МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Кузбасский государственный технический университет имени Т. Ф. Горбачева»

Кафедра информационных и автоматизированных
производственных систем

Составитель
А. В. Протодьяконов

Знакомство и основы работы с Visual Prolog 7.3. Структура Пролог-программы.

Методические указания по выполнению лабораторных работ
по дисциплине «Системы искусственного интеллекта и принятие
решений» для студентов направления подготовки 09.03.02
«Информационные системы и технологии», профиль
«Информационные системы и технологии»,
очной формы обучения

Рекомендовано учебно-методической комиссией направления
09.03.02 «Информационные системы и технологии»
в качестве электронного издания для использования
в учебном процессе

КЕМЕРОВО 2017

Рецензенты:

И. В. Чичерин – доцент, кандидат технических наук, председатель учебно-методической комиссии направления 09.03.02 «Информационные системы и технологии»

А. Н. Трусов – доцент кафедры информационных и автоматизированных производственных систем

Протодьяконов Андрей Владимирович

Знакомство и основы работы с Visual Prolog 7.3. Структура Пролог-программы: методические указания к лабораторным работам по дисциплине «Системы искусственного интеллекта и принятие решений» [Электронный ресурс]: для студентов по направлению подготовки 09.03.02 «Информационные системы и технологии», профиль «Информационные системы и технологии», очной формы обучения / сост. А. В. Протодьяконов; КузГТУ. – Электрон. дан. – Кемерово, 2017. – Систем. требования: Pentium IV; ОЗУ 8 Мб; Windows 95; мышь. – Загл. с экрана

1. Цель работы

Цель – ознакомиться с интерфейсом, основными правилами работы и настройками среды Visual Prolog 7.3. Получить навыки создания проектов и запуска приложений.

2. Основы работы с Visual Prolog 7.3

2.1. Запуск Visual Prolog

Актуальную версию Visual Prolog можно скачать с официального сайта разработчика visual-prolog.com. Для учебных целей подойдет персональная версия программы Visual Prolog Personal Edition for the Microsoft Windows platform.

Установка Visual Prolog в целом не сильно отличается от установки других программа в Windows. Обычно программа устанавливается в директорию C:\Program Files (x86)\Visual Prolog 7.3 PE\ если у вас x64 версия Windows. Запустить программу можно при помощи файла в папке bin с названием Vip.exe. Если при установке указать создание ярлыка на рабочем столе, то он там создастся. Пролог также можно запустить через меню «Пуск». В нем Пролог имеет стандартное название, например Visual Prolog 7.3 PE.

2.2. Интерфейс программы Visual prolog

При запуске пролог открывает следующую форму:

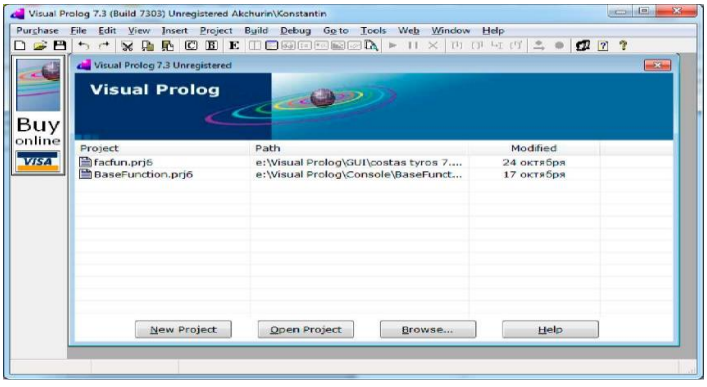


Рис. 1. Окно среды визуальной разработки

В центре может размещаться окно приглашения Visual Prolog, содержащее список ранее реализованных проектов и кнопки управления:

- New Project – новый проект.
- Open Project – открыть проект, выделенный в списке.
- Browse – поиск и выбор каталога для проекта.
- Help – справка. Открыть учебник, встроенный в ИСР.

Из этого окна можно выбрать проект для работы.

Чтобы создать приложение на VP сначала нужно запустить New Project. Откроется следующее окно:

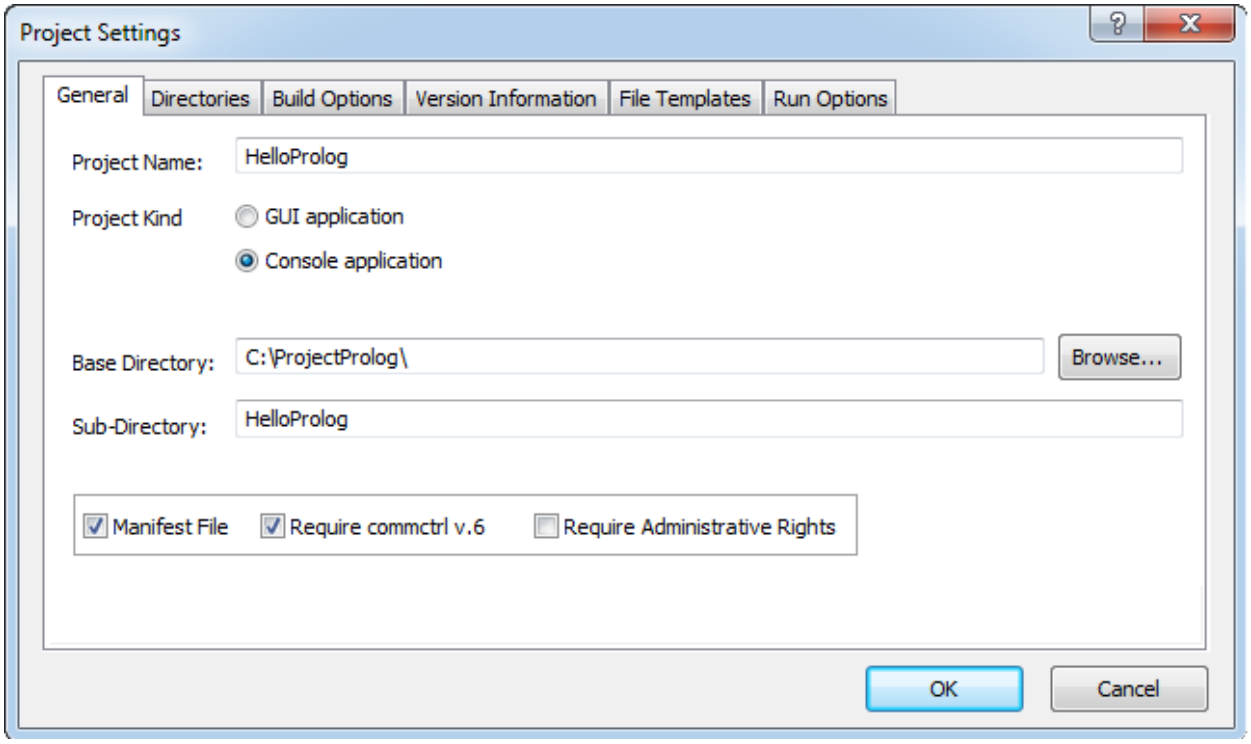


Рис. 2. Настройки проекта

Настройки создаваемого проекта довольно обширны. Для создания простейших приложений достаточно указать имя проекта в окне Project Name. Для корректной работы имя проекта не должно содержать пробелов. Далее нужно задать тип приложения в окне Project Kind, для данного курса это Console application, а также базовую директорию Base Directory в которой и расположен сам проект. После этого можно создать основу консольного приложения.

Для проекта строится дерево, отображаемое в окне. В левой панели окна показано дерево проекта, которое содержит системные шаблоны и встроенные библиотеки, а в правой – информационный браузер.

Стартовое дерево проекта включает системные компоненты, которые не надо редактировать (ИСР использует их при компиляции):

– Проект – ProjectName.prj6.
– Манифест для приложения – main.manifest.
– Пакет для приложения – main.pack.
– Подключаемые системные библиотеки – $(ProDir)\lib.

Программа создается какое-то время, после чего появится следующая форма:

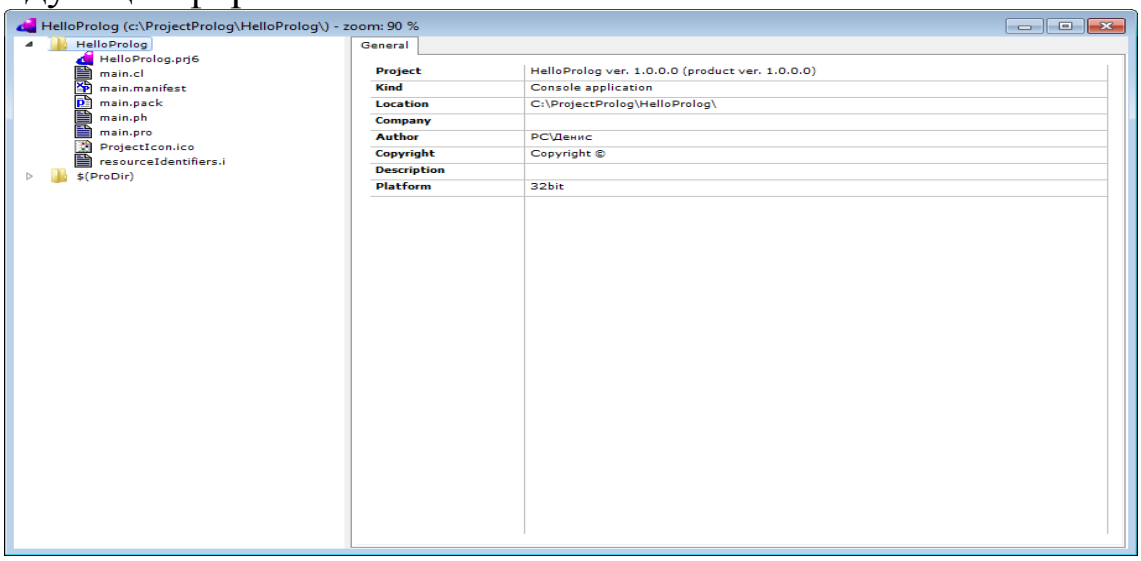


Рис. 3. Шаблон проекта

Теперь командой Build/Build осуществляется компиляция проекта, в дерево проекта добавляются файлы проекта без функциональности:

– Шаблон класса – main.cl.
– Шаблон проекта – main.ph.
– Шаблон реализации приложения – main.pro.

Для задания функциональности щелчком мыши выбираем main.pro, в шаблон которого нужно добавить свои операции.

В файле main.pro расположен код программы. Изначально в Прологе представлен следующий код (рис 4). В листинге шаблона программы отмечено место, куда надо добавить коды.

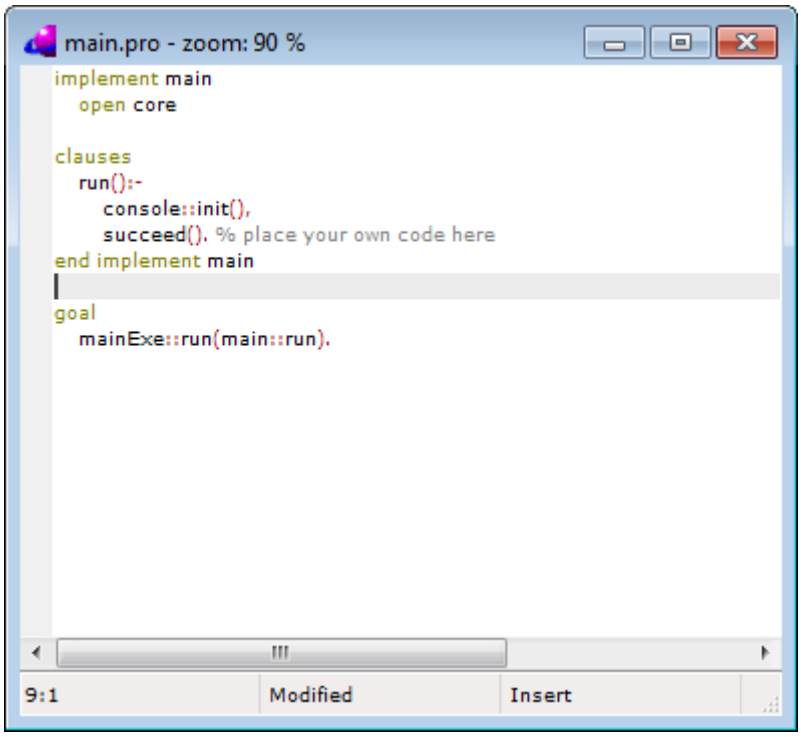


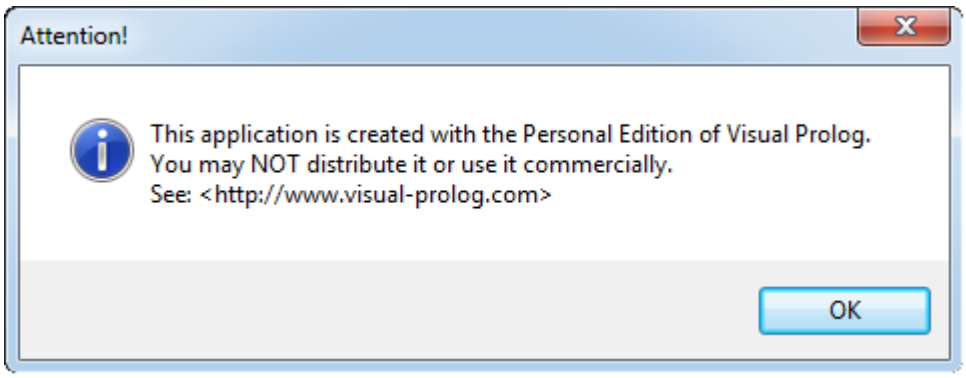
Рис. 4. Листинг шаблона программы

Так, например, для вывода текста "Привет из Visual Prolog " на экран и ожидания команды завершения работы, надо изменить содержание предиката run()

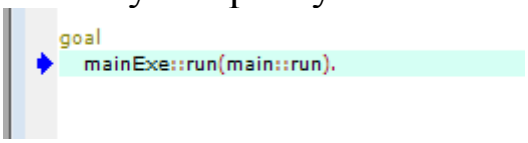
```
run():-
console::init(),
console::write("Привет из Visual Prolog "),
  =console::readChar().
```

Все элементы Пролога, кроме целей (goal), расположены в классе границы и определяются ключевыми словами “ *implement имя класса*” и “ *end implement имя класса*”. Ключевое слово “*open класс*” позволяет использовать предикаты из других классов. Данные в разделе “*clauses*” представляют собой описания фактов и правил, используемые в Прологе. Цель *mainExe::run (main::run)* запускает программу на выполнение предикатом run из класса main. Console::init() инициализирует окно консоли и закрывает приложение.

Для запуска компиляции программы нужно нажать на зеленую стрелочку  , после чего будет запущено окно с содержимым о компиляции:



Далее будет указана выполняемая цель. После чего нужно будет снова нажать на зеленую стрелку



и приложение будет запущено. Результат выполнения программы будет расположен вверху в отдельном окне.

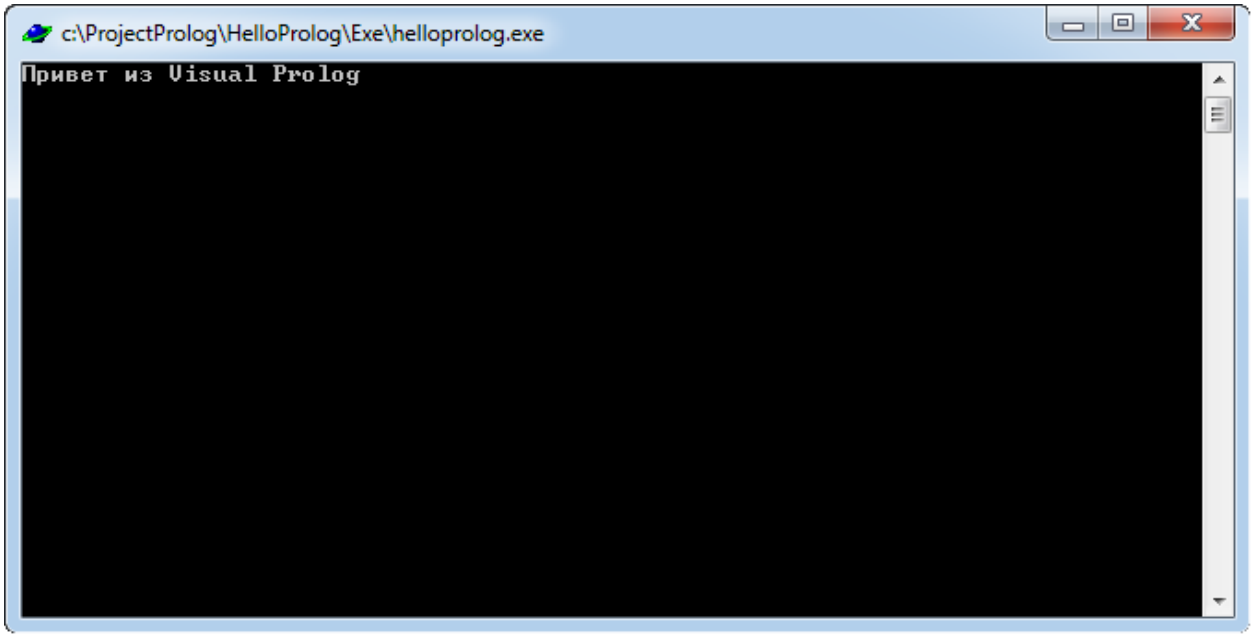


Рис. 5. Тестовая программа "Привет из Visual Prolog"

2.3. Обработка ошибок

Если вы допустили ошибки в программе и пытаетесь скомпилировать ее, то среда визуальной разработки отобразит окно **Errors (Warnings),** которое будет содержать список обнаруженных ошибок и предупреждений (рис. 6).

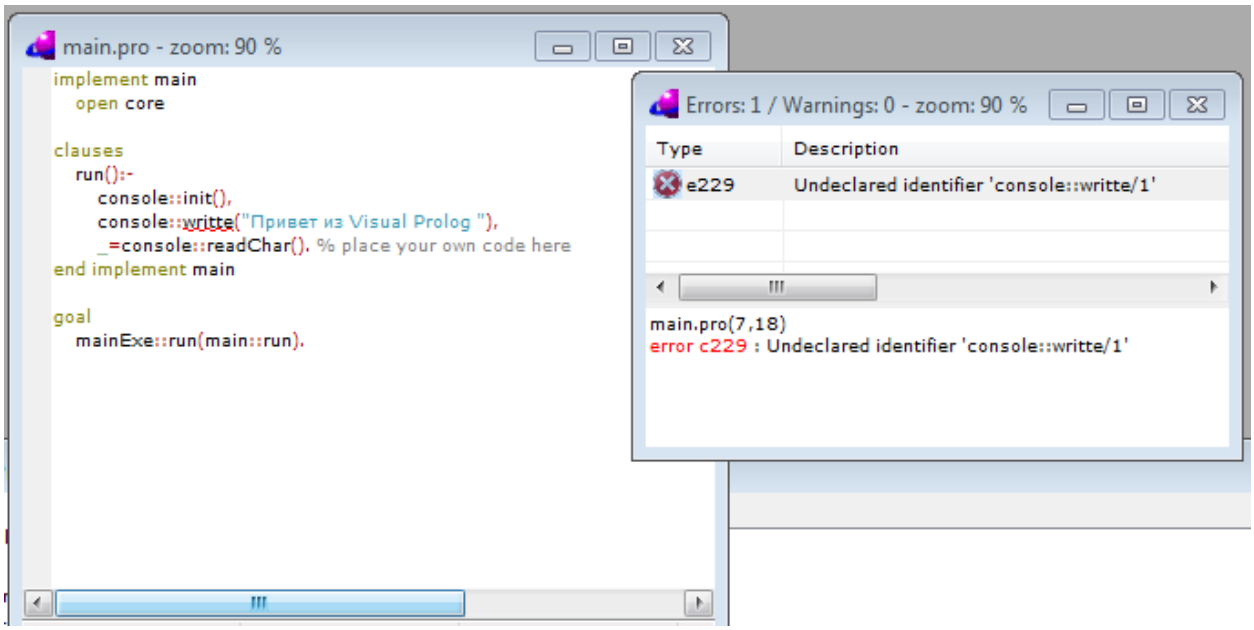


Рис. 6 Обработка ошибок

Дважды щелкнув на одной из этих ошибок, вы попадете на место ошибки в исходном тексте. Можно воспользоваться клавишей <F1> для вывода на экран интерактивной справочной системы Visual Prolog. Когда окно помощи откроется, щелкните по кнопке **Search,** наберите номер ошибки, и на экране появится соответствующее окно помощи с более полной информацией о ней.

3. Знакомство с основами языка Visual Prolog

ПРОЛОГ – язык логического программирования (ПРОграммирование в ЛОГике), используемый для представления и манипулирования знаниями в системах искусственного интеллекта (ИИ).

Программа на языке ПРОЛОГ – набор утверждений, составляющих базу фактов и базу правил, к которым допустимо

обращение с запросами, касающимися их содержимого. Запросы называются также *целевыми утверждения*.

Логическое программирование – программирование, основанное на использовании механизма доказательства теорем в логике, который позволяет выяснить, является ли противоречивым некоторое множество логических формул. При этом программа рассматривается как набор логических формул, описывающих предметную область, совместно с теоремой, которая должна быть доказана. Логическое программирование избавляет программиста от необходимости определения точной последовательности шагов выполнения вычислений (алгоритма).

3.1. Структура Пролог-программы

Программа на Visual Prolog, представляемая кодом, разделяется ключевыми словами на секции разного вида путем использования ключевых слов, предписывающих компилятору, какой код генерировать. К примеру, есть ключевые слова, обозначающие различие между декларациями и определениями предикатов и доменов. Обычно, каждой секции предшествует ключевое слово. Ключевых слов, обозначающих окончание секции, нет. Наличие другого ключевого слова обозначает окончание предыдущей секции и начало другой.

Исключением из этого правила являются ключевые слова ***implement*** и ***end implement***. Код, содержащийся между этими ключевыми словами, есть код, который относится к конкретному классу. Те, кто не понимает концепцию класса, может пока (в рамках этого руководства) представлять себе его как модуль или раздел какой-то программы более высокого уровня.

Ниже представлена только часть ключевых слов, синтаксис которых легко может быть изучен по документации.

implement и ***end implement***

Среди всех ключевых слов, обсуждаемых здесь, это единственные ключевые слова, используемые парно. Visual Prolog рассматривает код, помещенный между этими ключевыми словами, как код, принадлежащий одному классу. За ключевым словом implement обязательно ДОЛЖНО следовать имя класса.

open

Это ключевое слово используется для расширения области видимости класса. Оно должно быть помещено вначале кода класса, сразу после ключевого слова implement (с именем класса и, возможно, именем интерфейса).

constants

Это ключевое слово используется для обозначения секции кода, которая определяет неоднократно используемые значения, применяемые в коде. Например, если строковый литерал "PDC Prolog" предполагается использовать в различных местах кода, тогда можно единожды определить мнемоническое краткое, легко запоминаемое слово для использования в таких местах:

```
constants
        pdc="PDC Prolog".
```

Заметьте, что определение константы завершается точкой (.). В отличие от переменных Пролога константы должны начинаться со строчной буквы (нижний регистр).

domains

Это ключевое слово используется для обозначения секции, объявляющей домены, которые будут использованы в коде. Синтаксис таких объявлений позволяет порождать множество вариантов объявлений доменов, используемых в тексте программы. Вообще говоря, объявляется функтор, который будет использоваться в качестве домена и ряд доменов, которые используются в качестве его аргументов.

class facts

Это ключевое слово представляет раздел, в котором объявляются факты, используемые в тексте программы. Каждый факт объявляется как имя, используемое для обозначения факта, и набор аргументов, каждый из которых должен соответствовать либо стандартному (предопределенному), либо объявленному домену.

class predicates

Эта раздел содержит объявления предикатов, которые определяются в тексте программы в разделе clauses. И опять,

объявление предиката - это имя, которое присваивается предикату, и набор аргументов, каждый из которых должен соответствовать либо стандартному (предопределенному), либо объявленному домену.

clauses

Среди всех разделов, существующих в тексте программ на Visual Prolog, это единственный раздел, который близко совпадает с традиционными программами на Прологе. Он содержит конкретные определения объявленных в разделе class predicates предикатов, причем синтаксически им соответствующие.

goal

Этот раздел определяет главную точку входа в программу на языке системы Visual Prolog. Более детальное описание приводится ниже.

Goal (цель)

В традиционном Прологе, как только какой-либо предикат определен в тексте, Прологу-машине может быть предписано выполнение программы, начиная с этого предиката. В Visual Prolog это не так. Будучи компилятором, он отвечает за генерацию эффективного исполняемого кода написанной программы. Этот код исполняется не в то же время, когда компилятор порождает код. Поэтому компилятору надо знать заранее точно, с какого предиката начнется исполнение программы. Благодаря этому, когда программа вызывается на исполнение, она начинает работу с правильной начальной точки. Как можно уже догадаться, откомпилированная программа может работать независимо от компилятора Visual Prolog и без участия среды IDE.

Для того чтобы это стало возможным, введен специальный раздел, обозначенный ключевым словом Goal. Его можно представлять как особый предикат, не имеющий аргументов. Это предикат - именно тот предикат, с которого вся программа начинает исполняться.

Мы можем задавать Прологу такие же вопросы, которые мы могли бы задать вам об этих отношениях. Основываясь на известных, заданных ранее фактах и правилах, вы можете

ответить на вопросы об этих отношениях, в точности так же это может сделать Пролог.

Например, естественном языке мы спрашиваем: "Маша любит цветы?". На Прологе придется указать в блок ***class facts*** описание факта.

любит:(symbol, symbol, symbol) nondeterm.

Nondeterm здесь указывает на неопределенность фактов, например, Маша может любить не только цветы, но и конфеты.

В clauses добавляются значения фактов, например.

```
любит("Маша", "Цветы").
любит("Маша", "Конфеты").
любит("Маша", "Мороженое").
любит("Маша", "Платья").
```

Чтобы задать запрос Prology необходимо определить предикат запроса в class predicates

```
вопросЛюбит:(symbol,symbol) procedure(i,i).
```

procedure(i,i) – означает что предикат имеет два входных значения.

Далее ввести в clauses описание данного предиката, а также заглушку от некорректных значений при вводе.

```
ВопросЛюбит(Имя,Предмет):-
  любит(Имя,ЧтоЛюбит),
  ЧтоЛюбит=Предмет,
  console::write("Да"),
  !.

вопросЛюбит(_,_):-!.
```

В теле запроса мы используем переменные Имя и Предмет, в прологе переменные всегда начинаются с большой буквы. При обращении к факту любит(Имя,ЧтоЛюбит), мы получаем значения которых любит человек с заданным именем в переменную ЧтоЛюбит. Конструкция ЧтоЛюбит=Предмет сравнивает вкусы Маши с нашим вопросом, если они хоть один

из них совпадет с нашим программа предикат выведет Да и закончится, иначе предикат ничего не выведет.

Изменим предикат run, чтобы он обращался к данному предикату.

```
run():-
console::init(),
вопросЛюбит("Маша","Цветы"),
_=console::readChar().
```

При запуске это приложения Visual Prolog выведет ответ на запрос

«да».

Для написания комментариев к программе в Пролог используется знак “%” для строчного комментария и комбинация знаков “/**текст комментария**/” при многострочном комментарии.

4. Порядок выполнения работы

1. Изучить интерфейс среды визуальной разработки Visual Prolog.
2. Написать и запустить программу выводящую текст в консоль.
3. Написать и запустить программу про спорт. В окне нужно напечатать следующий факты:

```
играет ("Вася",футбол).
играет ("Петя",футбол).
играет ("Маша",баскетбол).
играет ("Ваня",дартс).
И задать к ним запрос во что играет Петя.
```

4. Составить программу “Телефонный справочник”, в которой будут содержаться данные о владельце (фамилия или имя) и номер телефона (если он есть).

Составить простые запросы, с помощью которых выяснить:

- номер телефона по фамилии (имени);
- владельца имеющего определенный номер телефона.

Составить составные запросы, с помощью которых выясним:

– фамилии тех, кто имеет телефон и их номер;
– фамилии тех, у кого номера одинаковые номера телефонов;
– фамилии тех, кто имеет два номера телефона.

5. Составить программу “Организация турнира по теннису”, которая содержит сведения об именах и возрасте игроков. Каждая пара игроков одного возраста должна провести между собой две игры. Задача программы – составить список игр турнира, используя для этого составные запросы.

5. Контрольные вопросы

1. Версии Visual Prolog.
2. Что такое проект?
3. Построение проекта.
4. Исполнение проекта.
5. Консольное приложение.

Литература

1. Visual Prolog 7.3 для начинающих. Встроенный в среду учебник
2. Коста Э. Visual Prolog 7.1 для начинающих / перевод с англ: И. Алексеев, Е.А. Ефимова. – Prolog Development Center, 2008.
3. http://wikiru.visual-prolog.com/
4. Costas-Tyros_rus.pdf
5. Цуканова Н. И. Логическое программирование на языке Visual Prolog: учеб. пособие для вузов / Н. И. Цуканова, Т. А. Дмитриева. – М.: Горячая линия – Телеком, 2008. – 144с.