|  |
| --- |
|  |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |
|  |
|  |

**Практическая работа 3-4.**

**Дисциплина: Основы экспертных систем**

**Тема: Основы работы с Prolog**

**Москва, 2020**

Оглавление

[Цель работы: 3](#_Toc37034680)

[Теоретическая часть. Основы языка Prolog 3](#_Toc37034681)

[Порядок выполнения практической работы 6](#_Toc37034682)

[Контрольные вопросы 10](#_Toc37034683)

[Варианты индивидуального задания 1: 11](#_Toc37034684)

[Варианты индивидуального задания 2: 12](#_Toc37034685)

[Список литературы 13](#_Toc37034686)

# Цель работы:

Познакомиться с основами работы с языком Prolog.

# Теоретическая часть. Основы языка Prolog

Начнем с того, что познакомимся с так называемой **нормальной формой Бэкуса-Наура (БНФ)**, разработанной в 1960 Джоном Бэкусом и Питером Науром и используемой для формального описания синтаксиса языков программирования. Впервые *БНФ* была применена Питером Науром при записи синтаксиса языка Алгол-60.

При описании синтаксиса конструкций используются следующие обозначения:

Символ ::= читается как "по определению" ("это", "есть"). Слева от разделителя располагается объясняемое понятие, справа - конструкция, разъясняющая его. Например,

<Имя> ::= <Идентификатор>

В угловые скобки заключается часть выражения, которая используется для обозначения синтаксической конструкции языка, в частности объясняемое понятие. В приведенном выше примере это <Имя> и <Идентификатор>.

Символ | означает в нотации *БНФ* "или", он применяется для разделения различных альтернативных растолкований определяемого понятия.

**Пример**. Десятичную цифру можно определить следующим образом:

<цифра> ::= 0|1|2|3|4|5|6|7|8|9

Часть синтаксической конструкции, заключенная в квадратные скобки, является необязательной (может присутствовать или отсутствовать);

**Пример**. Запись

<Целое число> ::= [-]<Положительное целое число>

означает, что целое число можно определить через положительное целое число, перед которым может стоять знак минус.

Символ \* обозначает, что часть синтаксической конструкции может повторяться произвольное число раз (ноль и более). Заметим, что иногда вместо символа \* используют фигурные скобки ( {, } ).

**Пример**. Определить положительное целое число в нотации *БНФ* можно следующим образом:

<Положительное целое число> ::= <цифра>[<цифра>]\*.

То есть положительное целое число состоит из одной или нескольких цифр.

Программа на языке Пролог, ее иногда называют базой знаний, состоит из *предложений* (или **утверждений** ), каждое *предложение* заканчивается точкой.

*Предложения* бывают двух видов: *факты*, *правила*.

*Предложение* имеет вид

A:-

B1,... , Bn.

A называется **заголовком** или **головой***предложения*, а B1,..., Bn - **телом**.

В принципе об этом уже говорилось в предыдущей лекции. Но там мы рассматривали эти понятия в основном с теоретической точки зрения, заходя со стороны математической логики, а сейчас наш подход будет больше практическим, со стороны программирования.

***Факт*** констатирует, что между объектами выполнено некоторое отношение. Он состоит только из заголовка. Можно считать, что *факт* - это *предложение*, у которого тело пустое.

Например, известный нам *факт*, что Наташа является мамой Даши, может быть записан в виде:

мама(Наташа, Даша).

*Факт* представляет собой безусловно истинное утверждение.

Напомню, что в математической логике, с которой мы познакомились в предыдущей лекции, отношения принято называть **предикатами**.

Если воспользоваться нормальной формой Бэкуса-Науэра, то предикат можно определить следующим образом:

<Предикат>::=<Имя> | <Имя>(<аргумент>[,<аргумент>]\*),

т.е. предикат состоит либо только из имени, либо из имени и следующей за ним последовательности аргументов, заключенной в скобки.

Аргументом или параметром предиката может быть константа, переменная или составной объект. Число аргументов предиката называется его **арностью** или **местностью**. Про переменные мы поговорим чуть-чуть позже, а подробное рассмотрение констант отложим до пятой лекции. Пока отметим, что константа получает свое значение в разделе описания констант, а переменная означивается в процессе работы программы.

***Правило*** - это *предложение*, истинность которого зависит от истинности одного или нескольких *предложений*. Обычно *правило* содержит несколько хвостовых *целей*, которые должны быть истинными для того, чтобы *правило* было истинным.

В нотации *БНФ* *правило* будет иметь вид:

<Правило>::=<предикат>:-<предикат>[,<предикат>]\*.

**Пример**. Известно, что бабушка человека - это мама его мамы или мама его папы.

Соответствующие *правила* будут иметь вид:

бабушка(X,Y):-

мама(X,Z),мама(Z,Y).

бабушка(X,Y):-

мама(X,Z),папа(Z,Y).

Символ " :- " означает "если", и вместо него можно писать if.

Символ " ," - это логическая связка "и" или конъюнкция, вместо него можно писать and.

Первое *правило* сообщает, что X является бабушкой Y, если существует такой Z, что X является мамой Z, а Z - мамой Y. Второе *правило* сообщает, что X является бабушкой Y, если существует такой Z, что Xявляется мамой Z, а Z - папой Y.

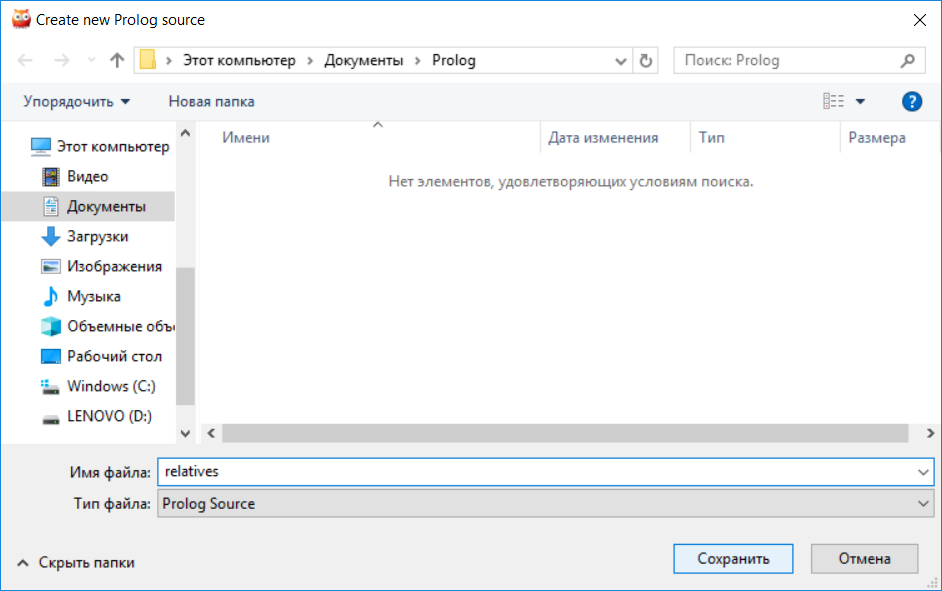
В данном примере X, Y и Z - это переменные.

Переменные могут быть **свободными** или **связанными**.

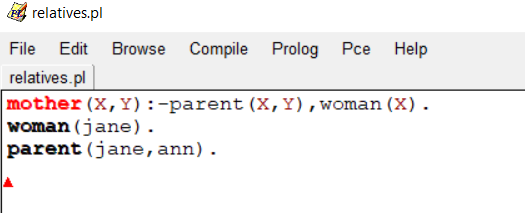
Оператор **отсечения** в программе на Прологе он обозначается восклицательным знаком " !". Этот предикат предназначен для ограничения пространства поиска, с *целью* повышения эффективности работы программ. Он всегда завершается успешно. После того, как до него дошла очередь, он устанавливает "забор", который не дает "откатиться назад", чтобы выбрать альтернативные решения для уже "сработавших" подцелей. То есть для тех, которые расположены левее *отсечения*. На *цели*, расположенные правее, *отсечение* не влияет. Кроме того, *отсечение* отбрасывает все *предложения* процедуры, расположенные после *предложения*, в котором находится *отсечение* .

# Порядок выполнения практической работы

1. Запустите SWI Prolog
2. Нажмите File->New. Задайте путь, по которому будет сохранена Ваша Prolog-программа.



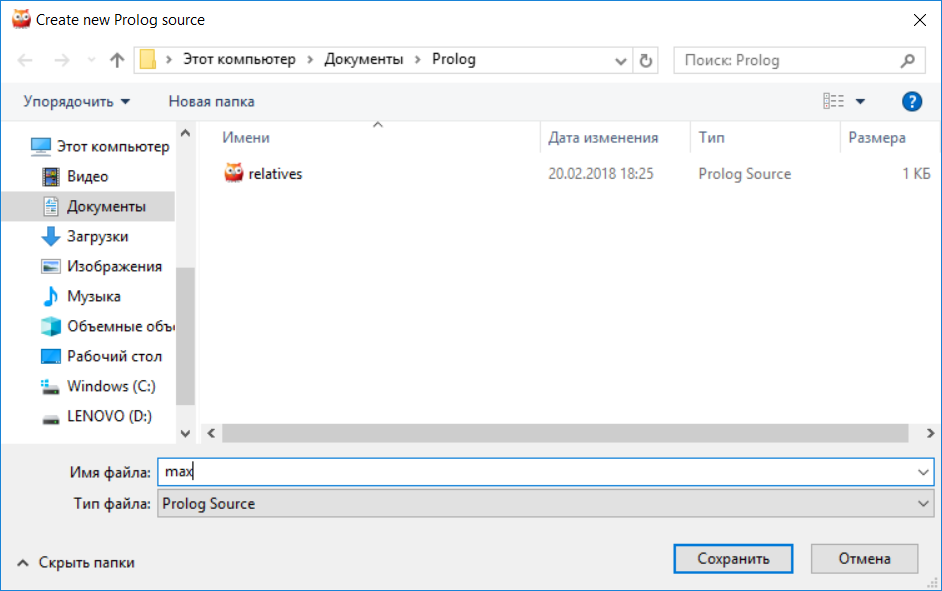
1. Введите следующую программу, описывающую родственные связи:



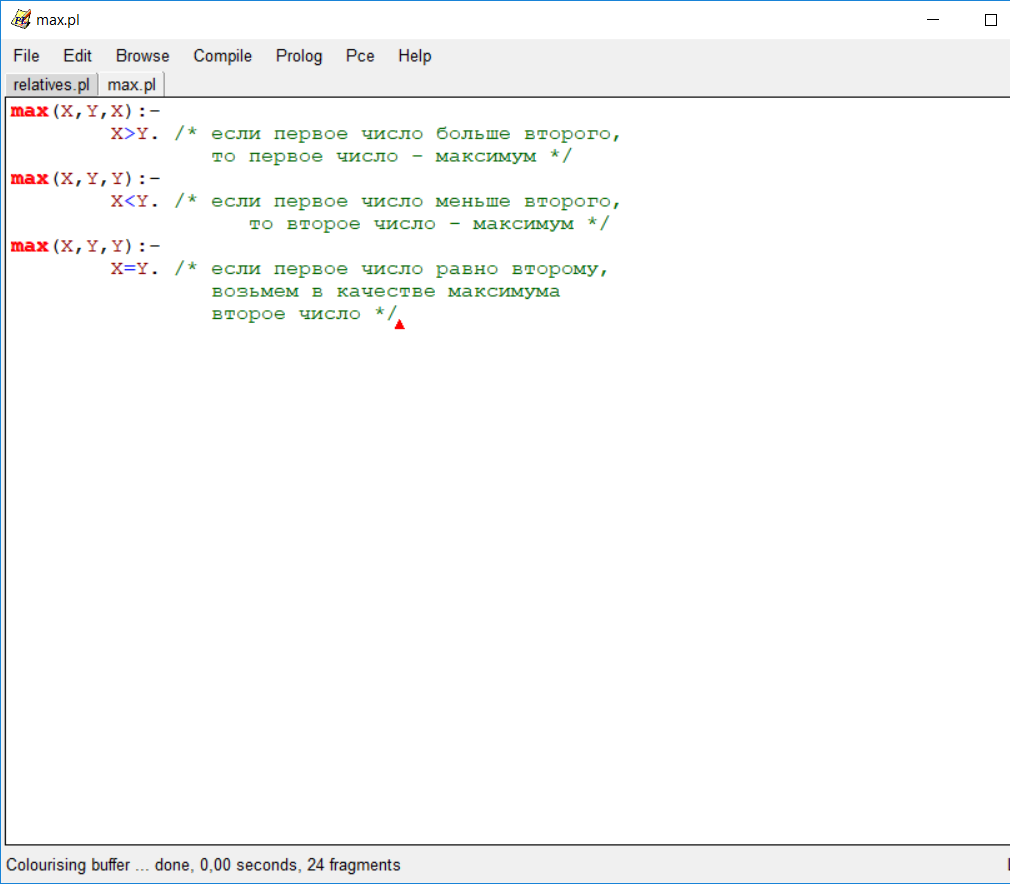
1. В главном окне выберите File->Consult->relatives.pl
2. Введите запрос:  
   mother(X,ann).
3. Введите запрос:   
   mother(jane,ann).
4. Введите запрос:

mother(ann,jane).

1. Создайте новую программу: File->New



1. Введите следующую программу поиска максимума из двух чисел:



1. File->Consult->max.pl
2. Введите запрос:

max(3,4,X).

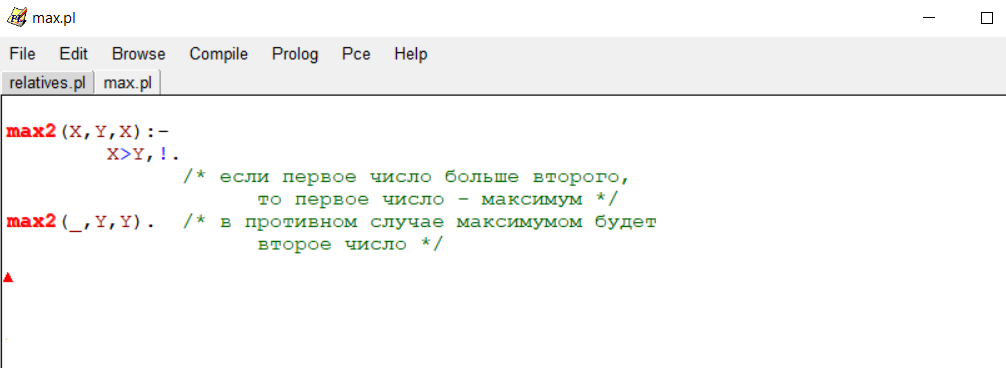
1. Введите запрос:

max(3,4,3).

1. Введите запрос:

max(3,4,4).

1. Видоизменим программу с использованием оператора отсечения:



1. Введите запрос:

max2(3,4,X).

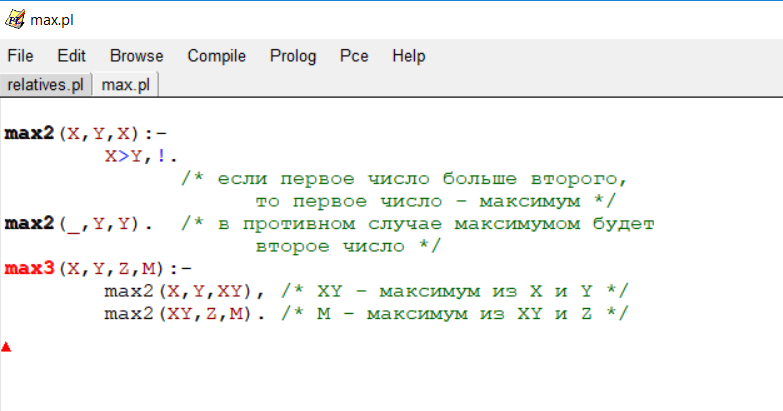
1. Введите запрос:

max2(3,4,3).

1. Введите запрос:

max2(3,4,4).

1. Добавим в программу предикат, находящий максимум из трех чисел



1. Введите запрос:  
   max3(3,4,6,X).
2. Введите запрос:

max3(3,4,6,6).

# Контрольные вопросы

1. Экспертные системы: определение, назначение
2. Основные функциональные блоки экспертной системы
3. К каким языкам относится Prolog
4. Из чего состоит программа на языке Prolog?
5. Что такое анонимная переменная?
6. Что такое оператор отсечения?

# Варианты индивидуального задания 1:

Вариант задания определяется по последней цифре в зачетной книжке (от 0 до 9)

1. Создайте на языке Prolog предикат, проверяющий, являются ли два человека дедушкой и внуком.
2. Создайте на языке Prolog предикат, проверяющий, являются ли два человека родственниками.
3. Создайте на языке Prolog предикат, проверяющий, являются ли два человека сёстрами.
4. Создайте на языке Prolog предикат, проверяющий, являются ли два человека дядей и племянником.
5. Создайте на языке Prolog предикат, проверяющий, являются ли два человека братьями.
6. Создайте на языке Prolog предикат, проверяющий, являются ли два человека супругами.
7. Создайте на языке Prolog предикат, проверяющий, являются ли два человека бабушкой и внучкой.
8. Создайте на языке Prolog предикат, проверяющий, являются ли два человека прапрабабушкой и праправнучкой.
9. Создайте на языке Prolog предикат, проверяющий, являются ли два человека двоюродными братом и сестрой.
10. Создайте на языке Prolog предикат, проверяющий, являются ли два человека тёщей и зятем.

# Варианты индивидуального задания 2:

Вариант задания определяется по последней цифре в зачетной книжке (от 0 до 9)

1. Создайте на языке Prolog предикат, находящий максимум из 4 чисел.
2. Создайте на языке Prolog предикат, находящий гипотенузу прямоугольного треугольника по длинам катетов.
3. Создайте на языке Prolog предикат, находящий абсолютное значение числа.
4. Создайте предикат, имеющий пять аргументов, и проверяющий, попадает ли точка, чьи координаты заданы первыми двумя параметрами, в круг, центр которого определяют третий и четвертый параметр, а радиус - пятый.
5. Создайте предикат, рассчитывающий площадь круга по его радиусу
6. Создайте предикат, находящий площадь квадрата
7. Создайте предикат, находящий объем куба по его ребру
8. Создайте предикат, находящий объем цилиндра по радиусу и высоте
9. Создайте предикат, находящий объем шара по радиусу
10. Создайте предикат, находящий объем параллелепипеда по высоте, ширине и длине

## Список литературы

1. <https://www.swi-prolog.org/download/stable>
2. <https://github.com/SWI-Prolog/swipl-devel>
3. <https://github.com/Anniepoo/prolog-examples>
4. <https://habr.com/ru/hub/prolog/>
5. <http://www.habarov.spb.ru/book_prolog_2013/SerpBook_Prolog.pdf>
6. <http://madiasunik.ucoz.ru/intel_system/prolog_metod.pdf>
7. <https://www.intuit.ru/studies/courses/44/44/lecture/32085?page=1>
8. <https://www.intuit.ru/studies/courses/44/44/lecture/32085?page=3>
9. <http://verim.org/project/prolog/swi/second>