Министерство высшего образования и науки Российской Федерации

#### ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

##### Кафедра «Математическое обеспечение и применение ЭВМ»

Пояснительная записка

к курсовому проектированию по дисциплине

«Логическое и функциональное программирование»

на тему: «Тестовый вариант экспертной системы»

|  |  |
| --- | --- |
| Автор работы: | Угроватов Д.В. |
| Направление бакалавриата | 09.03.04 («Программная инженерия») |
| Обозначение курсовой работы | ПГУ 09.03.04 – 05КР161.25 ПЗ |
| Группа | 16ВП1 |
| Руководитель работы | доцент кафедры «МО и ПЭВМ», к.т.н. Михалев А.Г. |
| Работа защищена «\_\_» \_\_\_\_\_\_ 2018 г. | Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Пенза 2018 г.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Реферат Пояснительная записка содержит 34 листа, 6 рисунков, 1 таблица, 5 использованных источников, 2 приложения.  ЛОГИЧЕСКОЕ ПРОГРАММИРОВАНИЕ, ЭКСПЕРТНАЯ СИСТЕМА, PROLOG, ИНТЕЛЛЕКТУАЛЬНЫЕ СИСТЕМЫ  Целью курсовой работы является разработка программы, выполняющей распознавание животных по их описанию и обзор интеллектуальных систем, разработанных на языке Prolog.  Разработка проводилась на языке программирования Prolog.  Разработка проведена с использованием операционной системы Windows 10. | | | | | | | | | | |
|  |  |  |  |  | ПГУ 09.03.04 - 05КП161.25 ПЗ | | | | | |
|  |  |  |  |  |
| Изм. | Лист | № докум. | Подп. | Дата |
| Разраб. | | Угроватов Д.В. |  |  | «Тестовый вариант экспертной системы»  Пояснительная записка. | Лист. | | | Лист | Листов |
| Пров. | | Михалев А.Г. |  |  |  |  |  | 3 | 34 |
|  | |  |  |  | Группа 16ВП1 | | | | |
| Н. контр. | |  |  |  |
| Утв. | |  |  |  |

Содержание

[Реферат 3](#_Toc533458465)

[Введение 6](#_Toc533458466)

[1. Теоретическая часть 8](#_Toc533458467)

[1.1. Общие сведения о языке Prolog 8](#_Toc533458468)

[1.2. Области применения языка Prolog 10](#_Toc533458469)

[1.3. Применение экспертных систем 12](#_Toc533458470)

[2. Практическая часть 14](#_Toc533458471)

[2.1. Постановка задачи 14](#_Toc533458472)

[2.2. Описание применения программы 15](#_Toc533458473)

[2.2.1. Общие сведения и функциональное назначение 15](#_Toc533458474)

[2.2.2. Проектирование программы 16](#_Toc533458475)

[2.2.3. Схема логической программы 18](#_Toc533458476)

[2.2.4. Входные данные 19](#_Toc533458477)

[2.2.5. Выходные данные 20](#_Toc533458478)

[2.2.6. Используемые технические средства 21](#_Toc533458479)

[2.2.7. Тестирование 22](#_Toc533458480)

[Заключение 24](#_Toc533458481)

[Список используемой литературы 25](#_Toc533458482)

[ПРИЛОЖЕНИЕ A 26](#_Toc533458483)

[ПРИЛОЖЕНИЕ B 32](#_Toc533458484)

# Введение

В начале 80-х годов в исследованиях по искусственному интеллекту сформировалось самостоятельное направление, получившее название "экспертные системы" (ЭС). Основным назначением ЭС является разработка программных средств, которые при решении задач, трудных для человека, получают результаты, не уступающие по качеству и эффективности решения, решениям получаемым человеком-экспертом. ЭС используются для решения так называемых неформализованных задач, общим для которых является то, что:

* задачи не могут быть заданы в числовой форме;
* цели нельзя выразить в терминах точно определенной целевой функции;
* не существует алгоритмического решения задачи;
* если алгоритмическое решение есть, то его нельзя использовать из-за ограниченности ресурсов (время, память).

Кроме того, неформализованные задачи обладают ошибочностью, неполнотой, неоднозначностью и противоречивостью как исходных данных, так и знаний о решаемой задаче. [1]

Экспертная система - это программное средство, использующее экспертные знания для обеспечения высокоэффективного решения неформализованных задач в узкой предметной области. Основу ЭС составляет база знаний (БЗ) о предметной области, которая накапливается в процессе построения и эксплуатации ЭС. Накопление и организация знаний - важнейшее свойство всех ЭС.

В данной работе будет рассмотрен простой вариант экспертной системы, разработанной на языке Prolog, которая имеет заранее заданную базу знаний, но не способна обучаться, которая будет задавать пользователю вопросы о животных и на основе ответов выдавать результат.

В теоретической части данной работы будут рассмотрены интеллектуальные системы, разработанные на языке Prolog.

# Теоретическая часть

## Общие сведения о языке Prolog

Интерес к Prolog поднимался и затихал несколько раз, энтузиазм сменялся жёстким неприятием. Наиболее высокий интерес к языку Prolog, как к языку будущего, возник в 1980-х годах во время разработки японской национальной программы «компьютеры пятого поколения». В рамках этого проекта разработчики надеялись, что с помощью Prolog можно будет сформулировать новые принципы, которые приведут к созданию компьютеров более высокого уровня интеллекта.

Эта надежда базировалось на том, что логическое программирование использует не последовательность абстракций и преобразований, которая отталкивается от машинной архитектуры фон Неймана и присущего ей набора операций, а на основе абстрактной модели, которая никак не связана с каким-либо типом машинной модели. Из этого подхода следовало, что не человека надо обучать мышлению в терминах операций компьютера, а компьютер должен выполнять инструкции, свойственные человеку. И на этом историческом этапе некоторые ученые и инженеры считали подобный путь простым и эффективным.

Prolog – язык программирования, который основан не на алгоритме, а на логике предикатов. Если программа на алгоритмическом (процедурном) языке является последовательностью инструкций, выполняющихся в заданном порядке, то Prolog, будучи декларативным языком, содержит только описание задачи, а Prolog-машина выполняет поиск решения, руководствуясь только этим описанием и используя механизм поиска с возвратом и унификацию.

Разработка языка Prolog началась в 1970 г. Аланом Кулмероэ (Alain Colmerauer) и Филиппом Русселом (Philippe Roussel) в университете города Марсель, Франция. Они хотели создать язык, который мог бы делать логические заключения на основе заданного текста. Их работа была частично мотивирована желанием примирить использование логики в качестве декларативного языка представления знаний с процессуальным подходом к представлению знаний, который был популярен в Северной Америке в конце 1960-х и начале 1970-х годов.

Название Prolog было выбрано Филиппом Русселом как аббревиатура от французского PROgrammation en LOGique (PROLOG) и первая реализация этого языка с использованием компилятора Николауса Вирта "AlgolW" была закончена в 1972 году. Основы современного языка были заложены в 1973 году. Prolog постепенно распространялся среди тех, кто занимался логическим программированием, в основном благодаря личным контактам, а не через коммерциализацию продукта.

Базовым принципом языка является равнозначность представления программы и данных (декларативность), отчего утверждения языка одновременно являются и записями, подобными записям в базе данных, и правилами, несущими в себе способы их обработки. Сочетание этих качеств приводит к тому, что по мере работы Prolog-системы знания накапливаются. Накапливаются как данные, так и правила. Поэтому Prolog-системы считают естественной средой для накопления базы знаний. База знаний — важный компонент интеллектуальной системы. Они предназначены для поиска способов решения проблем из некоторой предметной области, основываясь на записях базы знаний и на пользовательском описании ситуации. [2]

## Области применения языка Prolog

Говоря об областях применения Prolog следует сразу оговориться, что он слабо приспособлен для решения задач, связанных с обработкой графики, вычислениями или численными методами. Вместе с тем он с успехом может использоваться в компьютерной алгебре, которая в отличие от численных методов, занимается реализацией аналитических методов решения математических задач на компьютере и предполагает, что исходные данные, как и результаты решения, сформулированы в аналитическом виде. В качестве основных областей применения Prolog можно отметить следующие направления:

• разработка быстрых прототипов прикладных программ;

• управление производственными процессами;

• создание динамических реляционных баз данных;

• перевод с одного языка на другой;

• создание естественно-языковых интерфейсов;

• реализация экспертных систем и оболочек экспертных систем;

• создание пакетов символьных вычислений;

• доказательства теорем и интеллектуальные системы, в которых возможности языка Prolog по обеспечению дедуктивного вывода применяются для проверки различных теорий.

Prolog нашел применение и в ряде других областей, например, при решении задач составления сложных расписаний. Он используется в различных системах, но обычно не в качестве основного языка, а в качестве языка для разработки некоторой части системы. Достаточно часто Prolog используют для написания функций взаимодействия с базами данных.

Используют его и в сложных поисковых системах, которые выполняют не только поиск, но и играют роль некоторой "отвечающей системы" – программного комплекса, который умеет извлекать информацию из большой выборки текстовых файлов и баз данных, а затем вести диалог с пользователем, отвечая, в обычном понимании этого слова, на его вопросы.

Также Prolog используют при написании новых специфичных языков программирования. Например, функциональный язык Erland построен на основе Prolog. По сути, Erland является усовершенствованием Prolog для некоторых специфических целей, связанных с задачами реального времени.

В данное время Prolog, несмотря на неоднократные пессимистические прогнозы, продолжает развиваться в разных странах, включая в себя новые технологии и концепции. К ним относятся и парадигмы императивного программирования, при котором процесс вычисления описывают в виде инструкций, изменяющих состояние программы.

Одно из направлений развития языка, в том числе и в России, реализует концепцию интеллектуальных агентов. Под этим термином понимаются разумные сущности, наблюдающие за окружающей средой и действующие в ней. При этом их поведение рационально в том смысле, что они способны к пониманию, а их действия направлены на достижение какой-либо цели. Такой агент может быть как роботом, так и встроенной программой. Об интеллектуальности агента можно говорить, если он взаимодействует с окружающей средой примерно так же, как действовал бы человек. [2]

## Применение экспертных систем

В качестве примера, иллюстрирующего области использования Prolog рассмотрим следующие проекты:

* PDC SCORE и другие – это ИТ-решения в области авиации, которые предназначены для планирования и составления расписаний для бизнесавиации, а также для бизнес-планирования работы авиакомпаний, аэропортов и наземного обслуживания самолетов. Около 20% мировых авиаперевозок координируется с использованием PDC SCORE. Более 40 международных авиакомпаний и 280 аэропортов используют эти решения.
* ARGOS – информационная система управления и принятия решений в кризисных ситуациях.
* PDC StaffPlan – это ИТ-решение для экономически эффективного кадрового планирования и управления ресурсами. Используется уже около 20 лет в крупных и средних организациях и компаниях в сфере здравоохранения, розничной торговли, аэропортах и промышленности.
* Dictus – это программа распознавания и синтеза речи, позволяющая превращать речь в печатный текст с 98% точностью, управлять компьютером с помощью голосовых команд, использовать голосовую передачу и прием SMS-сообщений и т.п.

В этих проектах используется разработанная в датской компании PDC технология, которая позволяет применять правила и методы искусственного интеллекта для оптимальной и эффективной поддержки принятия решений. Основу этой технологии составляет Visual Prolog.

Примеры проектов, использующих ЭС, разработанных на других языках:

* OpenCyc – мощная динамическая ЭС с глобальной онтологической моделью и поддержкой независимых контекстов.
* WolframAlpha – база знаний и набор вычислительных алгоритмов, вопросно-ответная система.
* MYCIN – наиболее известная диагностическая система, которая предназначена для диагностики и наблюдения за состоянием больного при менингите и бактериальных инфекциях.
* Акинатор – интернет-игра, в которой игрок загадывает любого персонажа, а Акинатор должен его отгадать, задавая вопросы. База знаний автоматически пополняется, поэтому программы может отгадать практически любого известного персонажа. [3]

Если говорить о менее мощных реализациях Prolog, поддерживающих в основном эдинбургский стандарт ISO Prolog, то, вне всякого сомнения, программа на языке Prolog может стать "мозгом" различных чрезвычайно интересных приложений. Но при этом выполнение интерфейсных функций (ввод-вывод, преобразование форматов данных, диалог с пользователем и т.д.) целесообразно возложить на другие языки программирования. [4]

. В настоящее время подобная практика разработки прикладных программных комплексов на основе Prolog находит широкое распространение.

# Практическая часть

## Постановка задачи

В рамках данной работы необходимо написать программу на языке prolog, осуществляющую распознавание животных по их описанию.

Программа должна включать базу знаний. База знаний пополняется, когда программа выключена.

Интерфейс программы должен представлять собой консоль, взаимодействующую с пользователем в формате «запрос-ответ». При запуске, программа должна отображать возможные варианты ответа для пользователя и задавать вопрос. Пользователь должен ввести один из трех вариантов ответа. После ввода программа должна задавать следующий вопрос, пока не сможет дать ответ, что это за животное.

При получении выражения, включающего в себя посторонние символы, программа должна заново задавать вопрос.

## Описание применения программы

## Общие сведения и функциональное назначение

Разрабатываемая программа ориентирована на прямое взаимодействие с пользователем через консольный интерфейс. Программа не предусматривает специальных интерфейсов для включения в иные программные системы, но, тем не менее, не исключает возможность подобного применения.

Рассмотрим более подробно действия, которые должна выполнять программа:

1. При запуске программа должна отобразить в консоли сообщение с возможными вариантами ответа и задать первый вопрос.
2. Программа должна ожидать ввода ответа пользователем.
3. После ввода ответа, программа вывести следующий вопрос. Эти два действия должны повторяться, пока программа не сможет найти результат.
4. Программа должна вывести полученный результат и предложить пользователю завершить работу или предложить найти другой результат с теми же данными и, если потребуется, задать дополнительные вопросы.
5. При неудачном поиске программа должна вывести соответствующее сообщение.

## Проектирование программы

В программе можно выделить две главные части – заполнение базы-знаний и сопоставление результата с ней.

Для заполнения базы-знаний воспользуемся энциклопедиями о животных. Из них возьмем следующие признаки животных: способность плавать, наличие окраса, является ли вид этого животного прирученным, способность летать, наличие копыт, наличие больших когтей и зубов и принадлежность животного к одному из классов и отрядов.

Сопоставление происходит следующим образом: пользователь отвечает на заданный ему вопрос об одном из признаков, и программа сопоставляет этот ответ с содержимым базы-знаний. Если она находит совпадение, то задает вопрос о следующем признаке предполагаемого животного. После нахождения достаточного количества соответствий программа выводит предполагаемый результат и предлагает пользователю либо вывести еще один результат, либо завершить работу.

Пользователю доступны следующие варианты ответа:

* «Да»
* «Нет»
* «Не знаю»

При ответах «Да» и «Нет» программа уменьшает возможное количество конечных результатов. Если пользователь ответил «Не знаю», программа переходит к следующему вопросу, а ответ на предыдущий оставляет не закрытым и при выведет результат без учета данного признака.

Когда программа выводит результат, пользователюю доступны следующие варианты ответа:

* «Да»
* «Нет»

Если пользователь отвечает «Да», программа завершает работу. При ответе «Нет» программа выводит следующий результат. Если соответствий больше нет, программа выводит соответствующее сообщение.

## Схема логической программы

Для описания животного используем предикат Animal. В данном предикате опишем какие признаки имеются у соответствующего животного. Также опишем, что данному животному не свойственно. По этим признакам программа будет опрашивать пользователя.

Для описания используем предикаты Positive и Negative. В данных предикатах будет производиться опрашивание пользователя, и в зависимости от ответа программа либо перейдет к следующему признаку данного животного, либо перейдет к следующему предикату с описанием животного. При этом программа запоминает ответ пользователя на наличие признака в базе данных.

В предикатах Positive и Negative используем предикат Ask. В нем программа задает вопрос и считывает ответ пользователя.

Введем три базы данных для положительных ответов, отрицательных и ответа «Не знаю»: Yes, No и Maybe соответственно. В соответствии с ответом в одну из баз данных заносится признак.

Для очистки баз данных введем предикат Delete\_all.

Для проверки присутствия признаков в базах данных введем предикаты Xpositive и Xnegative.

В целях облегчения использования необходимо составить предикат, вызывающий остальные предикаты. Назовем этот предикат Run.

В соответствии с данными рассуждениями была реализована программа. Прокомментированный исходный код программы представлен в приложении A.

## Входные данные

При запуске программа вызывает предикат «Run», не имеющий параметров. В нем она ознакамливает пользователя с возможными вариантами ответа и начинает задавать вопросы.

После вывода вопроса программа ожидает, что пользователь введет в консоль ответ.

Пользователь может ввести один из вариантов ответа: 'y' (вариант «Да»), ‘n’ (вариант «Нет»), ‘q’ (вариант «Не знаю»).

Программа корректно обрабатывает нарушение данных условий – повторно задает вопрос и не сохраняет прошлый ответ.

## Выходные данные

При успешном завершении поиска соответствий программа выводит пояснение: «Это животное:», после чего на той же строке отображает животное, соответствующее выбранным признакам.

Если такое животное не было найдено, то программа сообщит об этом: «Это животное с другой планеты».

После этого программа предлагает пользователю выбор: завершить работу или попробовать найти еще соответствия. Если потребуется, то программа задаст дополнительные вопросы и выведет предполагаемый результат, затем снова предоставит выбор завершить работу или искать соответствия дальше.

## Используемые технические средства

При разработке программы была использована среда Turbo Prolog.

Программа разработана с использованием операционной системы Windows 10 и эмулятора MS DOS.

Программа поставляется в виде исходного кода.

Для использования программы необходимо наличие установленной среды Turbo Prolog и эмулятор MS DOS или ОС MS DOS.

## Тестирование

Приведем тестирование программы в табличной форме в таблице 1. Скриншоты программы, подтверждающие результаты тестов приведены в приложении B. Номер скриншота соответствует номеру теста.

Таблица 1 – Тестирование программы

|  |  |  |  |
| --- | --- | --- | --- |
| № | Вопрос/Ответ | Полученный результат | Результат |
| 1 | Может плавать? / Да  Имеет окрас? / Да  Приручено людьми? / Да  Может летать? / Да  Откладывает яйца? / Да | Гусь | Успешно |
| 2 | Может плавать? / Нет  Имеет окрас? / Да  Приручено людьми? / Нет  Может летать? / Да  Откладывает яйца? / Да | Стриж | Успешно |
| 3 | Может плавать? / Нет  Имеет окрас? / Нет  Может летать? / Нет  Имеет большие когти и зубы? / Нет  Приручено людьми? / Нет | Животное с другой планеты | Успешно |
| 4 | Может плавать? / Нет  Имеет окрас? / Нет  Может летать? / Нет  Имеет большие когти и зубы? / Да  Приручено людьми? / Нет  Слепое? / Да  Живорожденное? / Да | Крот | Успешно |
| 5 | Может плавать? / Да  Имеет окрас? / Да  Приручено людьми? / Нет  Может летать? / Да  Откладывает яйца? / Да | Пингвин | Успешно |
| 6 | Может плавать? / Да  Имеет окрас? / Нет  Может летать? / Да  Имеет большие когти и зубы? / Нет  Приручено людьми? / Да  Живорожденное? / Да  Имеет копыта? / Да | Слон | Успешно |

# Заключение

В рамках теоретической части данной работы были рассмотрены интеллектуальные системы, разработанные на языке Prolog, приведены его общие сведения и описаны области применения.

В рамках практической части на языке Turbo Prolog была реализована программа, выполняющая распознавание животного по его описанию. Был описан процесс анализа задачи, проектирования и реализации программы. Приведен протокол тестирования и подтверждающие его скриншоты.

# Список используемой литературы

1. Введение в экспертные системы. Основные понятия и определения [Электронный ресурс]. – Режим доступа: www.habarov.spb.ru/new\_us/exp\_sys/es01[/](http://www.swi-prolog.org/)es1.htm – Заглавие с экрана. – (Дата обращения: 1.12.2018).
2. Хабаров, С.П. prolog – язык разработки интеллектуальных и экспертных систем: учебное пособие / С.П.Хабаров. – Санкт-Петербург, 2013. – 138 с.
3. Экспертные системы [Электронный ресурс]. – Режим доступа: <http://www.aiportal.ru/articles/expert-systems/expert-systems.html> (Дата обращение: 1.12.2018)
4. Питер Джексон. Введение в экспертные системы = Introduction to Expert Systems. — 3-е изд. — М.: Вильямс, 2001. — С. 624.
5. Таунсенд К., Фохт Д. Проектирование и программная реализация экспертных систем на персональных ЭВМ: Пер. с англ. В. А. Кондратенко, С. В. Трубицына. — М.: Финансы и статистика, 1990. — 320 с.

# ПРИЛОЖЕНИЕ A

database

yes(string)

no(string)

maybe(string)

predicates

repeat

run

animal(string)

is(string)

positive(string)

negative(string)

xpositive(string)

xnegative(string)

ask(string,char)

remember(string,char)

delete\_all

clauses

repeat.

repeat:-repeat.

%-----------------------------------------------------------------------------

% описываем животных

% пингвин

animal("penguin"):-positive("can swim"),positive("has color"),

negative("tamed by man"),

negative("can fly"),is("bird").

% гусь

animal("goose"):-positive("can swim"),positive("has color"),

positive("tamed by man"),

positive("can fly"),is("bird").

% дикая утка

animal("wild duck"):-positive("can swim"),positive("has color"),

negative("tamed by man"),

positive("can fly"),is("bird").

% страус

animal("ostrich"):-negative("can swim"),positive("has color"),

negative("tamed by man"),

negative("can fly"),is("bird").

% стриж

animal("swift"):-negative("can swim"),positive("has color"),

negative("tamed by man"),

positive("can fly"),is("bird").

% соловей

animal("nightingale"):-negative("can swim"),positive("has color"),

negative("tamed by man"),

positive("can fly"),is("bird").

% щегол

animal("goldfinch"):-negative("can swim"),positive("has color"),

negative("tamed by man"),

positive("can fly"),is("bird").

% попугай

animal("parrot"):-negative("can swim"),

positive("has color"),

positive("tamed by man"),

positive("can fly"),is("bird").

% заяц

animal("hare"):-positive("can swim"),positive("has color"),

negative("can fly"),

positive("has big teeth and claws"),

negative("tamed by man"),

is("mammal").

% крот

animal("mole"):-negative("can fly"),negative("has color"),

positive("has big teeth and claws"),

negative("tamed by man"),

negative("can swim"),positive("blind"),

is("mammal").

% зебра

animal("zebra"):-negative("can fly"),positive("has color"),

negative("has big teeth and claws"),

positive("has horns"),

negative("tamed by man"),

positive("can swim"),is("mammal"),

is("hoofed").

% корова

animal("cow"):-negative("can fly"),positive("has color"),

negative("has big teeth and claws"),

positive("has horns"),

positive("tamed by man"),

positive("can swim"),is("mammal"),is("hoofed").

% бобер

animal("beaver"):-negative("can fly"),negative("has color"),

positive("has big teeth and claws"),

negative("tamed by man"),

positive("can swim"),is("mammal").

% волк

animal("wolf"):-negative("can fly"),negative("has color"),

positive("has big teeth and claws"),

negative("tamed by man"),

positive("can swim"),is("predator").

% тигр

animal("tiger"):-negative("can fly"),positive("has color"),

positive("has big teeth and claws"),

negative("tamed by man"),

positive("can swim"),is("predator").

% слон

animal("elephant"):-negative("can fly"),negative("has color"),

negative("has big teeth and claws"),

positive("tamed by man"),

positive("can swim"),is("mammal"),is("hoofed").

% людоед

animal("human\_eater"):-negative("can fly"),negative("has color"),

negative("has big teeth and claws"),

negative("tamed by man"),

positive("can swim"),is("predator").

% животное не найдено

animal("Something from another planet...").

%-----------------------------------------------------------------------------

% птица

is("bird"):-positive("lay eggs"),!;

positive("has plumage").

% хищник

is("predator"):-positive("live birth"),!;

negative("eating grass"),

positive("eating animals").

% млекопитающее

is("mammal"):-positive("live birth"),!;

positive("eating grass"),

negative("eating animals").

% парнокопытное

is("hoofed"):-positive("has hoof").

%-----------------------------------------------------------------------------

% работа с динамической базой данных

positive(X):-xpositive(X),!; xnegative(X),!,fail; ask(X,'y').

negative(X):-xnegative(X),!; xpositive(X),!,fail; ask(X,'n').

ask(X,R):-write(X," ? : "),readchar(Reply),write(Reply,"\n"),

remember(X,Reply),R=Reply.

% проверка присутствия фатов в динамической базе данных

xpositive(X):-yes(X);maybe(X).

xnegative(X):-no(X);maybe(X).

% запоминаем факт в динамической базе данных

remember(X,'y'):-asserta(yes(X)).

remember(X,'n'):-assertz(no(X)).

remember(X,'q'):-asserta(maybe(X)).

% очищаем память от фактов динамической базы данных

delete\_all:-retract(yes(\_)),delete\_all.

delete\_all:-retract(no(\_)),delete\_all.

delete\_all:-retract(maybe(\_)),delete\_all.

delete\_all.

%-----------------------------------------------------------------------------

run:-delete\_all,

repeat,

write("Otvechat' (y-da, n-net, q-ne znayu).\n"),

animal(X),

write("Eto zhivotnoe : ",X,"\n"),

write("Zavershit' rabotu programmy (y/n): "),readchar(Z),

write(Z),nl,

Z='y'.

goal

makewindow(1,10,12," Expertnaya sistema : Zhivotnye ",0,0,25,80),

write("--------------------------------------------"),nl,

run.

# ПРИЛОЖЕНИЕ B

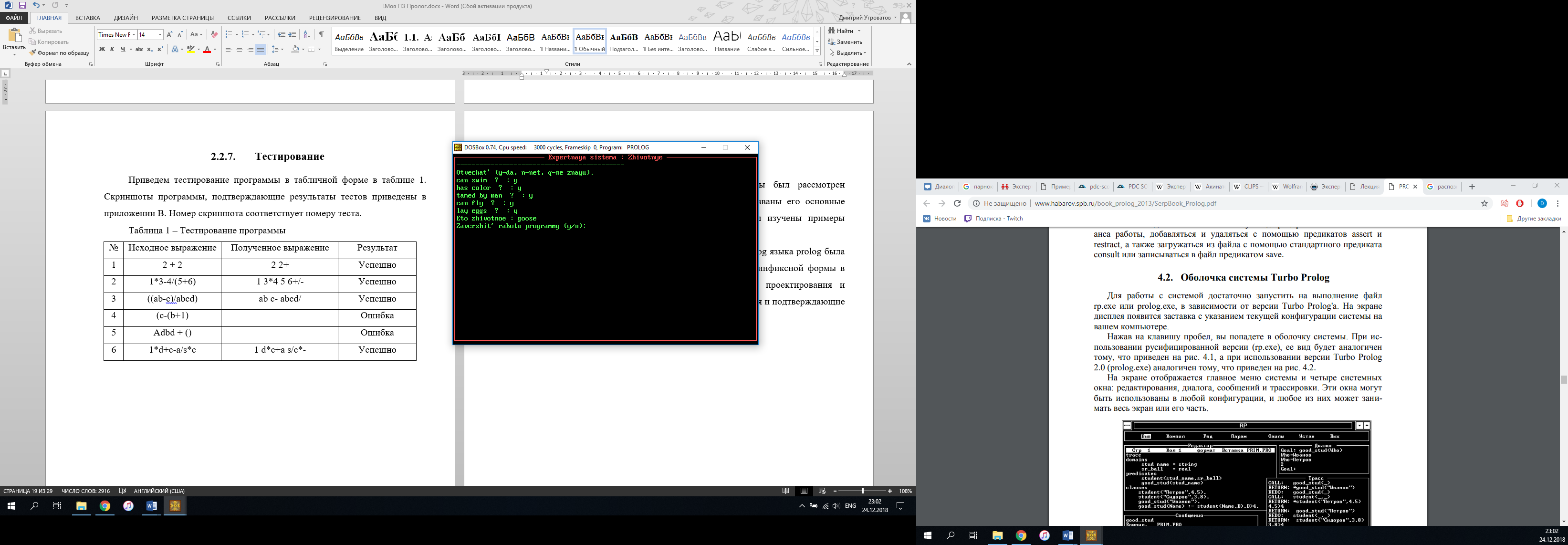


Рисунок 1 – Тест №1

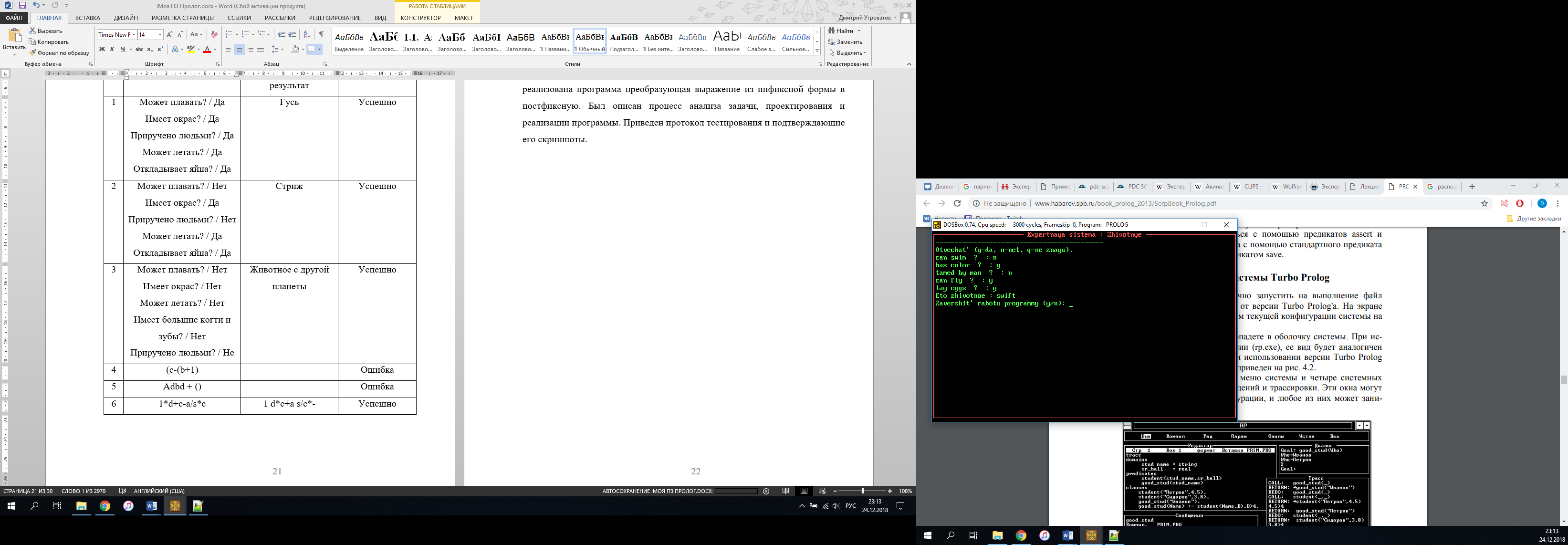


Рисунок 2 – Тест №2

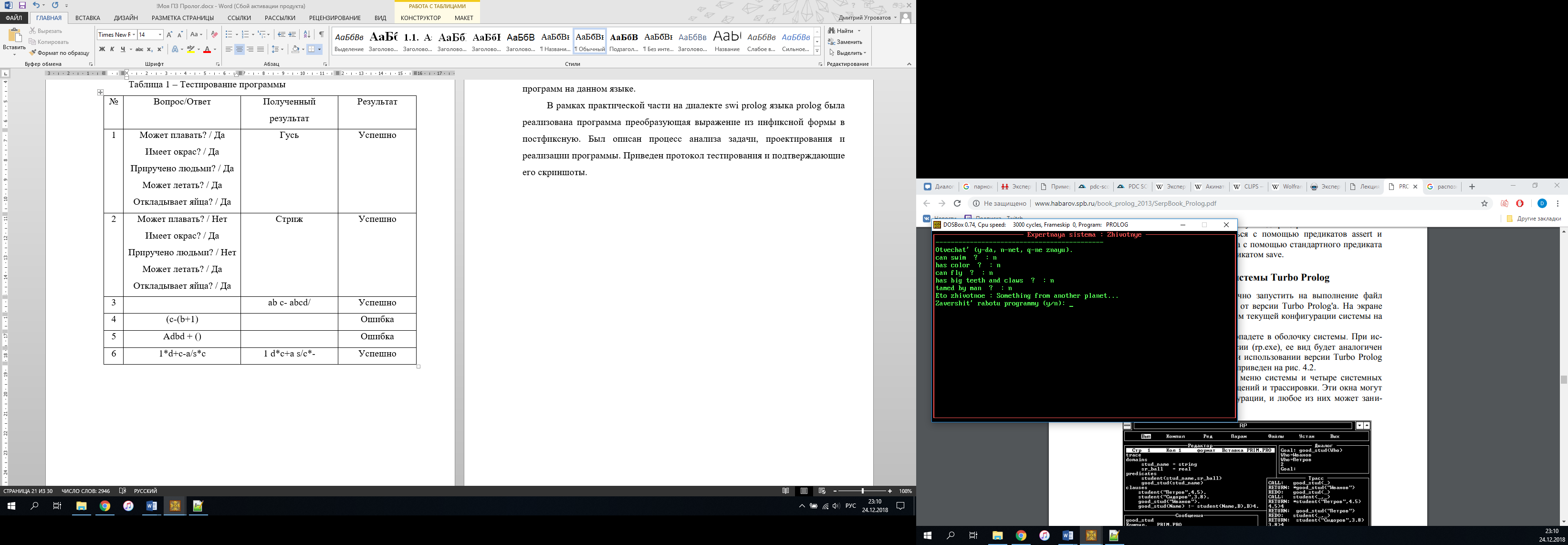


Рисунок 3 – Тест №3

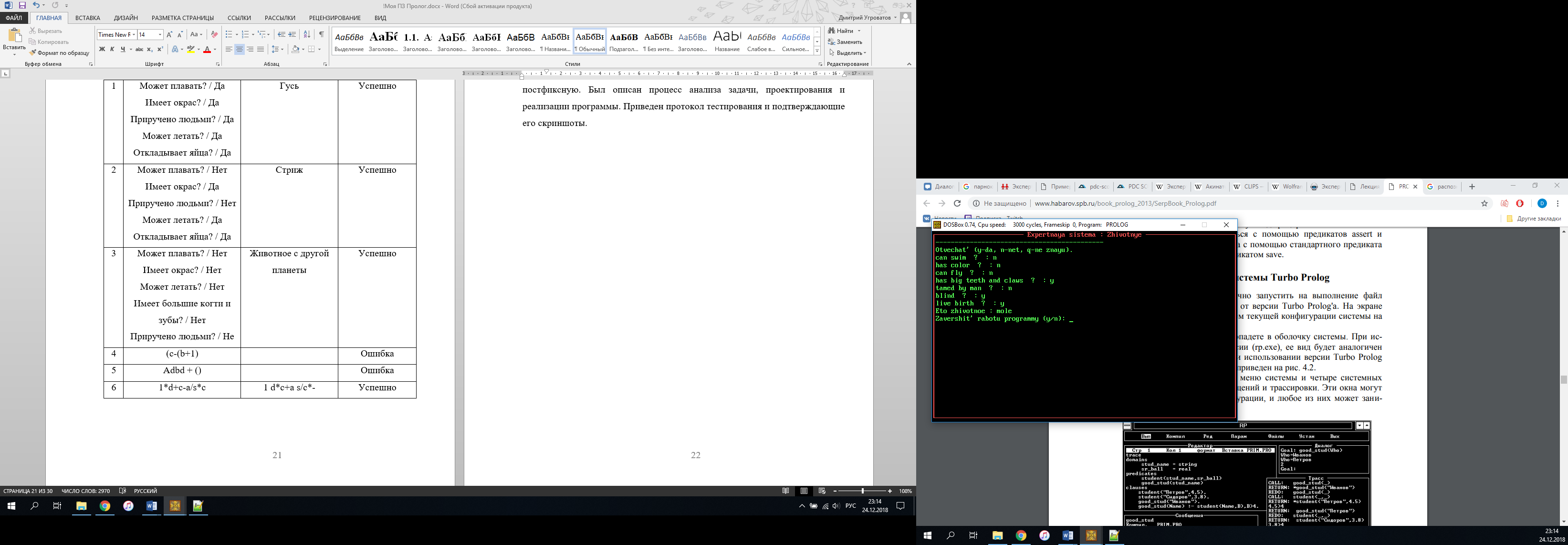


Рисунок 4 – Тест №4

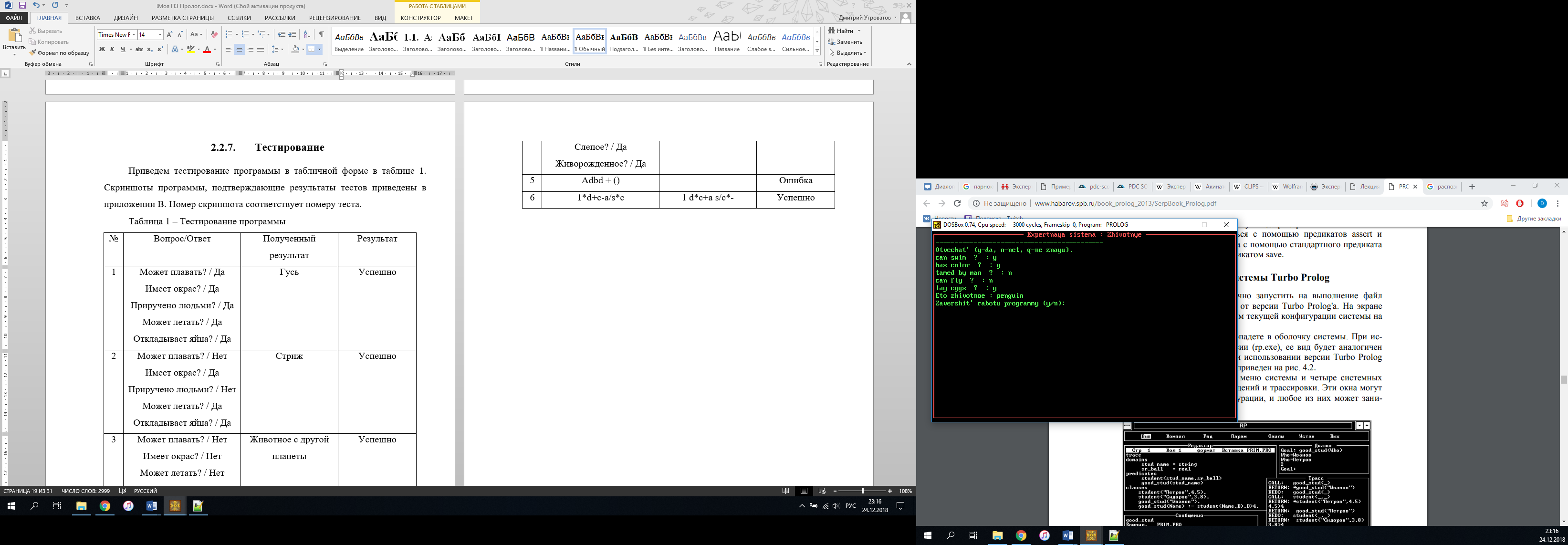


Рисунок 5 – Тест №5

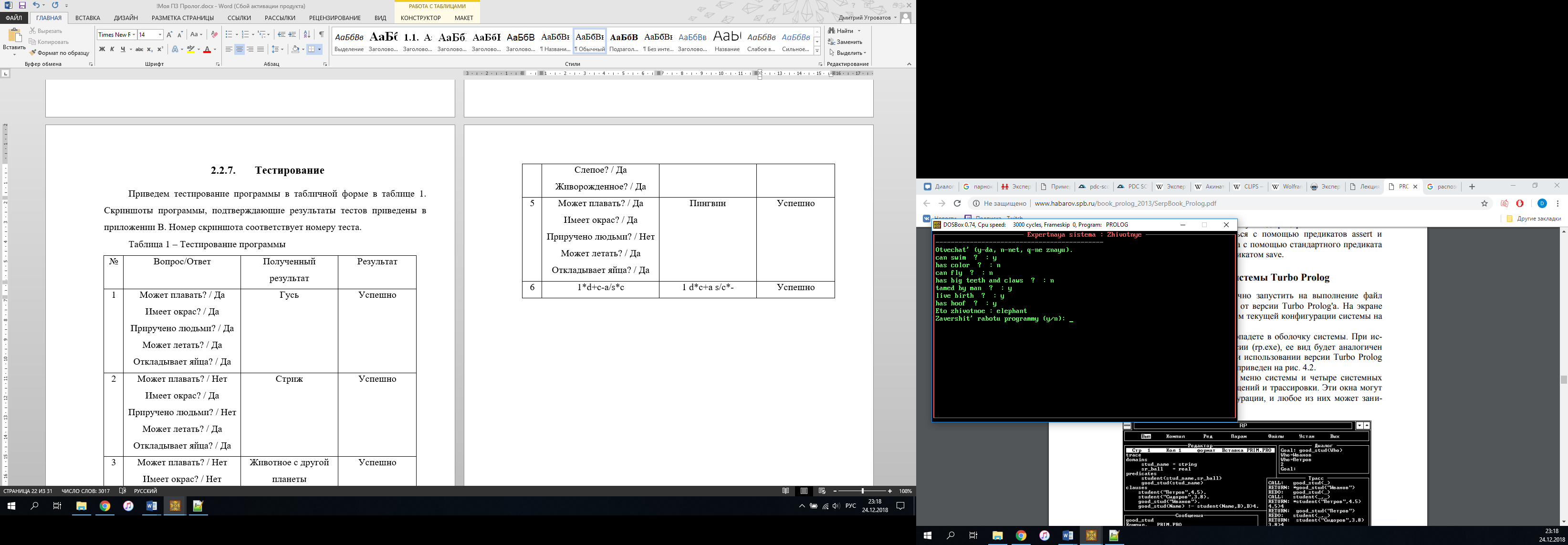


Рисунок 6 – Тест №6