисел
Database        d(l) % база данных, где объектом в предиката является список
Predicates      sw      bn(i)   fb      sp(l)
Goal            makewindow(1,1,7,"",0,0,25,80),sw.
Clauses
        sw:-    assert(d([])),    % создаем в памяти факт БД с объектом типа "список"
        fb.             % переходим к правилу формирования списка
                % правила формирования списка с помощью бектрекинга
        fb:-bn(N),d(S),sp([N|S]),fail.
        sp(S):-retractall(d(_)),assert(d(S)),
        write(S," "),readchar(_).
                % база фактов непосредственно в тексте программы
        bn(1). bn(2).
        bn(3). bn(4).
```

Преобразование данных в языке Prolog

Преобразование данных необходимо, если тип объектов встроенного предиката отличается от типа объектов предиката, определенного пользователем. Все предикаты преобразования данных содержат два объекта. Имена предикатов показывают тип выполняемого преобразования , а так же указывают и порядок следования объектов.

Особенностью является то, что предикаты преобразования имеют два направления. Основные виды преобразований представлены в примере 8.

Пример 8.

```
Predicates
                start  conv(char,string,integer,real)
Goal start.
Clauses
        start:-C='A',S="A",I=65,R=3.14,nl,conv(C,S,I,R).
        conv(C,S,I,R):-char_int(C,X),nl,write(X),fail.          %  X=65
        conv(C,S,I,R):-char_int(C1,I),nl,write(C1),fail.        %  C1=A
        conv(C,S,I,R):-str_int(S1,I),nl,write(S1),fail.%  S1=65
        conv(C,S,I,R):-str_char(S,C1),char_int(C1,I1),nl,write(I1),fail.   %  I1=65
conv(C,S,I,R):-str_real(S1,R),nl,write(S1),fail.         S1=3.14
```

Кроме встроенных предикатов преобразования данных пользователь может ввести свои. В примере 9 приведено нестандартное преобразование цифр в соответствующие им слова.

Пример 9.

```
Predicates
        start(string)  conv(string,integer)
Goal
        start("0123456789").
Clauses
start(S):- frontstr(1,S,S1,S2),  % расщепляем строку на элементы
           str_int(S1,K),conv(X,K),write(X),nl,start(S2).
        % База фактов преобразования
conv("ноль",0).       conv("один",1).       conv("два",2).        conv("три",3).
conv("четыре",4).     conv("пять",5).       conv("шесть",6).      conv("семь",7).
conv("восемь",8).     conv("девять",9).
```

Логические возможности языка Prolog

С помощью правил можно описать какой-либо процесс принятия решения или логический вывод. В этом случае в программе могут использоваться не только встроенные механизмы вывода или вывод, смысл которого сводится просто к поиску объекта в базе фактов, но и собственные пользовательские правила принятия решения.

Рассмотрим пример программы (см. пример 10), в которой определяется родственная связь на основе неявно заданных фактов, т.е. в базе отсутствуют факты типа «Объект_1

является_сестрой Объекта_2». В соответствии с примером, если ввести имя «Таня», то результатом вывода будет фраза: « Имеется брат Борис».

Пример 10. Определение родственной связи

```
Domains         s=string

Predicates
        start w(s) p(s,s,s) per(s) is_rs(s,s) m(s) wiw(s)
Goal    start.
Clauses
        start:- makewindow(1,52,37,"Определение родственной связи",0,0,25,80),
        write("  Введите имя : "),readln(X),nl,per(X).
        per(X):- is_rs(X,Y),X<>Y,wiw(Y),fail. %  основное правило проверки
        per(X):- nl,nl,write("Конец поиска."), readchar(_).
        % правило проверки общих родителей
        is_rs(X,Y):- p(X,M,F),p(Y,M,F).
        % уточнение и вывод родственной связи
        wiw(Y):-w(Y),write("  Имеется сестра ",Y),nl,!.
        wiw(Y):-m(Y),write("  Имеется брат ",Y,"."),nl,!.
  % база фактов женщин
w("Катя"). w("Света"). w("Таня").
  % база фактов мужчин
m("Дима").   m("Борис").   m("Вася").
  % база фактов родителей
p("Катя","Лена","Сергей").  p("Света","Галина","Сергей").
p("Таня","Зоя","Костя").                p("Дима","Лена","Сергей").
p("Вася","Лена","Сергей").  p("Борис","Зоя","Костя").
```

Реализация вспомогательных функций на языке Prolog

В языке Prolog имеется много встроенных предикатов для реализации вспомогательных функций, которые облегчают создание больших систем. Перечислим некоторые из них:

* вывод и просмотр файла (строки) со скроллингом;
* редактирование файла (строки);
* вывод текущего каталога;
* реализация многооконного интерфейса и т.д.

Однако, это не все функции, которые требуется реализовать при создании больших систем.

В примере 11 приведена программа реализации меню. Пункты меню можно выбирать с помощью клавиш стрелок или с помощью мышки.

Пример 11.

```
Constants
    cz0=63  % начальный цвет  пунктов меню
    col=28  % цвет пометки пунктов меню
    c1=63   % цвет фона главного окна
    c2=118   % цвет рамки
```

```
    kp=5     % количество пунктов в меню
Domains        i = integer  s = string  r=real
Database       ar(i,i) mn(i,i,i)
Predicates
        mkw(i,i,i,s,i,i,i,i)  rmw
        r_k(i)  c_k(i,i)   ko(i,s,i,i,i)  ord(i,i,i)
        bor  an(i,i)  start  menu(i,i)  ed_ar(i,i) uz(i,i,s)
        codm(i,i) ed_mn(i,i,i) my(i,i) pm(i,i) zn(i) initm
Goal    initm,% инициализация мыши
        start.% запуск меню
Clauses
start:- mkw(1,31,52," ОТДЕЛ ГРАФИЧЕСКИХ СИСТЕМ ",0,0,25,80),
        Xnt=22,mkw(2,0,0,"",8,Xnt,9,33),
        Xn=Xnt-2,mkw(3,c1,c2," Г Л А В Н О Е    М Е Н Ю ",7,Xn,9,33),
        menu(1,kp),pm(1,1),zn(Xn), ed_ar(80,1),bor,!.
 % Вывод пунктов меню с помощью рекурсивного правила
menu(X,Y):- ko(X,A,Y1,X1,C),cursor(Y1,X1),write(A),
        str_len(A,L),field_attr(Y1,X1,L,C),X2=X+1,X<Y,menu(X2,Y),!.
menu(X,Y):-X=Y,!.
 % создание окна
mkw(A1,A2,A3,S,B1,B2,B3,B4):-makewindow(A1,A2,A3,S,B1,B2,B3,B4,1,255,"г¬L-=¦").
rmw:-removewindow,!.  % удаление окна
% главный цикл опроса
bor:- keypressed,ar(_,R),r_k(K), an(K,R),fail.
bor:- codm(K,R),an(K,R),fail,!.
bor:- bor.
/*   основное правило    в  меню  */
an(Kod,R):- ko(R,A,Y,X,C),ord(R,Kod,Rs), pm(1,Rs),pm(0,R),ed_ar(Kod,Rs),!.
an(Kod,R):-uz(R,Kod,S),existfile(S),!.
an(Kod,R):-uz(R,Kod,S),mkw(26,112,67,"",4,10,3,52),cursor(0,5),
        write("  ФАЙЛ ",S," отсутствует !  "),readchar(_),rmw,!.
an(Kod,R):-R=5,Kod=13,mkw(10,7,0,"",0,0,25,80), exit,!.
an(Kod,R):-!.
% факты срабатывания пункта меню
% номер пункта, код срабатывания, имя файла
        uz(1,13,"form.exe").uz(2,13,"registr.exe").
        uz(3,13,"report.exe").uz(4,13,"servis.exe").
% модификация служебной базы
ed_ar(A,B):-retractall(ar(_,_)),assert(ar(A,B)),!.
% определение следующего пункта меню
ord(R,Kod,Rs):-R<kp,Kod=80,Rs=R+1.
ord(R,Kod,Rs):-R>1,Kod=72,Rs=R-1.
% факты пунктов меню
% ko(номер пункта,имя пункта,координата по Y, координата по X, цвет)
ko(1,"1. Формирование заказа",1,2,cz0).
ko(2,"2. Регистрация клиента",2,2,cz0).
ko(3,"3. Формирование отчета",3,2,cz0).
ko(4,"4. Сервисные функции",4,2,cz0).
ko(5,"5. Выход",5,2,cz0).
%Чтение (расширенного) кода
 r_k(Kod):- readchar(C), char_int(C,A),c_k(A,Kod),!.
c_k(A,Kod):- A<>0,Kod=A,!.
c_k(0,Kod):- readchar(C),char_int(C,Kod),!.
% пометка или разметка пунктов меню
pm(1,R):-ko(R,A,Y,X,_),str_len(A,L),field_attr(Y,X,L,col),!.
```

```
pm(0,R):-ko(R,A,Y,X,C),str_len(A,L),field_attr(Y,X,L,C),!.
pm(N,R).
 % Правила для мышки
initm:-  bios($33, reg(0,0,0,0,0,0,0,0),reg(_,_,_,_,_,_,_,_)),
         bios($33, reg(1,0,0,0,0,0,0,0), reg(_,_,_,_,_,_,_,_)),!.
initm.
codm(K,R):- bios($33, reg(3,0,0,0,0,0,0,0),reg(_,M,X,Y,_,_,_,_)),M=1,
         my(Y,R),mn(R,Xn,Xk),X>=Xn,X<=Xk,       ar(_,Rr),pm(0,Rr),
         pm(1,R),ed_ar(0,R),K=13,!.
my(72,1).my(80,2).my(88,3).my(96,4).my(104,5).
zn(O):- ko(R,A,Y,X,_),str_len(A,L),Xn=(O+X+1)*8,Xk=Xn+L*8-1,ed_mn(R,Xn,Xk),fail.
zn(O).
ed_mn(R,Xn,Xk):-retractall(mn(R,_,_)),assert(mn(R,Xn,Xk)),!.
```

Краткий список встроенных предикатов

Полный список приведен в файле prolog.hlp.

Предикаты ввода.

readln(StringVariable) - читает строку.

readint(IntgVariable) - читает целое число.

readreal(RealVariable) - читает действительное числощ.

eadchar(CharVariable) - читает символ.

file_str(DosFileName,StringVariable) - читает строку из файла.

inkey(CharVariable) - читает ключ (символ).

keypressed - проверяет, нажата ли клавиша.

Предикаты вывода.

write(Variable|Constant *) - производит запись на текущее устройство вывода.

nl - перевод строки.

Предикаты для работы с файлами.

openread(SymbolicFileName,DosFileName) - открывает файл для чтения.

openwrite(SymbolicFileName,DosFileName) - открывает файл для записи.

openappend(SymbolicFileName,DosFileName) - открывает файл для добавления.

openmodify(SymbolicFileName,DosFileName) - открывает файл для чтения/записи.

readdevice(SymbolicFileName) - определяет или считывает символическое имя файла устройства ввода.

writedevice(SymbolicFileName) - определяет или считывает символическое имя файла устройства вывода.

filemode(SymbolicFileName,FileMode) - установка или чтение типа файла.

closefile(SymbolicFileName) - закрывает файл.

filepos(SymbolicFileName,FilePosition,Mode) - установка или чтение позиции указателя.

eof(SymbolicFileName) - проверка на конец файла

flush(SymbolicFileName) - очищает содержимое буфера.

existfile(DosFileName) - проверка существования файла.

deletefile(DosFileName) - удаляет файл.

renamefile(OldDosFileName,NewDosFileName) - переименовывает файл.

disk(DosPath) - устанавливает или показывает накопитель или путь.

Предикаты экрана.

scr_attr(Row,Column,Attr) - устанавливает или считывает атрибут.

field_str(Row,Column,Length,String) - записывает или читает строку.

field_attr(Row,Column,Length,Attr) - устанавливает или читает атрибут поля экрана.

cursor(Row,Column) - считывает или устанавливает позицию курсора.

cursorform(Startline,Endline) 0<Startline<14, 0<Endline<14

- считывает или устанавливает форму курсора.

attribute(Attr) - считывает или устанавливает цвет фона текущего окна.

Предикаты работы с окнами.

makewindow(WindowNo,ScrAtt,FrameAtt,Framestr,Row,Column,Height,Width)

- - создает окно.

shiftwindow(WindowNo) - меняет текущее окно или считывает номер текущего окна.

gotowindow(WindowNo)- -активизирует окно с заданным номером.

existwindow(WindowNo) - проверяет существование окна.

removewindow - удаляет текущее окно.

clearwindow - чистка окна.

window_str(ScreenString) - записывает (или считывает) строку в текущее окно

window_attr(Attribute) - определяет атрибуты текущего окна

scroll(NoOfRows,NoOfCols) - сдвиг содержимого текущего окна.

Предикаты для работы со строками.

frontchar(String,FrontChar,RestString) - разделяет заданную строку на первый символ и оставшуюся часть.

fronttoken(String,Token,RestString) - разделяет строку на лексему и остаток.

frontstr(Lenght,Inpstring,StartString,RestString) - разделяет строку на две части, количество первой части равно Lenght.

concat(String1,String2,String3) - String3 = String1 + String2.

str_len(String,Length) - определяет длину строки.

Предикаты преобразования данных.

char_int(CharParam,IntgParam) - преобразует символ в целое число или наоборот.

str_int(StringParam,IntgParam) - преобразует строку в целое число или наоборот.

str_char(StringParam,CharParam) - преобразует строку в символ или наоборот.

str_real(StringParam,RealParam) - преобразует строку в действительное число или наоборот.

upper_lower(StringInUpperCase,StringInLowerCase) - преобразует прописные буквы в строчные и наоборот.

Предикаты баз данных.

consult(DosFileName) - загружает или добавляет текстовый файл базы данных в ОП.

consult(DosFileName,InternalDatabaseName) - загружает в ОП группу поименованную группу фактов.

save(DosFileName) - записывает на диск все факты динамической БД.

save(DosFileName,InternalDatabaseName) - записывает на диск поименованную группу фактов БД.

assert(Term) - добавляет факт БД в ОП.

retractall(Term) - удаляет все факты с указанным термом.

retractall(_, InternalDbaseName) - удаляет все факты поименованной группы.

Предикаты для работы с редактором.

display(String) -) - показывает в текущем окне строку(до 64Кбайт).

edit(InputString,OutputString) - вызов редактора.

editmsg(InputString,OutputString,Headstr,Headstr2,Msg,Pos,Helpfilename,RetStatus) - вызов редактора с дополнительными возможностями.

Системные предикаты.

system(DosCommandString) - выполнение команд DOS.

dir(Path,Filespec,Filename) - выводит текущий каталог.

comline(LineBuffer) - читает параметры командной строки.

port_byte(PortNo,Value) - посылает байт в порт или читает его из порта.

ptr_dword(8086Ptr,Segment,Offset) - читает строку или адрес строки.

memword(Segment,Offset,Word) - запоминаетр или считывает слово по заданному адресу.

membyte(Segment,Offset,Byte) - запоминает или считывает байт.

bios(Interruptno,reg(AXi,BXi,CXi,DXi,SIi,DIi,DSi,ESi), reg(AXo,BXo,CXo,DXo,SIo,DIo,DSo,ESo)) - объявляет прерывания для вызова процедур BIOS/

exit - выход из программы.

storage(StackSize,HeapSize,TrailSize) - определяет размер имеющейся памяти.

sound(Duration,Frequency) - звуковой сигнал с параметрами.

beep - звуковой сигнал.

date(Year,Month,Day) - установка или считывание даты.

time(Hours,Minutes,Seconds,Hundredths) - устанавливает или считывает

системное время.

findall(Variable, Atom, ListVariable) - собирает значения, возникающие в процессе бектрегинга, в список.

free(Variable) - проверяет свободная ли переменная.

bound(Variable) - проверяет связана ли переменная.

Арифметические операции.

+, -, *, /, mod, div

Операции отношения.

>, <, =, >=, <=, <>, ><

Логические операции.

not(Atom) - отрицание.

and , or

Функции.

sin, cos, tan, arctan, ln, log, exp, sqrt, round, trunc, abs

Порядок выполнения работы (Prolog)

1. Ознакомиться с основными понятиями и механизмами языка Prolog.
2. Изучить работу основных встроенных предикатов и механизмов языка Prolog на конкретных примерах из методических указаний.
3. Сформулировать основные отличительные особенности языка Prolog относительно языка процедурного типа (Pascal, CI++ и т.п.).
4. Выполнить задания в соответствии с вариантом, полученным у преподавателя.

Требования к отчету (Prolog)

Отчет по каждому заданию должен включать:

- Номер варианта и задание;
- Структурную карту Константайна (схему вызовов микромодулей);
- Текст отлаженной программы с комментариями;
- Фрагмент базы данных (если данные в отдельном файле);
- Результаты работы программы;
- Заключение (перечисляются, какие основные механизмы языка Prolog были использованы).

Порядок проведения защиты:

- Ответить на вопросы, предложенные преподавателем по теоретическому материалу.
- рассказать принцип работы программы, выполненной в соответствии с вариантом задания программы.

Контрольные вопросы (Prolog)

1. Какие разделы имеются в языке Prolog (Borland) ?
2. Сколько минимум может быть разделов в языке Prolog для работы программы в пакетном режиме?
3. Приведите конструкцию (шаблон), который реализует счетный оператор цикла?
4. Приведите конструкцию (шаблон), который реализует цикл с предпроверкой условия?
5. Приведите конструкцию (шаблон), который реализует цикл с постпроверкой условия?
6. Приведите конструкцию (шаблон), который реализует цикл, в котором тело цикла повторяется столько раз, сколько фактов в базе данных?
7. Сформулируйте правила, по которым работает основной механизм вывода?
8. Что значит обратный порядок вывода?
9. Какие виды переменных имеются в языке Prolog?
10. Как долго хранится значение переменной?
11. Приведите примеры конструкций фактов и правил?
12. Как работает программа, приведенная в примере №X?

Задания к 1-й части лабораторных работ (Prolog)

Общие требования:

1. Кроме часто используемых служебных данных небольшого объема, все остальные данные должны подгружаться с диска.
2. Обеспечить своевременную чистку стека, т.е. программа не должна прерываться при большом количестве повторений.
3. Обеспечить приемлемую универсальность программы, т.е. в правилах необходимо использовать переменные, постоянные составляющие определять как факты и т.п.

Таблица 1 - Варианты заданий.

Вариант	**Задание** **(реализовать на языке Prolog)**
Вариант 1	Дана строка (до 64 Кбайт) в текстовом файле. Создать динамическую базу предложений данной строки. По номеру предложения выдавать его на экран. Для создания интерфейса использовать стандартные средства.
Вариант 2	Дан текстовый файл. Создать новый файл, преобразовав старый файл путем перевода всех букв (латинского и русского алфавитов) в строчные или прописные по желанию пользователя. Обеспечить просмотр файлов с помощью стандартных средств.
Вариант 3	Дан текстовый файл. Создать новый файл путем удаления лишних пробелов между словами. Обеспечить просмотр файлов с помощью стандартных средств.
Вариант 4	Дана строка (до 64 Кбайт). Создать базу фактов слов данной строки. По номеру слова выдавать его на экран. Использовать стандартные средства для разработки интерфейса.
Вариант 5	Дан текстовый файл. Создать список слов данного файла и сохранить его как элемент динамической базы данных. Реализовать функцию проверки на вхождение какого-либо слова в сформированный список.

Вариант 6	Дана база фактов синонимов и антонимов слов на естественном языке. Реализовать функцию проверки на существование антонима или синонима слова, которое вводится с клавиатуры, а также функцию просмотра фактов в отсортированном виде.
Вариант 7	Создать базу данных записной книжки. Поля: ФИО, ТЕЛЕФОН, ДАТА РОЖДЕНИЯ. С помощью меню дать пользователю возможность выбрать одну из следующих функций: - просмотр БД; - поиск по фамилии; - добавление нового факта.
Вариант 8	Реализовать следующие функции: - выбор текстового файла с помощью меню в текущем каталоге; - редактирование выбранного файла; - определять входит ли данное слово в выбранный файл.
Вариант 9	Создать базу данных студентов. Поля: ФИО, ГРУППА, СРЕДНИЙ ОЦЕНКА. Реализовать функции: - просмотр полей; - сортировка по ФИО.
Вариант 10	Создать базу данных сотрудников. Поля: ФИО, ДОЛЖНОСТЬ, ОБРАЗОВАНИЕ. Реализовать функции: просмотр полей со скроллингом; поиск по ФИО; редактирование полей.
Вариант 11	Дана база данных недопустимых в литературе слов и соответствующих допустимых слов. Создать программу, которая позволяет загружать текстовый файл, проверять текст и заменять недопустимые слова на литературные.
Вариант 12	Дан текстовый файл. Реализовать его вывод на экран с пометкой всех слов начинающихся на гласную букву и длиной более N.
Вариант 13	Дана строка (до 64 Кбайт) в текстовом файле. Создать динамическую базу предложений данной строки. По введенным начальным буквам (фразам) выдавать все предложение на экран. Для создания интерфейса использовать стандартные средства.
Вариант 14	Дан текстовый файл. Написать программу, которая создает динамическую базу данных предложений файла. А также предоставляет пользователю возможность по введенному слову выдавать все предложения, которые его содержат. Для создания интерфейса использовать стандартные средства.
Вариант 15	Дана база фактов элементов электрической принципиальной схемы. Написать программу, которая на основе фактов базы выводит на экран схему и выдает комментарии по каждому элементу.
Вариант 16	Написать программу, которая подсчитывает количество слов текстового файла, которые начинаются на глухую согласную букву.
Вариант 17	Дана база данных вопросов и ответов. Написать программу, которая выводит вопросы на экран в заранее заданном порядке, фиксирует ответы пользователя и проверяет их правильность.
Вариант 18	Написать программу, которая дает возможность пользователю: добавлять факты, если их не существует; удалять факты; выводить факты на экран.
Вариант 19	Дана строка (до 64 Кбайт). Создать множество слов данной строки, которые начинаются на букву, которую ввел пользователь, а также вывести это множество на экран. Использовать стандартные средства для разработки интерфейса.
Вариант 20	Дан текстовый файл и множество плохих слов. Написать программу, которая определяет плохие слова текстового файла и выводит их на экран. Для создания интерфейса использовать стандартные средства.
Вариант 21	Дан текстовый файл. Написать программу, которая удаляет плохие слова текстового файла. Для создания интерфейса использовать стандартные средства.
Вариант 22	Даны два множества слов в базе данных. Написать программу, которая определяет количество одинаковых слов. Для создания интерфейса использовать стандартные средства.
Вариант 23	Даны два текстовых файла. Написать программу, которая содержимое одного файла вставляет в середину другого файла. Для создания интерфейса использовать стандартные средства.
Вариант 24	Даны два текстовых файла. Написать программу, которая создает третий файл, в котором сохраняются общие слова двух данных файлов. Для создания интерфейса использовать стандартные средства.
Вариант 25	Дан текстовый файл предложений на естественном языке. Написать программу, которая подсчитывает количество слов того предложения, номер которого пользователь ввел с клавиатуры. Для создания интерфейса использовать стандартные средства.

Смешанное программирование на языкеVisual Prolog

С помощью Visual Prolog проектирование пользовательского интерфейса и связанных с ним окон, диалогов, меню, строки уведомлений о состояниях и т. д. производится в графической среде. Современная версия языка *Visual* Prolog обладает всеми средствами для быстрой разработки современных приложений. Она предоставляет возможность сочетать логическое, функциональное и объектно-ориентированное *программирование*.

В настоящее время язык *Visual* Prolog используется для создания систем управления ресурсами больших комплексов, экспертных систем, систем медицинской диагностики, обработки текстов на естественном языке и др.

Особенности версии Visual Prolog 7

Рассмотрим некоторые отличия версии Visual Prolog 7 от предыдущих версий, таких как TurboProlog, PDC Prolog и др.

Можно выделить следующие отличия:

- В Visual Prolog 7 между именем предиката и его аргументами надо ставить двоеточие, а после аргументов указывать режим детерминизма и направление потока аргументов для каждой переменной (входной или выходной или любой);
- Роль раздела *Goal* в Visual Prolog 7 играет предикат *run()*, расположенный в конце исходного кода;
- В Visual Prolog 7 можно использовать конструкцию *if-then-else-end if* и циклы, например, *foreach* и *repeat*;
- В Visual Prolog 7 объявить список можно сразу в разделе *class predicates* без использования раздела *domains* .

В языке Visual Prolog используются следующие разделы программы:

- ✓ (class) facts — объявление предикатов, описывающих факты (а также внутренних баз данных и фактов-переменных;
- ✓ (class) predicates — объявление предикатов;
- ✓ domains — объявление доменов (типов данных);
- ✓ constants — объявление констант;
- ✓ clauses — раздел предложений, которые определяют предикаты;
- ✓ goal — раздел, в котором определяется цель программы. Раздел goal может быть только один в проекте, он располагается в файле main.pro.

Ниже приведены разделы программы, которые связаны со структурой классов:

- ✓ class <имя класса> … end class <имя класса> — декларация класса;

- ✓ class <имя класса > : <имя интерфейса> … end class <имя класса> — декларация класса, порождающего объекты;
- ✓ interface <имя интерфейса> … end interface < имя интерфейса > — интерфейс;
- ✓ implement <имя класса > … end implement <имя класса > — имплементация класса;
- ✓ open — имена "открытых" классов и интерфейсов;
- ✓ properties — объявление свойств;
- ✓ constructors — объявление конструкторов (в декларациях классов).

Кроме этого, имеются разделы supports, resolve, inherits, delegates, predicates from и другие.

Создание консольного приложения.

Visual Prolog позволяет работать как с графическими приложениями(Graphical User Interface(GUI)), так и с консольными приложениями(Console).

Для того чтобы создать консольное приложение в системе Visual Prolog следует войти в среду разработки и выбрать команду меню задач Project>New. В открывшемся диалоговом окне Project Settings в поле Project Name нужно вписать имя проекта (например, Prim1), в поле Project Kind следует указать Console application. После нажатия кнопки Finish или Next создается проект (рис.1) (при нажатии кнопки Next появляется окно установки дополнительных параметров).

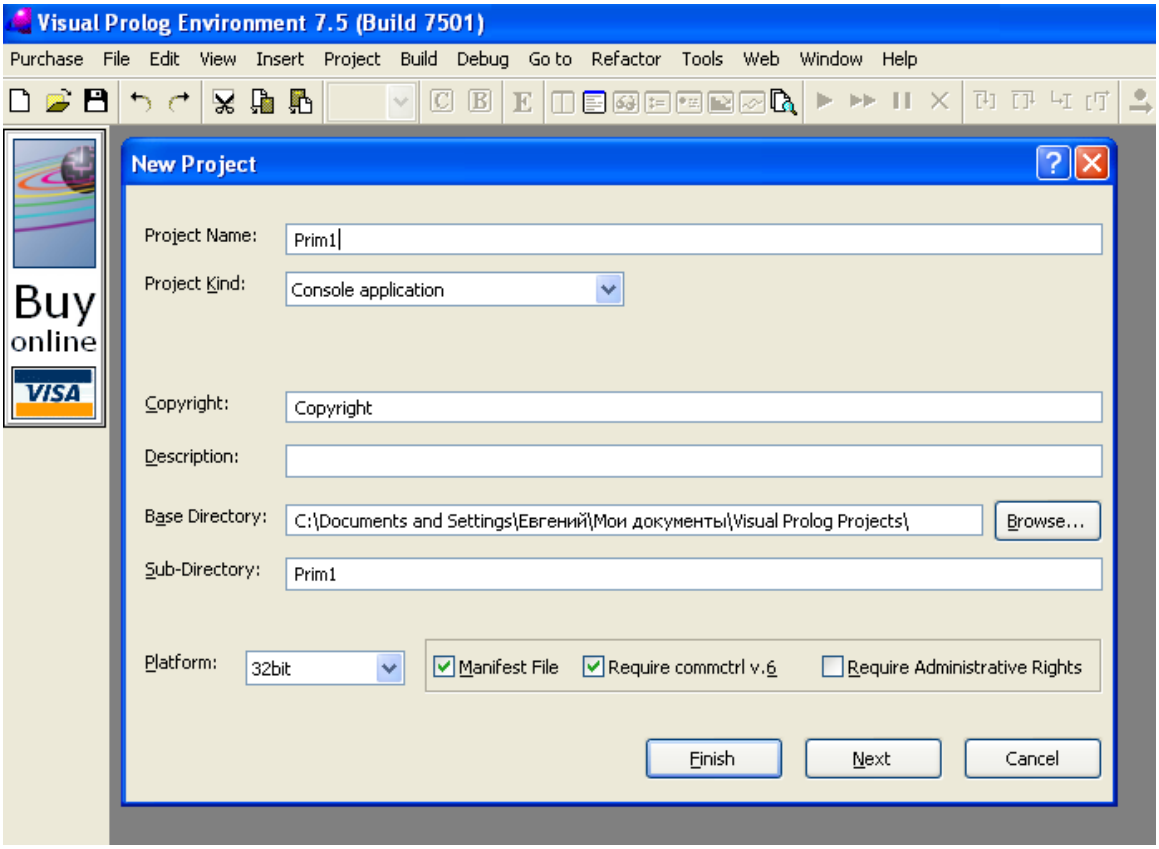

Рисунок 1 - Создание нового консольного приложения.

Создать проект можно с помощью команды меню Build>Build (или кнопки B панели инструментов), скомпилировать — команды Build>Compile (или кнопки C панели инструментов). Всякий раз, когда система будет спрашивать о добавлении директивы include <…>, можно просто отвечать Add All.

После компиляции в дереве (списке файлов проекта) появится файл main.pro консольного приложения Prim1 (рис. 2).

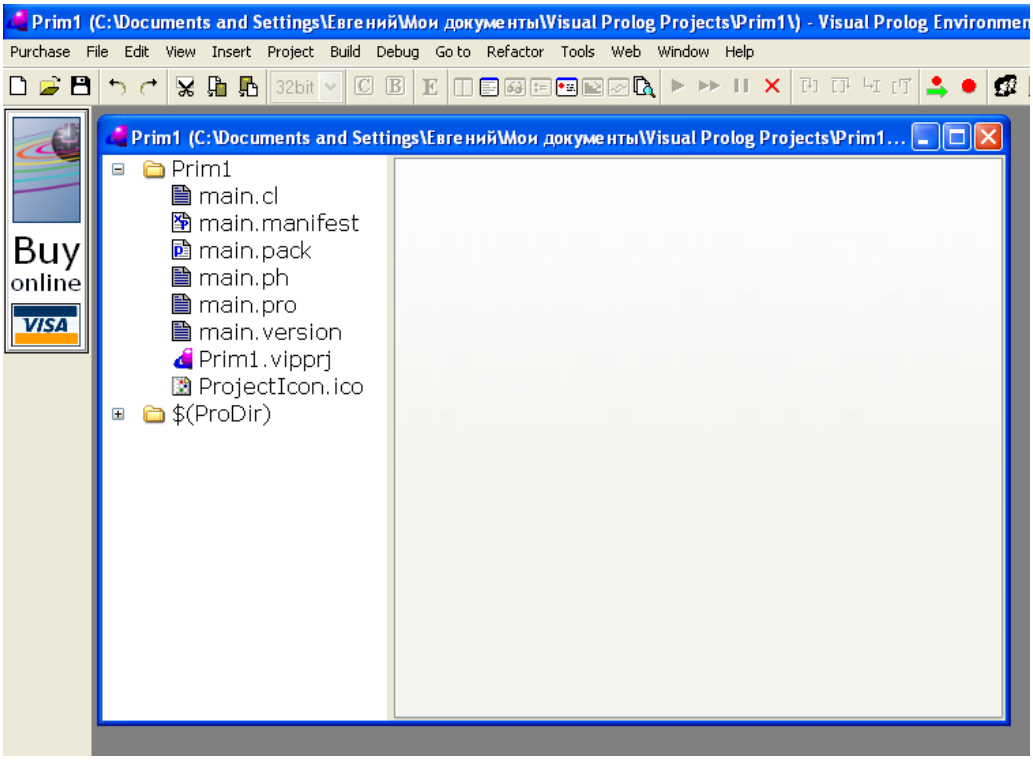

Рисунок 2 - Дерево проекта (консольного приложения)

Далее нужно открыть файл main.pro и в результате откроется окно редактора для написания кода на языке Visual Prolog (рис.3).

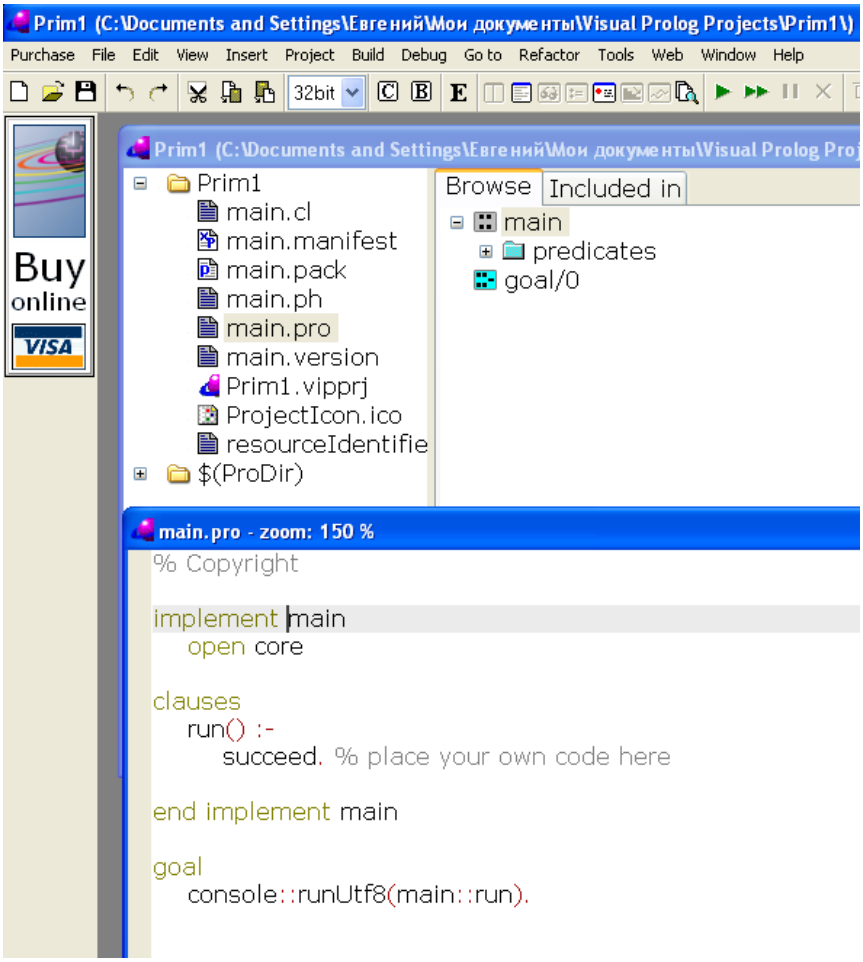

Рисунок 3 – Окно редактора с начальным кодом консольного приложения.

Создание базы фактов в отдельном файле

Факты базы данных можно хранить в отдельном файле (модуле) и подгружать их во время работы. Для этого необходимо сделать следующее:

- ✓ Создать (открыть) новый проект ConsDB(см. ранее) и выделить корень дерева проекта ConsDB (см. рис 4);

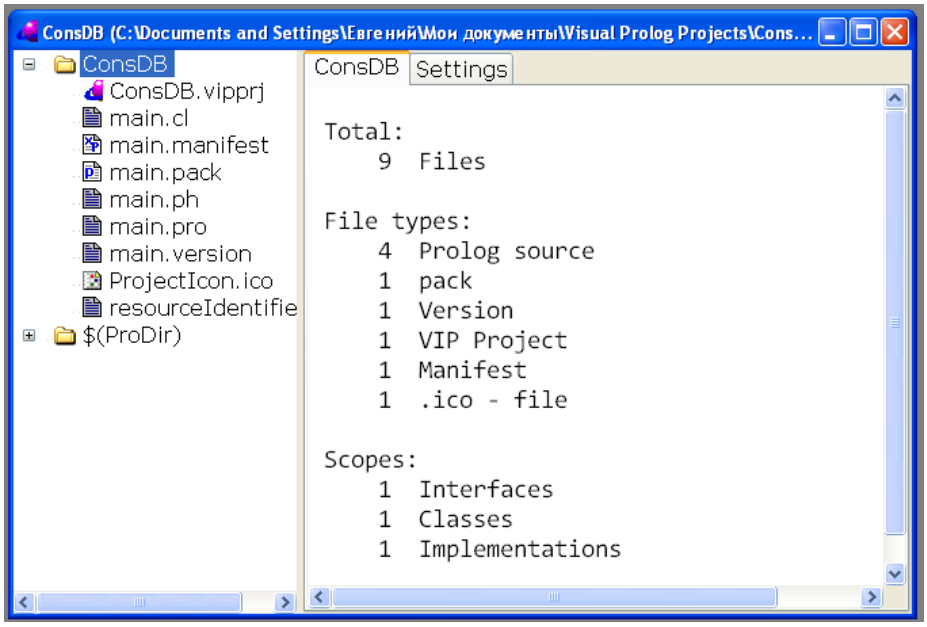

Рисунок 4 – Окно проекта ConsDB.

- ✓ выбрать команду меню File>New In New Package и в появившимся окне Create Project Item выбрать элемент Text File;
- ✓ в поле Name написать имя dbfile, а в поле Parent Directory вписать (или выбрать) слово Exe (рис. 5);

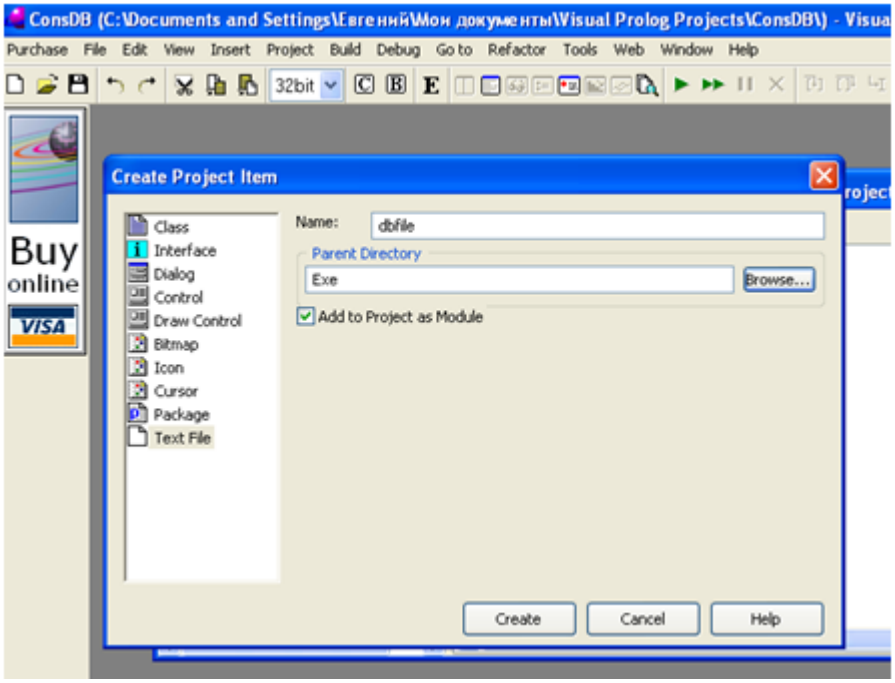

Рисунок 5 – Окно проекта ConsDB.

- ✓ Нажать кнопку Create, чтобы создать в директории Exe проекта файл dbfile.txt.
- ✓ открыть из дерева проекта файл dbfile.txt и поместить в него факты базы данных (см. листинг 1а).

```
clauses
who("Сергей","продавец").
who("Таня","кассир").
who("Семен","водитель").
who("Катя","безработная").
```

```
man("Сергей").
man("Семен").
woman("Таня").
woman("Катя").
```

Листинг 1а – Содержимое файла dbfile.txt.

- ✓ открыть из дерева проекта файл main.pro и поместить в него следующий код (см. листинг 1б).

```
implement main

class facts - relatives
who:(string Объект,string Работа).
man:(string).
woman:(string).

clauses
   run() :-
   console::init(),
   console::clearOutput, % чистка выходного потока (есть еще console::clearInput)
   file::consult("dbfile.txt", relatives),  % загрузка  базы фактов
   S=stdio::readline(),
   if  who(S,W) and man (S) then  stdio::write("Объект ",S," мужчина и он ",W) else
   if  who(S,W) and woman (S) then  stdio::write("Объект ",S," женщина и она ",W) else
   stdio::write("Объекта ",S," не существует ")
   end if end if,
   _ = console::readLine().
end implement main

goal
mainExe::run(main::run).
```

Листинг 1б – Код файла main.pro проекта ConsDB.

- ✓ Осуществить прогон программы с помощью клавиш Alt-F5 или из меню (Build->Run in Window).

В объявлении предиката указывается его имя, ставится знак двоеточия, а затем в круглых скобках через запятую перечисляются имена доменов (типов данных) аргументов:

class facts - relatives
who:(string Объект,string Работа).

Словом relatives обозначено имя базы данных. В объявлениях предикатов можно использовать комментарии специального вида. Слова Объект и Работа в этом объявлении обозначают комментарии. Такие комментарии пишутся в одно слово с прописной буквы.

Предикаты, объявленные в разделе class facts, определяются только в виде фактов. Если предикаты объявить в разделе class predicates, то в разделе clauses можно описывать факты и правила.

Цель программы формулируется в разделе goal, который находится в файле main.pro. Обычно в разделе goal только вызывается некоторый предикат, который используется для составления запросов. В данном примере таким предикатом является run.

В проекте ConsDB для получения частного ответа используется конструкция if-then-else-end if.

Чтобы вывести содержимое всей базы фактов, можно воспользоваться предикатом fail.

На листинге 2 приведен другой вариант кода файла main.pro проекта ConsDB (содержимое файла dbfile.txt не меняется), в котором используется предикат fail.

```
implement main
  open  console, file

class facts - relatives
who:(string Объект,string Объект).
man:(string).
woman:(string).

class predicates
   aout:() .

clauses
    aout():-who(X,Y), write(X," - ",Y), nl, fail.
    aout().

    run() :-init(), clearOutput,
   consult("dbfile.txt", relatives),  % загрузка  базы фактов
   aout,
    _ = readLine().
end implement main
 goal
mainExe::run(main::run).
```

Листинг 2 – Альтернативный код файла main.pro проекта ConsDB.

Если добавить имя класса в раздел open, то предикаты этого класса можно использовать без указания имени этого класса. В коде файла листинга 2 добавлены имена классов console и file, что позволяет:

- ✓ console::write(X," - ",Y) заменить на write(X," - ",Y);
- ✓ console::nl заменить на nl;
- ✓ console::init() заменить на init()
- ✓ console::clearOutput заменить на clearOutput
- ✓ file::consult("dbfile.txt", relatives) заменить на consult("dbfile.txt", relatives)
- ✓ _ = console::readLine() заменить на _ = readLine()

В итоге получили код программы, который очень близок к коду на языке Turbo-Prolog.

На листингах 3а и 3б приведены коды базы фактов и основной программы проекта Parent, а на рисунке 6 результаты работы данного проекта.

.

```
clauses
parent("Иван","Мария").          parent("Анна","Мария").      parent("Мария","Павел").
parent("Мария","Петр").          parent("Мария","Елизавета").
spouse("Иван","Анна").           spouse("Павел","Юлия").
male("Иван").          male("Павел").      male("Петр").
female("Мария").       female("Анна").     female("Елизавета").          female("Юлия").
```

Листинг 3а – Содержимое файла family.txt.

```
implement main
open console
class facts-relatives
parent:(string Родитель, string Ребенок).
spouse:(string Муж, string Жена).
male:(string).
female:(string).
class predicates
father:(string Отец,string Ребенок) nondeterm anyflow.
mother:(string Мать, string Ребенок) nondeterm (o,o).

clauses
father(X, Y):-  parent(X, Y), male(X).
mother(X, Y):- parent(X, Y), female(X).
run():-init(),console::clearOutput,
file::consult("family.txt",relatives),
father(X,Y),write("отец - ",X,", ребенок - ",Y),nl,fail;
mother(X,Y),write("мать -",X,", ребенок - ",Y),nl,fail;
if father("Иван","Петр") then write("\nИван является отцом Петра")
else write("\nИван не является отцом Петра") end if,
_=readLine().
end implement main
goal mainExe::run(main::run).
```

Листинг 3б – Код файла main.pro проекта Parent.

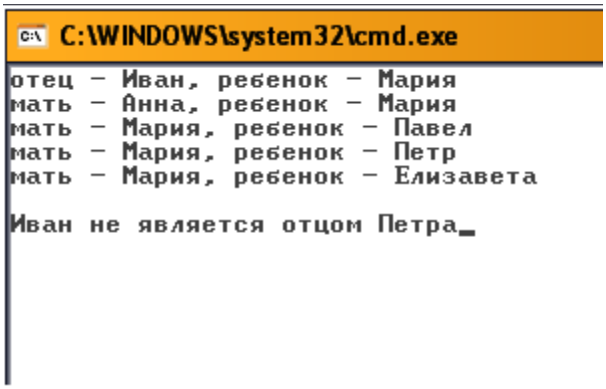

Рисунок 6 – Результаты работы проекта Parent.

Ключевое слово nondeterm (листинг 3б) в объявлении предиката означает, что область истинности этого предиката может содержать более одного элемента или не содержать ни одного. Ключевое слово anyflow означает, что некоторые аргументы предиката могут быть как входными, так и выходными. Последовательность (o,o) означает, что оба аргумента предиката — выходные, они возвращают некоторые значения.

Режимы детерминизма предикатов

В языке Visual Prolog при объявлении предикатов указывается режим детерминизма. Это позволяет повысить эффективность работы встроенных механизмов. Например, если для вычислений откат не нужен, то нет необходимости ставить точки возврата. Эта особенность предиката отмечается с помощью специального ключевого слова. В результате экономится память, и вычисления становятся быстрее.

В таблице 2 приведены режимы детерминизма предикатов. Из таблицы видно, что режим определяется количеством альтернативных решений при вызове предиката и успехом выполнения.

Таблица 2 - Режимы детерминизма

	>1 решения	≤ 1 решения	0 решений
Возможна ложь	nondeterm	determ	failure
Всегда истина	multi	procedure	Erroneous

Если объявленный режим детерминизма предиката не соответствует фактическому определению предиката, то компилятор выдает сообщение об ошибке или предупреждение.

Предикат succeed() является примером предиката с режимом procedure, предикат fail имеет режим failure.

Предикаты с режимом детерминизма procedure, называют процедурами. Если предикат не может порождать более одного решения, то он называются *детерминированным*, а если может, то *недетерминированным*.

По умолчанию используется режим procedure, если предикат объявляется в разделе (class) predicates, и режим nondeterm, если предикат объявляется в разделе (class) facts.

В объявлении предиката требуется указывать не только режим детерминизма, но и поток параметров (flow pattern). В нем описывается, какие аргументы предиката при его вызове являются входными, а какие выходными. Произвольный поток параметров обозначается с помощью ключевого слова anyflow. По умолчанию все аргументы предиката являются входными.

Поток параметров указывается в виде последовательности (i,o,o,…) , либо с помощью слова [out], которое ставится после имени домена аргумента предиката, например:

indicator: (integer Индикатор [out]) multi.

В тоже время предикат может иметь разные потоки параметров и режимы детерминизма. В этом случае они перечисляются последовательно. Например,

parent: (string, string) nondeterm (o,o) (i,o) (o,i) determ.

Факт-переменная

Факты-переменные являются аналогами глобальных переменных в процедурных языках программирования. Они объявляются следующим образом:

```
class facts
   счетчик : positive := 0.
   список : string* := [].
```

После знака двоеточия указывается домен факта-переменной, справа от знака := начальное значение. Для изменения значений фактов-переменных используется оператор присваивания :=.

На листинге 4 приведена программа, в которой подсчитывается в факт-перенной counter количество фактов с предикатом man.

```
implement main
open core,console
class facts
   man:(string).
   counter : positive := 0.
clauses
man("Петя").
man("Саша").
man("Павел").

run() :-  man(_),
            counter := counter + 1,
            fail;
            write(counter),
      _ = readLine().
end implement main
goal
   console::run(main::run).
```

Листинг 4 – Использование факт-переменных.

Организация циклов

Циклы можно организовать несколькими способами. Ранее уже использовался цикл на основе предиката fail, в котором тело цикла выполняется столько раз, сколько фактов в базе данных.

Можно использовать цикл foreach, который, также как и цикл на основе предиката fail, завершается, когда не остается точек возврата. Он имеет структуру:

foreach <действие с возвратом> do <процедура> end foreach.

На листинге 5 приведен пример, в котором происходит вывод авторов книг, которые в базе хранятся под четными номерами.

```
implement main
   open core,console
class facts
   book: (integer Номер, symbol Автор, symbol Название).
clauses
   book(1, "Толстой", "Война и мир").
   book(2, "Чехов", "Вишневый сад").
   book(3, "Пушкин", "Евгений Онегин").
   book(4, "Достоевский", "Преступление и наказание").
run():-
write("Авторы и их произведения\n\n"),
   book (_, Author, Title),
   write(Author, " - ", Title), nl, fail;
      nl,
      foreach  book(N, Author, _), N mod 2 = 0 do
         write(Author), nl
      end foreach,
      _ = readLine().
end implement main
goal  console::run(main::run).
```

Листинг 5 – Использование цикла foreach.

Можно использовать цикл repeat, который в отличие от предыдущих циклов может использоваться в двух вариантах:

- ✓ когда в повторяемом действии есть откат;
- ✓ когда нет отката.

Цикл repeat имеет структуру:

repeat() <действие> <условие выхода из цикла>.

Определение предиката repeat имеет вид:

```
class predicates
   repeat: () multi.
clauses
   repeat().
   repeat():-
      repeat().
```

Этот предикат ничего не делает, кроме того, что ставит точки возврата, потенциально бесконечное число раз. Условие выхода из цикла должно быть истинным только в момент самого выхода. В остальное время оно вызывает откат. Если в разделе <действие> нет точек возврата, то откат идет к предикату repeat.

На листинге 6 приведен код программы с использованием цикла std::repeat(). Данная программа позволяет вводить символы, преобразовывать их,а также выводить символы с задержкой и подсчитывать их количество.

```
        % Copyright
implement main
    open core,console
class facts
        c: positive := 0.
clauses
run():-clearOutput(),  % очищает выходной буфер
    std::repeat(),
        Char = readChar(), % считывание символа из буфера ввода
        if not(Char = '\n') then c:=c+1 end if, % подсчет количества символов
        UpperCaseChar = string::charUpper(Char),
        programControl::sleep(500), %  1000 = 1 сек
        write(UpperCaseChar),
    Char = '\n',  % условие выхода из цикла
    !,
    clearInput(),  % очищает буфер ввода
    write(c),                  % вывод факт-переменной
    _ = readLine();
    succeed().
end implement main
goal
    mainExe::run(main::run).
```

Листинг 6– Использование цикла repeat.

Предикаты базы фактов

Динамическая база данных (аналогично Turbo-Prolog) может загружаться из файла на диске и записываться в файл. Объявление поименованной базы фактов было рассмотрено выше.

Имена внутренних баз данных принадлежат домену factDB. База данных может не иметь имени. В имплементации одного и того же класса может быть несколько безымянных внутренних баз данных. Но если требуется загружать базу данных из файла или сохранять ее в файле, она должна иметь имя.

Ниже приведены примеры использования предикатов для работы с фактами:

- ✓ asserta(man(“Сергей”)) - добавляет факт в начало базы данных (assert и assertz добавляют факт в конец базы данных).
- ✓ retract(man(“Сергей”)) - удаляет первый факт, который унифицируется с аргументом этого предиката. Этот предикат принимает значение *ложь*, если факт, который требуется удалить, отсутствует в базе данных.
- ✓ retractAll(man(“Сергей”)) удаляет все факты, которые унифицируются с аргументом этого предиката. Этот предикат всегда успешный, значения аргументов он не возвращает.
- ✓ retractFactDb(<имя базы фактов>) удаляет все записи из базы данных.
- ✓ file::save(<имя файла>,<имя базы фактов>) - сохранение в файл фактов базы данных. Если указанного файла нет, то он будет создан.
- ✓ file:: consult (<имя файла>,<имя базы фактов>) - загружает факты базы данных из файла, при этом они добавляются к текущему состоянию базы данных.
- ✓ file::reconsult(<имя файла>,<имя базы фактов>) - заменяет текущее состояние базы данных фактами из файла.
- ✓ file::existFile(<имя файла>) – проверка существования файла.

Можно выдавать сообщения пользователю, если файла не существует, например:

```
constants  fileName="dbfile.txt".
……….
if not(existFile(fileName))
then  write("Файл ",fileName," не существует !" )  end if
```

Если содержимое файла базы фактов не соответствует формату описания, то это можно выявить так:

```
        ……..
constants  fileName="dbfile.txt".
class facts - relatives
……..
clauses
   run() :-
      existFile(fileName),!,
      try consult(fileName, relatives) % обработка ошибок
      catch Error do
         writef("Ошибка %. Невозможно загрузить базу фактов из файла %\n", Error, fileName)
      end try;
      ……
```

Работа с файлами

Рассмотрим пример работы с файлами, который приведен на листинге 7. Программа, представленная на данном листинге, определяет способ кодировки файла a.txt, а именно

Unicode или нет. Если способ кодировки Unicode, то из файла a.txt считываются строки и записываются в новый файл с именем aout.txt в кодирове ANSI.

```
implement main
   open core,console
class predicates
   type: (string FInput, stream::mode Mode) procedure (i,o).
   rewrite: (string, stream::mode, string FOutut)    determ (i,o,i).
clauses
   type(FInput, stream::unicode()):- file::isUnicode(FInput), !.
   type(_, stream::unicode()).
clauses
   rewrite(FInput, Mode, FOutut):-
      file::existFile(FInput),
      type(FInput, Mode),
      Input = inputStream_file::openFile(FInput, Mode),
      F = outputStream_file::create8(FOutut),
      std::repeat(),
         Str = Input:readLine(),
                  F:write(Str), F:nl,
      Input:endOfStream(),
      !,
      Input:close(),
      F:close().
  run() :-
  FInput = "a.txt", % кодировка Юникод
  FOutut = "aout.txt",
      rewrite(FInput, Mode, FOutut),!,
      writef("FileName %\nMode: %\n", FInput, Mode);
      _ = readLine().
end implement main
goal  console::runUtf8(main::run).
```

Листинг 7– Перекодирование файла.

Предикат isUnicode проверяет, используется ли для файла кодировка символов Юникод. Конструктор openFile открывает файл для чтения, предикат close закрывает файл. Предикат create8 создает ANSI-файл для записи. Предикат endOfStream проверяет, достигнут ли конец файла. Предикат readLine() считывает файл в строку, а write(Str) записывает строку в новый файл.

Рекурсия

Рассмотрим пример листинга 8, в котором используется недетерминированный рекурсивный предикат. В представленном примере на основании фактов «родитель-ребенок» определяется отношение «предок-потомок». Программа выдает результаты, которые соответствуют дереву, представленному на рисунке 7.

```
implement main
   open console
class facts - relatives
   rod_reb: (string Родитель, string Ребенок).
clauses
```

```
   rod_reb("Саша", "Лена").
   rod_reb("Галя", "Лена").
   rod_reb("Лена", "Сергей").
   rod_reb("Лена", "Дима").
   rod_reb("Дима", "Света").
class predicates
   pred_pot: (string Предок, string Потомок) nondeterm (o,o) (i,o).
clauses
   pred_pot(X, Y):-  rod_reb(X, Y).
   pred_pot(X, Y):-  rod_reb(X, Z), pred_pot(Z, Y).
   run():-
      pred_pot(X, Y), write(X, " - ", Y), nl, fail;
      _ = readLine().
end implement main
goal   console::run(main::run).
```

Листинг 8– Определение потомков.

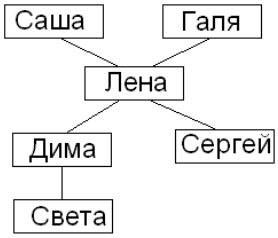

Рисунок 7 – Дерево потомков.

Работа с атрибутами текста

В языке Visual Prolog имеются средства, которые позволяют использовать различные цвета фона и символов, события мыши, события нажатия клавиши и др. Программа, приведенная на листинге 9, выводит цветные полосы и коды цвета в диапазоне от 1 до 15. В данной программе используется функциональный стиль программирования. Функции объявляются следующим образом:

class predicates

имя функции: (домен1, домен2, ...) -> домен значений.

В программе листинга 9 используются генератор случайных чисел, repeat-цикл, задержка и события нажатия мыши.

```
implement main
open core, console, console_native
class predicates
   getAttribute: (unsigned16 ЦветФона, unsigned16 ЦветТекста)
      -> unsigned16.
clauses
   getAttribute(BgColor, TextColor) =
      bit::bitOr(bit::bitLeft(BgColor, 4), TextColor).
class predicates
   stop: () determ.
   process: ().
clauses
   stop():-  L = getEventQueue(),
```

```
            mouse( _, _, 1, _, 0) = list::getMember_nd(L), !.
    process():-
        std::repeat(),
            N = 1 + math::random(15),
            setTextAttribute(bit::bitLeft(N, 4)),
            write(" ",N," "),
            programControl::sleep(10), stop(), !.
    process().
    run():-
        setLocation(coord(1,5)),% установка курсора x=1,y=5
        setTextAttribute(getAttribute(3, 10)), % установка атрибутов текста
        write("Остановка с аомощью мышки!"),
        programControl::sleep(2000), % 1 сек
        clearOutput(), % очищает буфер вывода
        setLocation(coord(0, 1)),
        process(),
        _ = readLine().
end implement main
goal  console::run(main::run).
```

Листинг 9 – Управление интерфейсом.

Если необходимо остановить цикл по нажатию клавиши, то вместо

mouse(_, _, 1, _, 0) = list::getMember_nd(L)

необходимо написать

key(_, _, _, _, 'z', _) = list::getMember_nd(L)

Ниже приведен список основных предикатов данного примера:

- ✓ getEventQueue - возвращает для окна консоли список текущих событий;
- ✓ getMember_nd - недетерминированно возвращает произвольный элемент списка;
- ✓ setTextAttribute - устанавливает в консоли атрибуты текста;
- ✓ setLocation - помещает курсор в точку с заданными координатами;
- ✓ random - генерирует целое неотрицательное случайное число.

Создание модулей

Чтобы создать модуль, необходимо выделить корень дерева проекта и выбрать команду меню New In New Package. В открывшемся диалоговом окне Create Project Item необходимо выбрать в левом поле раздел Class (см. рис. 8), убрать флажок из поля Create Interface, ввести в поле Name имя модуля (mod1) и нажать на кнопку Create. В результате будет создана папка mod1и ряд файлов модуля.

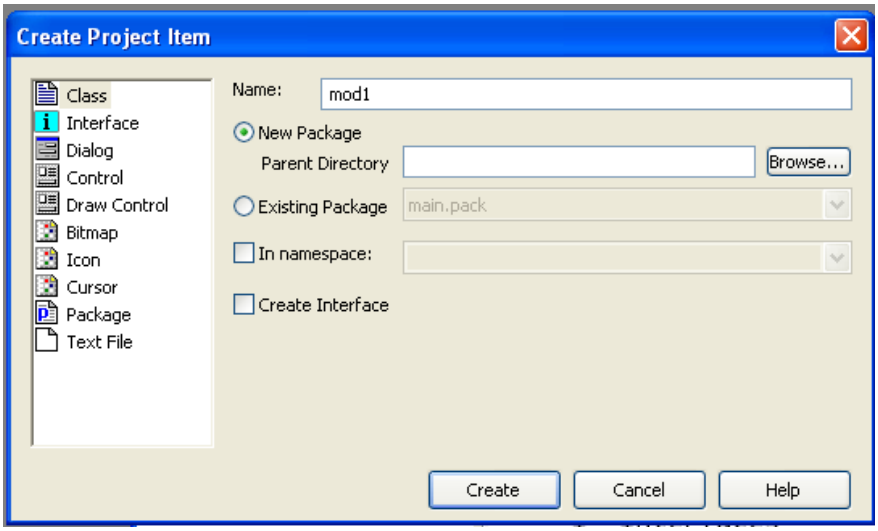

Рисунок 8 – Создание модуля.

Далее в декларации класса (файл mod1.cl) нужно объявить предикат run, с помощью которого будет указываться цель:

predicates run: ().

Раздел open имплементации класса (файл mod1.pro) должен иметь вид:

open core, console

Далее можно включить в проект вызов модуля, т.е. пишется имя класса, ставится двойное двоеточие, а затем вводится имя предиката.

В таблице 3 приведены коды файлов mod1.cl, mod1.pro и main.pro. Приведенные коды реализуют вывод двух строк, одна из которых выводится в основном файле, а другая в модуле.

Таблица 3.

mod1.cl	mod1.pro	main.pro
class mod1 open core predicates run: (). end class mod1	implement mod1 open core,console clauses run():- write("Строка модуля") . end implement mod1	implement main open core,stdio clauses run() :- write("Строка основной программы") ,nl, mod1::run(), % вызов модуля succeed. end implement main goal console::run(main::run).

Списки

При выполнении операций, где тип элементов не важен, например, при вычислении длины списка, для объявления предиката обработки списков можно использовать полиморфные домены. Имя полиморфного домена в объявлении предиката пишется с прописной буквы, например:

class predicates length: (Name) -> positive.*

или *class predicates length: (N*) -> positive.*

Имена Name или N являются переменными, и обозначают произвольный домен элементов списка. Предикаты, аргументы которых могут принадлежать произвольным доменам, называются *полиморфными* предикатами. Определение пролиморфного предиката не зависит от типа аргументов. Например, предикат вычисления длины списка одинаково определяется для списка чисел, списка строк и т. д. Но предикат, который вычисляет сумму элементов списка, не может быть полиморфным. Полиморфизм, в котором предикаты определяются независимо от типа аргументов, называется *параметрическим*.

Имена доменов могут иметь аргументы — параметры, которые представляют также имена доменов. В качестве аргументов имени домена в объявлении этого домена могут указываться только переменные. Аргументы имен доменов заключаются в фигурные скобки, они являются попарно независимыми. На листинге 10 приведен пример объявления предикатного домена и принадлежащих ему предикатов.

```
% Посчет количества и суммы элементов
implement main
   open core,console
domains
   op{A, B} = (A*) -> B.
class predicates
   length : op{A, positive}.    % длина списка
   sum : op{integer, integer}.    % сумма элементов списка
clauses
   length([]) = 0.
   length([_ | T]) = 1 + length(T).

   sum([]) = 0.
   sum([H | T]) = H + sum(T).

 run():- write("N=",length([1, 2, 3,4,5]), " Sum=",sum([1, 2, 3,4,5])),
       _ = readLine().
end implement main
goal  console::run(main::run).
```

Листинг 10 – Использование предикатного домена.

Функции вычисления длины списка и суммы его элементов определены выше без применения хвостовой рекурсии. Поэтому их нельзя использовать для списков, содержащих большое количество элементов. На листинге 11 приведено определение операций с использованием хвостовой рекурсии.

```
% Посчет количества элементов различных типов
implement main
   open core,console
class predicates
   length: (A*) -> positive.
   length: (A*, positive) -> positive.
clauses
   length(L) = length(L, 0).
   length([], N) = N.
   length([_ | T], N) = length(T, N + 1).

   run():- write(length([1, 2, 3,4,5]) + length(["C", "D"])),
      _ = readLine().
end implement main
goal  console::run(main::run).
```

Листинг 11 – Использование хвостовой рекурсии.

В программе листинга 12 определяются операции проверки принадлежности элемента списку, а также возвращения произвольного элемента списка, в предикатном и в функциональном стиле. Операция определяется рекурсивно, в соответствии с правилом — *элемент принадлежит списку, если он совпадает с головой списка или принадлежит его хвосту.*

```
        implement main
   open core,console

class predicates
   member_dt: (A, A*) determ.
   member: (A [out], A*) nondeterm.
   member_nd: (A*) -> A nondeterm.
clauses
   member_dt(H, [H | _]):- !.   % проверка принадлежности
   member_dt(H, [_ | T]):-
      member_dt(H, T).

   member(H, [H | _]).    % возвращение элемента списка
   member(H, [_ | T]):-
      member(H, T).

   member_nd([H | _]) = H.
   member_nd([_ | T]) = member_nd(T).

   run():-
      X = 2, L = [1, 2, 3],
      if member_dt(X, L) then
         writef("Элемент % принадлежит списку %\n\n", X, L)
      else
         writef("Элемент % не принадлежит списку %\n\n", X, L)
```

```
    end if,

    writef("Элементы списка %:\n", L),
    foreach member(Y, L) do
      write(Y), nl
    end foreach,

    L1 = ["H", "E", "L", "L", "O"],
    writef("\nЭлементы списка %:\n", L1),
    write(member_nd(L1)), nl,
    fail;
    _ = readLine().
end implement main
goal  console::run(main::run).
```

Листинг 12 – Проверка принадлежности элемента к списку.

В версии 7.5 языка Visual Prolog имеется оператор in, который можно использовать для выполнения опреаций принадлежности к списку, например:

```
implement main
  open core,console
clauses
  run() :-S=" принадлежит списку", X="B",
          if X in ["A","B","C"] then write(X,S)  else write(X,"не",S)
          end if,
          succeed.
end implement main
goal  console::run(main::run).
```

На листинге 13 приведен код программы, который реализует разбивку списка на два, в первом накапливаются четные, а во втором нечетные элементы.

```
implement main
    open core,console
class predicates
    sp: (unsigned*, unsigned* [out], unsigned* [out]).
clauses
    sp([A | L], [A | L1], L2):-
       A mod 2 = 0, !,
    sp(L, L1, L2).
    sp([A | L], L1, [A | L2]):-  sp(L, L1, L2).
    sp([], [], []).
    run():-
       sp([1, 2, 3, 4, 5, 9, 8, 7, 6,8,9], L1, L2),
       write(L1), nl,
       write(L2),
       _ = readLine().
end implement main
goal    console::run(main::run).
```

Листинг 13 – Разбивка списка.

Анонимные предикаты

Для определения анонимного предиката используются фигурные скобки. Это определение должно состоять только из одного предложения, при этом в заголовке правила отсутствует имя предиката, пишутся только его аргументы (см. листинг 14). Заголовок может быть пустым.

```
implement main
    open core,console
clauses
   run():-
    write({(X) = X *2}(5)), nl, % нач. значение X =5 умножается на 2
    F = {(X, Y):- X > Y},
    if F(3^2, 2^3) then write("yes") % сравнение двух выражений
                   else write("no") end if,
    _ = readLine().
end implement main
goal  console::run(main::run).
```

Листинг 14 – Определение анонимного предиката.

Анонимные предикаты можно определять как в функциональном стиле, так и в предикатном стиле, как это было показано выше. Они могут быть как детерминированными, так и недетерминированными. Их особенность заключается в том, что все их аргументы должны быть входными.

Анонимные предикаты принадлежат соответствующим предикатным доменам и могут использоваться в аргументах предикатов высших порядков (см. листинг 15).

```
implement main
    open core,console
clauses
  domains   op = (real, real) -> real.
class predicates   fun: (string) -> op.
clauses
    fun("+") = {(X, Y) = X + Y}:- !.
    fun("*") = {(X, Y) = X * Y}:- !.
    fun(_) = {(X, _) = X}.
    run():-
```

```
        write(fun("+")(10, 20) / fun("*")(5, 2)),
    _ = readLine().
end implement main
goal  console::run(main::run).
```

Листинг 15 – Анонимные предикаты в предикатах высших порядков.

Предикаты высших порядков — это предикаты, среди аргументов которых имеются другие предикаты. Например, в коде листинга 16 используются предикаты второго порядка, определенные в классе list. Данный код реализует наращивание элементов списка на 2 и отбрасывание нечетных элементов из списка.

```
implement main
    open core,console
clauses
run():-
    write(list::map([1, 2, 3, 4, 5], {(X) = X + 2})), nl,
    write(list::filter([1, 2, 3, 4, 5], {(X):- X mod 2 = 0})),
    _ = readLine().
end implement main
goal  console::run(main::run).
```

Листинг 16 – Предикаты второго порядка.

При использовании предиката map (см. листинг 16) и предиката filter создается новый список. Анонимный предикат (он определен прямо в аргументе каждого из этих предикатов), в первом случае является процедурным (он определен в виде функции), а во втором — детерминированным.

На листинге 17 приведен пример сортировки, в котором формируется упорядоченный список из элементов исходного списка. Изначально формируемый список пуст. Далее на каждом шаге берется голова исходного списка и вставляется формируемый список в нужное место.

```
mplement main
    open core,console
class predicates
    insertionSort: (A* List) -> A* SortedList.
    sort: (A* List, A* SortedList) -> A* SortedList.
    insert: (A, A* SortedList) -> A* SortedList.
clauses
    insertionSort(L) = sort(L, []).
    sort([H | T], L) = sort(T, insert(H, L)).
    sort([], L) = L.

    insert(A, [B | L]) = [B | insert(A, L)]:-
        B < A,
        !.
    insert(A, L) = [A | L].
```

```
    run():-
        R = 20, % числа в диапазоне от 0 до 19 (другой вариант   R = upperBound( integer )
),
        L = [math::random(R) || _ = std::fromTo(1, 10)], % количество чисел 10
        write(L, "\n", insertionSort(L)),
        _ = readLine().
end implement main

goal
    console::run(main::run).
```

Листинг 17 – Сортировка списка.

Порядок выполнения работы(Visual Prolog)

1. Изучить структуру языка Visual Prolog .
2. Изучить работу основных встроенных предикатов (объектов).
3. Сформулировать основные отличительные особенности языка Prolog и Visual Prolog.
4. Выполнить задания в соответствии с вариантом, полученным у преподавателя.

Требования к отчету(Visual Prolog)

Отчет по каждому заданию должен включать:

- Номер варианта и задание;
- Результаты извлечения, структуризации и формализации семантической информации;
- Структурную карту Константайна (схему вызовов микромодулей);
- Текст отлаженной программы с комментариями;
- Фрагмент базы данных (если данные в отдельном файле);
- Результаты работы программы;
- Заключение (перечисляются, какие основные механизмы языка Prolog были использованы).

Порядок проведения защиты:

- Ответить на вопросы, предложенные преподавателем по теоретическому материалу.
- рассказать принцип работы программы, выполненной в соответствии с вариантом задания программы.

Контрольные вопросы (Visual Prolog)

1. Какие разделы имеются в языке Visual Prolog?
2. Как создать проект в консольном режиме в Visual Prolog?
3. Какие циклы имеются в языке Visual Prolog?
4. Как создать базу фактов в отдельном файле?
5. Какие существуют режимы детерминизма предикатов?
6. Как создать модуль в языке Visual Prolog?
7. Какие операции со списками можно реализовать в языке Visual Prolog?
8. Какие виды переменных имеются в языке Visual Prolog?
9. Какие переменные можно использовать в языке Visual Prolog?
10. Приведите примеры конструкций фактов и правил на язые Visual Prolog?
11. Как работает программа, приведенная на листинге №X?

Задания к 1-й части лабораторных работ (Visual Prolog)

Общие требования:

1. Факты должны подгружаться с диска;
2. Использовать новые возможности программирования (функциональное и др.);
3. Обеспечить приемлемую универсальность программы;
4. Код программы должен быть унифицирован;
5. Обеспечить независимость модулей (микромодулей, базы фактов и др.).

Выполнение задания предполагает следующие шаги:

- выделить существенную информацию из текста задания;
- определить и классифицировать семантические единицы;
- определить структуры фактов;
- определить логические цепочки фактов;
- сформировать правила (представить в формализованном виде);
- выполнить кодирование, тестирование и отладку программы, которая позволяет формировать заключение.

Таблица 4 - Варианты заданий

Вариант	**Задание (реализовать на языке Visual Prolog)**
Вариант 1	Детская скарлатина Это контактная и очень заразная инфекция, которая может передаваться воздушно-капельным путем. Болезнь скарлатина в основном возникает в холодное время года, пик заболеваемости приходится на октябрь — ноябрь и февраль-апрель. Скарлатина у детей может возникать внезапно, на фоне видимого общего благополучия. Симптомы скарлатины развиваются постепенно, начиная с воспалительных реакций в месте входных ворот инфекции. Симптомы скарлатины у детей до момента высыпания на коже могут напоминать клиническую картину ангины или тонзиллита. Признаки скарлатины включают в себя боль в горле, резкое повышение температуры тела до экстремально высоких цифр, увеличение, уплотнение и болезненность подчелюстных лимфатических узлов. Ребенок становится вялым, капризным, отказывается от приема пищи, жалуется на сильную головную боль и ломоту в крупных мышцах. Могут возникать спонтанные боли в области сердца. Признаки скарлатины у детей включают в себя быстрое появления мелкой точечной сыпи насыщенного красного цвета. Обычно она возникает спустя 12-24 часа после повышения температуры тела. Поэтому спутать первые признаки скарлатины с другими воспалительными заболеваниями достаточно сложно.
Вариант 2	Средний отит Это воспалительное заболевание среднего уха, которое отделено от наружного барабанной перепонкой. Следовательно, данная патология является более серьезной, чем наружный отит, хотя их и не всегда получается отличить друг от друга. Главные причины развития среднего отита: 1. Проникновение в барабанную полость инфекции. 2. Травмы. Средний отит практически всегда сопровождается сильной болью в ухе. Она значительно усиливается во время жевания и глотания, в результате чего пациент может отказываться от приема пищи. Также значительное усиление болевого синдрома отмечается при нажатии на ушную раковину и потягивании за ухо. Чтобы уменьшить боль, пациент ложится на бок, соответствующий больному уху. У детей этот симптом выражен еще более ярко. Кроме того, при среднем отите присутствуют следующие симптомы: Временная потеря слуха, а если барабанная перепонка будет разрушена инфекционным процессом, то может развиться полная глухота на одно ухо. Повышение температуры тела, общее недомогание. Неприятные ощущения в ухе, как и при наружном отите: звон, шум, заложенность. Сильные боли в ухе при среднем отите могут продолжаться достаточно долго, а при отсутствии адекватного лечения заболевание может переходить в хроническую форму, и приводить к серьезным осложнениям.
Вариант 3	Евстахиит Евстахиева труба – это тоненький канал, который соединяет глотку и барабанную полость (среднее ухо). При острых респираторных заболеваниях в нее может попасть

	инфекция, приводя к развитию евстахиита. При данной патологии боль в ушах может иметь разную интенсивность. Иногда она очень сильная, а в некоторых случаях отсутствует вовсе. При евстахиите имеет место следующая симптоматика: заложенность в ушах; пациент слышит треск, шум; собственный голос слышится очень громко, в то время как в общем слух нарушен, речь окружающих воспринимается плохо; в ухе как будто переливается жидкость: если вам попадала в уши вода во время плавания, то вам знакомо это неприятное ощущение. При отсутствии лечения евстахиит зачастую переходит в хроническую форму, и в дальнейшем служит причиной возникновения частых гнойных отитов, снова и снова приводя к появлению острых сильных болей в ухе. Диагностика и лечение патологии осуществляются ЛОР-врачом. Предусмотрена медикаментозная терапия с применением антибактериальных препаратов.
Вариант 4	При сахарном диабете выраженность симптомов зависит от степени снижения секреции инсулина, длительности заболевания и индивидуальных особенностей больного. Как правило симптомы диабета 1 типа — острые, болезнь начинается внезапно. При диабете 2 типа состояние здоровья ухудшается постепенно, в начальной стадии симптоматика скудная. При заболевании избыток сахара (глюкозы) накапливается в крови. Ваши почки вынуждены интенсивно работать для того, чтобы фильтровать и поглощать избыток сахара. Если ваши почки не справляются, избыток сахара выводится из организма в моче с жидкостью из тканей. Это вызывает более частое мочеиспускание, которое может привести к обезвоживанию. Появляется усталость, вызванная обезвоживанием, частым мочеиспусканием и неспособностью организма функционировать должным образом, потому что меньше сахара можно использовать для получения энергии. Появляется жажда, но не к воде, а к пище. Человек ест и при этом чувствует не сытость, а заполнение желудка пищей, которое затем достаточно быстро превращается в новый голод. При диабете I типа (инсулинозависимому) потеря веса. Стоит отметить, что похудание проходит на фоне повышенного аппетита и обильного питания, что не может не настораживать. Достаточно часто сброс веса приводит к истощению. Имеют место проблемы проблемы со зрением, а также медленное заживление ран или частые инфекции. Может быть покалывание в руках и ногах, красные, опухшие, чувствительные десны.
Вариант 5	В зависимости от вида клеток, из которых сформировалась опухоль, выделяют мелкоклеточный рак легкого и немелкоклеточный. Симптоматика среди мужчин и женщин схожа, не имеет кардинальных отличий. Как правило, при развитии опухоли длительное время не наблюдается каких-либо настораживающих, неприятных признаков. Это характерная особенность при длительном росте опухоли. Признаки рака легких на разных стадиях: Опухоль находится в одном месте, не дает метастаз, диаметр образования не более 4 см. Внешние и рентгенологические симптомы на этой фазе отсутствуют либо проявляются настолько незначительно, что человек не придает им значения. На этой фазе беспокоит кашель, головная боль, общее недомогание, снижение аппетита, повышение температуры. Диагностировать рак легких по этим симптомам, исходя из жалоб пациента, невозможно. Для этого следует сделать МРТ или флюорографию. Вторая стадия может иметь первичные метастазы в лимфоузлах, размер новообразования растет. Симптомы все еще смазанные, но уже более ощутимые, что должно насторожить пациента и врача. В этот период могут начаться кровохарканья, боли в груди, появление хрипов при дыхании наряду с вышеописанными симптомами. На третьей стадии рак может быть диагностирован на основе жалоб больного и медобследования. Подтверждением становится наличие метастазов во всех региональных лимфоузлах, некоторых отдаленных. Опухоль разрастается настолько, что выходит за пределы легкого. Симптомы такие же, как на второй стадии, но с более высокой интенсивностью. Отмечается мокрый кашель со слизью, иногда с кровью и гноем, дыхание затруднено, одышка, боль в горле при глотании. На четвертой стадии наблюдается тяжелое течение болезни с яркими симптомами. Кашель становится постоянным, беспокоящим и надсадным, регулярные кровохарканья, накапливается жидкость в легком при раке, боль в груди разной интенсивности. Помимо дыхательной системы, к признакам рака легких относят симптомы со стороны сердечно сосудистой, пищеварительной. Связано это с увеличением, разрастанием опухоли.
Вариант 6	При туберкулезе важно не пропустить первые симптомы, когда шанс вылечить заболевание остается высоким. Однако, зачастую туберкулез легких долгое время протекает без заметной симптоматики, и обнаруживается совершенно случайно, например, при проведении флюорографии.

	Для большинства форм туберкулёза лёгких характерны следующие признаки: - Общее состояние человека — взрослые с ограниченными формами туберкулеза жалуются на повышенную утомляемость, слабость, особенно выраженную в утренние время, также характерно понижение работоспособности. - Кашель. От сухого до влажного, с заметными отделением мокроты. Она может быть творожистого, гнойного вида. При присоединении крови – принимает вид от «ржавой» до примеси жидкой, не изменённой (кровохаркание). - Общий вид: больные теряют в весе до 15 и более килограмм, поэтому выглядят худыми, лицо бледное, черты лица заостряются и потому оно кажется более красивым, на фоне бледной кожи заметен румянец на щеках. - Одышка. Обусловлено сокращением дыхательной поверхности лёгких при воспалении и склерозировании (рубцевании). - Увеличение температуры тела: при ограниченных формах повышение температуры незначительное (37,5-38 С), но продолжительное. - Температура повышается вечером или в ночное время, ночью наблюдается обильное потоотделение, озноб. - Боль в грудной клетке. Присоединяются в развёрнутых стадиях заболевания и при переходе туберкулёзного процесса на плевру. Поражения других органов сопровождаются признаками, которые на первый взгляд неотличимы от симптомов других распространенных недугов, поэтому в рамках данного материала рассматривать их не имеет смысла.
Вариант 7	Любая болезнь характеризуется воздействием бактерий или вирусов на организм. Признаки сальмонеллеза у взрослых начинают проявляться в 24 часа и отличаются по типу заболевания. Вот каковы симптомы сальмонеллеза у взрослого человека по формам: Гастроинтестинальный – недомогание, головокружение, головная боль, увеличение температуры. Боль в желудке, в пупочной области, рвота с остатками пищи. Диарея с зеленым калом и слизью, который пенится. Язык покрывается сухим белым налетом, живот вздувается, урчит. Печень с селезенкой увеличиваются в меру. Длятся симптомы пять дней, могут привести к регидратации и дисфункции метаболизма. При легкой форме наблюдается непродолжительная рвота, нестабильный стул, которые проходят через три дня. В усложненной форме лихорадит больного пять дней, рвота и стул многократные, болезнь напоминает дизентерию. Тифоподобный – лихорадка в течение недели, интоксикация, бред, галлюцинации. На животе видна сыпь, язык серо-коричневый, кожа бледная, живот вздут, внутренние органы увеличены. Проходит через 1,5 месяца. Септический – длительная лихорадка, обильное отделение пота. Озноб, кожные покровы желтеют. Возможно развитие гнойных процессов, способных привести человека к смерти.
Вариант 8	Родителям важно знать, какие симптомы при пневмонии у ребенка должны их настораживать, так как у детей признаки пневмонии могут иметь определенные особенности. Как проявляется пневмония у детей, зависит от особенностей болезни и от возраста ребенка. Детская пневмония может развиваться, если у ребенка отмечаться определенные симптомы: Можно заподозрить воспалительный процесс, если повышение температуры (более 38 градусов) продолжается дольше трех дней, при этом сбить ее обычными препаратами не удается. Должна обеспокоить и температура, не поднимающаяся выше 37,5 градусов, у маленьких детей. Особенно, если при этом отмечается также ряд признаков интоксикации – высокий уровень потливости, слабость, плохой аппетит. У новорожденного, а также у грудничков, может не отмечаться резких скачков температуры тела в процессе проявления воспалений, так как терморегуляция у них еще не совсем совершенна, а иммунная система пока остается незрелой. У больных детей дыхание очень частое, поверхностное. Груднички до 2 месяцев в минуту делают 60 вдохов в минуту дети до 1 года – 50, те, кому уже исполнился 1 год – 40. Как правило, при воспалении, малыш произвольно пытается лежать на одном боку. Также может быть отмечен еще один признак: раздев малыша, родители могут заметить, что в процессе дыхания с той стороны, где больное легкое, происходит втягивание кожи между ребрами и ее отставание во время дыхания. Иногда у малыша нарушается ритм дыхания, происходят его периодические остановки, меняется часта

	и глубина. Самые маленькие дети могут начать кивать в такт дыхания, раздувать щеки, вытягивать губы. Иногда из носа и рота появляются пенистые выделения. Самые маленькие дети, заболевшие пневмонией, плачут и капризничают, становятся вялыми. Они плохо спят, не желают кушать. Часто отмечается рвота и диарея, грудички срыгивают, отказываются брать грудь. У ребенка может развиваться не только стрептококковая, но и атипичные пневмонии. Какие симптомы при этом могут проявляться, зависит от возбудителя, особенностей протекания. Как правило, при болезни, спровоцированной хламидиями и микоплазмой, изначально болезнь развивается как простуда. Малыша беспокоит сухой кашель, першение в горле, насморк. Изначально возможен проявиться покашливание вследствие першения, позже кашель перерастает в мучительный, когда ребенок плачет или кушает. Важно учитывать, что при наличии ряда факторов (загрязненность воздуха, действие аллергенов или химических веществ) у малыша может развиться хроническая пневмония, симптомы которой проявляются периодически.
Вариант 9	Основными симптомами язвы являются стойкие боли. Пациент ощущает их долго, в зависимости от своей терпеливости - неделю, месяц, полгода. Язвенная боль чаще локализуется в подложечной области, на середине расстояния между пупком и концом грудины; при язве желудка - по средней линии или слева от нее; при язве двенадцатиперстной кишки - на 1-2 см вправо от средней линии. Язвенные боли могут быть разной интенсивности, что зависит как от терпеливости пациента, так и от глубины язвы. При прочих равных условиях боли при язве двенадцатиперстной кишки более сильные, чем при язве желудка. Чаще боли, по сравнению, например, с коликами, значительно слабее, интенсивность их небольшая или средняя, характер боли - ноющий. Для язвенных болей характерна связь с приемом пищи. При локализации язвы в желудке боли возникают после еды - тем скорее, чем «выше» язва (т. е. ближе к пищеводу); натощак боли успокаиваются. При язвах двенадцатиперстной кишки типичны так называемые голодные и ночные боли, которые, наоборот, уменьшаются или проходят сразу после еды, а через 2-3 часа - возобновляются вновь. Если боли мгновенно, как стенокардия от нитроглицерина, проходят, то скорее всего перед вами именно «язвенный» больной. Появлению или усилению язвенных болей предшествует прием острой пищи, непривычной пищи. Боли более интенсивны в плохую погоду, когда дует сильный ветер и дождь льет как из ведра. Часто «язвенные» больные чувствуют, что язва у них «открылась» и заболела после ссоры, скандала, неприятностей на работе, похорон и т. п. Интересно, что и чрезмерно сильные положительные эмоции тоже могут спровоцировать боль. Язвенные боли часто сопровождаются изжогой. Изжога натощак свидетельствует о высокой и неправильной секреции желудком соляной кислоты. В период обострения, примерно у 30-40% больных с язвой, наблюдаются рвота, причем содержимое рвотных масс кислое на вкус. Для язв, протекающих с чрезмерно высокой кислотностью, характерны запоры, нередко с кишечными коликами. Наконец, у язвенных больных часто наблюдается чувство внутренней напряженности и повышенная раздражительность.
Вариант 10	Симптомы аллергии зависят от типа аллергена, а точнее от места контакта аллергена с частью вашего тела. Так в зависимости от места (дыхательные пути, пазухи носа, кожа, пищеварительная система) могут проявляться различные симптомы. • Чихание (обычно сильно и часто). • Кашель, стеснение в груди, ощущение нехватки воздуха, затрудненное дыхание или одышка. • Зуд в носу и обильное выделение жидкого секрета из носа. • Зуд в глазах, слезотечение, покраснение глаз и отечность век. • Кожный зуд, покраснение кожи, высыпания на коже, шелушение кожи. • Покалывание во рту, покалывание или онемение языка. • Отек губ, языка, лица, шеи. • Тошнота, рвота, диарея. Большинство аллергических реакций имеют локальный характер (в месте контакта тела с аллергеном), например аллергические реакции на коже, в носу, в ротовой полости или пищеварительной системе. Когда имеет место анафилактический шок, аллергической реакции подвержен весь организм, реакция развивается спустя несколько минут после контакта с аллергеном. Симптомы анафилактического шока могут включать в себя все из представленных ниже или некоторые из них: Отек

	горла или полости рта. Тяжело глотать и/или говорить. Сыпь на любом участке тела. Покраснение и зуд кожи. Спазмы в животе, тошнота и рвота. Внезапное ощущение слабости. Резкое снижение артериального давления. Слабый и быстрый пульс. Головокружение и потеря сознания.
Вариант 11	В случае возникновения бронхита, кашель является главным симптомом. Важно понимать, что на самом деле кашель – это защитная функция организма. По сути, это усиленный выдох, с помощью которого организм пытается избавиться от патогенных агентов, попавших в дыхательные пути (в данном случае – вирусов, бактерий). Помимо этого взрослый человек ощущает общее недомогание, ухудшение аппетита, быструю утомляемость, жар. Все это проявления общей интоксикации организма, вызванной воспалением бронхов. Температура обычно достигает высоких значений − 38 -39^{0}С. Но иногда она может быть и ниже, это зависит от индивидуальной реактивности организма. При значительном повреждении дыхательных путей могут повреждаться мелкие сосуды легких, вследствие чего в мокроте могут быть и примеси крови. Период острой симптоматики при бронхите, как правило, длится 3-4 дня. Возможны также сильные боли за грудью. Особенно это характерно в период кашля. Зачастую больные жалуются на повышенную потливость. Когда появились первые симптомы, важно подумать о том, чем лечить бронхит, и какие препараты для этого использовать. При хроническом бронхите кашель с отхождением скудной мокроты, одышка при физической нагрузке могут быть постоянными симптомами, сопровождающими пациента на протяжении жизни. В этом случае про обострение бронхита говорят, если отмечается значительное усиление вышеуказанных симптомов: усиление кашля, увеличение объема отделяемой мокроты, усиление одышки, появление температуры и т.д. Бронхит у взрослых, особенно острый, достаточно редко протекает изолированно. Чаще всего он сочетается с явлениями ринита (насморка), трахеита. Это безусловно оказывает влияние на общую клиническую картину.
Вариант 12	Артроз височно-нижнечелюстного сустава Если пациента беспокоят стреляющие боли в ухе справа или слева (иногда с обеих сторон) по утрам - скорее всего, имеет место артроз височно-нижнечелюстного сустава – дегенеративное заболевание, которое поражает суставной хрящ. В отличие от острого отита, при котором боль в ухе продолжается в течение короткого времени, а затем проходит, артроз височно-нижнечелюстного сустава обычно имеет длительное, упорное течение. Кроме стреляющих болей в ушах по утрам, во время пробуждения, данное заболевание имеет следующие симптомы: боль в ушах и в самом суставе может беспокоить постоянно в течение дня, при этом она имеет больше ноющий характер, является умеренной; движения нижней челюсти становятся затрудненными; во время открывания и закрывания рта ощущается хруст в височной области; при длительном течении артроза нарушается нормальное смыкание челюстей, у пациента нарушает прикус, что иногда, в особенно тяжелых случаях, может приводить к расстройствам жевания и артикуляции. Артроз височно-нижнечелюстного сустава не является инфекционным или воспалительным заболеванием, поэтому признаки воспалительного процесса отсутствуют. У пациентов, страдающих данной патологией, никогда не бывает высокой температуры тела.
Вариант 13	В основе мочекаменной болезни лежит процесс формирования камней (конкрементов) в почках или других структурах мочевыводящего тракта (чаще всего речь идет о мочевом пузыре). Более подробно о видах камней в почках можно почитать в этой статье. Мочекаменная болезнь возникает под воздействием сразу нескольких факторов, как внешнего (неправильное питание, прием лекарств из разных групп и т.д.), так и внутреннего происхождения (например, пороки развития почек, сужение просвета уретры). Все вместе они становятся причиной обменных нарушений в организме больного. МКБ (приступ почечной колики) характеризуется следующими симптомами: • острый и нестерпимый приступ боли, который возникает на фоне закупорки просвета мочевыводящих путей крупным конкрементом; • нарушение процесса мочеиспускания (оно учащается и становится болезненным); • на пике болевого синдрома возможна сильная тошнота и приступы рвоты, которые не приносят облегчения; • повышение температуры тела, резкая слабость, недомогание;

	• изменение цвета мочевого осадка (появление в ней следов крови). Диагностика болезни заключается в рентгенологическом и ультразвуковом исследовании (конкременты хорошо визуализируются благодаря УЗИ, в том числе и «рентгеннегативные»). При необходимости проводят КТ или МРТ органов мочевыделительной системы.
Вариант 14	Артрит височно-нижнечелюстного сустава Это заболевание, которое чаще всего имеет воспалительную природу и симптомы, напоминающие артроз или острый отит. Для артрита височно-нижнечелюстного сустава характерны следующие проявления: боль в ухе на стороне поражения может иметь различную степень выраженности: от легкого дискомфорта до очень сильной, мучительной; часто отмечается снижение слуха, иногда вплоть до его полной утраты; очень характерна скованность в нижней челюсти по утрам: пациент практически совсем не может открыть рот; во время движений в нижней челюсти больной ощущает шум разного характера: щелканье, хруст, шуршание. Если развивается гнойный артрит, он может сопровождаться интенсивными болями в ухе, нарушением слуха и ощущением заложенности в ушах. В области височно-нижнечелюстного сустава отмечается покраснение кожи, припухлость. Повышается температура тела. При выраженном гнойном процессе может потребоваться хирургическое вмешательство.
Вариант 15	Мастоидит Позади ушной раковины на черепе расположен костный выступ, который называется сосцевидным отростком. У разных людей он может быть либо заполнен костным веществом, либо содержать внутри полости. Если в них попадут с током крови болезнетворные микроорганизмы, или произойдет травма, то может развиться воспалительный процесс в сосцевидном отростке – мастоидит. Главный симптом мастоидита – это пульсирующая боль в ухе и за ушной раковиной. Кроме того, зачастую присутствуют и другие симптомы: припухлость позади ушной раковины, может иметь место покраснение кожи; густые выделения из ушей; слабость, повышение температуры тела, лихорадка; снижение слуха, вплоть до его полной утраты. Для того, чтобы мастоидит, симптомом которого в данном случае является пульсирующая боль за ухом, не дал осложнений и не перешел в хроническую форму, требуется назначение правильного лечения. При появлении описанных выше признаков нужно немедленно обратиться к отоларингологу.
Вариант 16	Паротит (воспаление слюнной железы) Слюнная железа находится под кожей спереди от ушной раковины. Гнойный паротит вызывается стафилококками или стрептококками, и попадает в железу одним из трех путей: с током крови; с током лимфы; из больного зуба. Для заболевания характерна острая боль в области уха. Она сопровождается другими симптомами, такими как: повышение температуры тела, лихорадочное состояние, общая слабость и недомогание; боли в области уха может предшествовать головная боль, ощущение разбитости; болевые ощущения могут возникать или усиливаться в процессе глотания или жевания; кпереди от ушной раковины под кожей находится болезненная припухлость, ощупывание которой приводит к еще большему усилению боли; если осмотреть ротовую полость, то в месте отверстия протока слюнной железы можно увидеть покраснение и выделение капелек гноя.
Вариант 17	Если зубы не прорезываются вовсе, отсутствуют какие-либо из зубов, при этом остальной ряд смещен в сторону дефекта, то это первичная полная адентия. Если при удалении зуба, присутствующие зубы смещаются в сторону дефекта, то это вторичная адентия. Если потеряны все зубы и лицевой скелет изменился и нижняя половина лица западает, то это полная адентия.
Вариант 18	Лимфаденит В районе ушной раковины под кожей находятся околоушные лимфатические узлы. При их воспалении пациента могут беспокоить боли в ухе. Чаще лимфаденит развивается в результате проникновения в лимфатический узел инфекции из больного зуба, или из других очагов инфекции в организме, с током крови или лимфы. Помимо боли в ухе, прочие симптомы лимфаденита характерны для воспалительного заболевания: в области пораженного лимфатического узла под кожей находится болезненная припухлость, покраснение; у пациента повышается температура тела, может развиваться лихорадка; общая слабость, недомогание, разбитость, как при респираторной инфекции; иногда в результате отека и боли затруднено жевание, имеет место заложенность, шум в ушах, нарушение слуха.

Вариант 19	Синусит Это воспаление слизистой оболочки придаточных пазух носа – гайморовых (гайморит), лобных (фронтит), решетчатых (этмоидит). Для этих патологий характерны боли в области воспаленной пазухи (при фронтите – в области лба, при гайморите – в верхней челюсти). Но со временем боль теряет четкую привязку. Пациента могут беспокоить головные, зубные боли. Возможный симптом синусита – боль в ухе на стороне поражения. О том, что острая боль в ухе и повышение температуры тела являются признаками синусита, свидетельствуют следующие дополнительные симптомы: постоянная заложенность носа и гнусавый голос, которые не проходят после закапывания лекарства; несмотря на лечение, симптомы респираторной инфекции не только не исчезают, но даже усиливаются; першение в горле, кашель; нарушение обоняния; припухлость лица на стороне поражения.
Вариант 20	Для острого фарингита характерны першение, сухость, дискомфорт и боли в горле при глотании (особенно при пустом глотке), реже – общее недомогание, подъем температуры (обычно 37,5-38°С). При воспалении тубофарингеальных валиков боль обычно иррадиирует в уши. При пальпации может отмечаться болезненность и увеличение верхних шейных лимфоузлов. При фарингоскопии видны гиперемия задней стенки глотки и небных дужек, отдельные воспаленные лимфоидные гранулы, но при этом отсутствуют характерные для ангины признаки воспаления небных миндалин. Следует помнить, что острый фарингит может быть первым проявлением некоторых инфекционных болезней: кори, скарлатины, коревой краснухи. В ряде случаев требуется проведение дифференциальной диагностики с болезнью Кавасаки и синдромом Стивенса–Джонсона. Для хронического фарингита не характерны повышение температуры и существенное ухудшение общего состояния. Ощущения характеризуются больными как сухость, першение и ощущение комка в горле, что вызывает желание откашляться или «прочистить горло». Кашель обычно упорный, сухой и легко отличимый от кашля, сопровождающего течение трахеобронхита. Дискомфорт в горле часто связан с вынужденной необходимостью постоянно проглатывать находящуюся на задней стенке глотки слизь, что делает больных раздражительными, мешает их обычным занятиям и нарушает сон.
Вариант 21	Возможность определения по заданным признакам неисправностей топливной системы и электронной системы зажигания: • повышенный расход топлива; двигатель не развивает номинальной мощности; затрудненный пуск двигателя; неустойчивый холостой ход соответствует *неисправность* - засорение (деформация) сливного топливопровода. • повышенный расход топлива; запах бензина; подтеки топлива; двигатель не развивает номинальной мощности; затрудненный пуск двигателя; неустойчивый холостой ход соответствует *неисправность* - негерметичность системы. • двигатель не запускается или запускается с трудом; неустойчивая работа двигателя на холостом ходу соответствуют *неисправности* обрыв (пробой) высоковольтных проводов; неисправность свечей зажигания; неисправность катушки зажигания; неисправность входных датчиков (*датчика частоты вращения коленчатого вала, датчика холла*); неисправность электронного блока управления. • повышенный расход топлива; снижение мощности двигателя соответствуют *неисправности* - неисправность свечей зажигания; неисправность входных датчиков; неисправность электронного блока управления.
Вариант 22	Симптомы острой и хронической форм лейкемии различны. При хронической больной постоянно испытывает недомогания, которые можно спутать с симптомами других болезней. Стоит сразу отметить, что все эти недомогания не постоянны, а появляются и исчезают совершенно неожиданно. При хронической лейкемии работоспособность всегда понижена. Даже небольшой физический труд будет выполняться с большими усилиями. Хроническая лейкемия, симптомы которой мы рассматриваем, становится причиной нарушения сна. Речь идет не только о бессоннице, но и о постоянной тяге ко сну. Во втором случае человек не может выспаться независимо от того, сколько спит. При всем этом головной мозг настолько перегружается, что становится сложно воспринимать, а уж тем более запоминать какую-либо информацию. Температура тела может быть повышена. Повышение может быть незначительным, но дискомфорт от него все равно ощутим. Сбить температуру практически невозможно. Синие круги под глазами – еще один симптом лейкемии. Больные выглядят так, будто бы не спали несколько ночей подряд. Лицо становится очень бледным. Могут происходить носовые кровотечения. Симптомы

	лейкемии – это еще и мелкие пятнышки на теле. Они синего цвета и похожи на звездочки. При данном заболевании крайне плохо заживают различные повреждения кожи. Даже маленькая царапина будет беспокоить больного больше месяца. Иммунитет ослаблен, а значит, частые простудные заболевания просто неизбежны. Болячки сменяются одна другой.
Вариант 23	Возможность определения по заданным признакам неисправностей системы охлаждения: • перегрев двигателя соответствуют *неисправности* низкий уровень охлаждающей жидкости; ослабление привода водяного насоса; нарушение герметичности водяного насоса; неисправности привода вентилятора; неисправности термостата;засорение сердцевины радиатора; загрязнение наружной поверхности радиатора;засорение патрубков. • переохлаждения двигателя соответствуют *неисправности* - неисправность термостата; неисправность привода вентилятора; неисправность указателя температуры; неисправность датчика температуры. • наружная утечка охлаждающей жидкости соответствуют *неисправности* нарушение герметичности крепления патрубков; повреждение патрубков; нарушение герметичности центробежного насоса; нарушение герметичности радиатора; трещины в рубашке охлаждения; прогорание прокладки головки блока цилиндров. • внутренняя утечка охлаждающей жидкости соответствуют *неисправности* - трещины в рубашке охлаждения; прогорание прокладки головки блока цилиндров.
Вариант 24	Соматические симптомы невроза характеризуются проявлением болевых ощущений, таких как: Возникновение головных болей, причём характеризующихся продолжительностью и внезапностью появления. Болевые ощущения в области сердца и живота, мышцах и суставах, что является первопричиной недомогания. Также свойственно появление дрожания рук и частые мочеиспускания, необязательно подкреплённые заболеваниями почек и половых органов. Человеку свойственно быстро уставать, даже если он ничего и не делал. При этом усталость как физическая, так и умственная. Нет желания совершать какую-либо работу, возникает снижение работоспособности. Человек с симптомами невроза становится сонным и мрачным. Потемнение в глазах, дезориентация в местности, головокружения и даже обмороки — все это является симптомами заболевания. Человеку свойственно появление потливости, которая характеризуется частотой появления. Эта потливость возникает не от жаркой погоды, а от постоянной боязни, переживаний, нервозности. Особенно активно выделяется пот в ночное время, когда человек спит, а наутро обнаруживает влажную подушку. Психические расстройства влияют на снижение потенции и могут итоге развить такое заболевание, как простатит. Нарушается вестибулярный аппарат. Признаками этого нарушения являются частые головокружения, особенно при опрокидывании головы назад. Эти головокружения на начальных этапах проявляются редко, но с развитием недуга усиливаются и вызывают дискомфорт при выполнении физических работ. Нарушение режима питания. Психологический вид вызывает нарушение аппетита у человека, причём это может быть как недоедание, так и переедание. Переедание или чрезмерное потребление жирной пищи свидетельствует о том, что у человека булимический невроз. На фоне психических расстройств человек находит утешение в употреблении пищи, отчего возникает другая проблема — ожирение. Частое питание также не решает проблему невроза, поэтому потребуется проведение лечебных мероприятий. Возникновение бессонницы или постоянное желание спать. Во время сна возникают частые пробуждения, вызываемые кошмарами. Проблемы со здоровьем, которые отражаются на психике человека. Он переживает за своё здоровье, о том, что делать далее, как быть.
Вариант 25	Возможность определения по заданным признакам неисправностей механической коробки передач: • *Признаку:* шум в нейтральном положении соответствуют *неисправности* износ подшипника ведущего вала; низкий уровень масла в коробке. • *Признаку:* шум при включении передач соответствуют *неисправности* - износ или деформация блокирующего устройства; износ муфт синхронизаторов; ослабление резьбовых соединений крепление коробки передач; неполное выключение сцепления. • *Признаку:* шум при работе коробки соответствуют *неисправности* износ подшипников; износ муфт синхронизаторов; низкий уровень масла в коробке.