



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        «Программное обеспечение ЭВМ и информационные технологии»

## ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1,2,3

По курсу: "Функциональное и Логическое программирование"

Тема \_\_\_\_\_ Основы prolog.  
Группа \_\_\_\_\_ ИУ7-63Б  
Студент \_\_\_\_\_ Сукочева А.  
Преподаватель \_\_\_\_\_ Толпинская Н.Б.  
Преподаватель \_\_\_\_\_ Строганов Ю. В.

# Теоретическая часть

## Логическое программирование

**Логическое программирование** – это программирование на знаниях. Логическое программирование принципиально работает не с данными, а со знаниями. Порядок действий в системе логического программирования prolog определяется алгоритмом унификации.

Программируя на императивном я.п., программист должен задать порядок действий, (который в дальнейшем будет последовательно выполнен), т.е. объяснить компьютеру **как** решать задачу. В случае декларативного я.п. программист должен описать **что** нужно решить (А уже с помощью алгоритма унификации будет решена задача)

## Элементы языка

**Элементы языка:** терм.

**Терм** - это:

### 1. константа:

- (a) число (пример: 1; 55; 12.32);
- (b) символьный атом. Начинается со **строчной** буквы и используется для обозначения конкретного объекта предметной области или для обозначения конкретного отношения (пример: alice, iu7\_63B);
- (c) строка - последовательность символов, заключенная в кавычки (пример: "Элис Сукочева").

### 2. переменная

- (a) Именованная - комбинация символов, цифр и ' \_ ', которая начинается с **прописной** буквы или с символа ' \_ ' (пример: Name, \_A). Переменные могут связываться с различными объектами – конкретизироваться;
- (b) Анонимная - обозначается символом ' \_ '. Анонимные переменные не могут быть связаны со значением.

3. **составной терм** - взаимосвязанная информация. Показывает связь между объектами. Т.е. кто от чего зависит. Синтаксис:  $f(t_1, t_2, \dots, t_m)$ , где  $f$  - функтор - символьная константа, обозначающая имя отношения.  $t_1, t_2, \dots, t_m$  - термы, являющиеся аргументами. Арность - число аргументов. (Пример: учится(элис, мгу) - знание о то, что Элис учится в МГТУ.

## Программа на prolog

**Программа на prolog** - база знаний и вопрос.

База знаний - это факты и правила. Каждое предложение (факт или правило) должно заканчиваться точкой.

**Правило** имеет вид:  $A :- B_1, \dots, B_n$

A - заголовок правила (терм).

$B_1, \dots, B_n$  - тело правила (термы).

Символ ":-" это специальный символ-разделитель.

**Факт** – это частный случай правила. Факт – это предложение, в котором отсутствует тело (т.е. тело пустое).

**Заголовок** - составной терм, который содержит знание. Знания в заголовках.

**В теле** прописаны условия истинности этого знания (которое написано в заголовке).

В разделе **CLAUSES** записываются факты и правила.

Пример:

```
CLAUSES
    study(alice, bmstu).
    study(ivan, bmstu).
```

**Вопрос** - частный случай правила, состоит только из тела (составного терма или нескольких составных термов). Используется, чтобы определить, выполняется ли некоторое отношение между описанными в программе объектами. Ответом может быть "Yes" или "No".

В разделе GOAL содержатся цели, которые нужно достигнуть. Главная задача заключается в том, чтобы дать ответ "Yes" на поставленный вопрос. В случае, если система не может ответить "Yes" система отвечает "No".

Пример:

```
GOAL
    study(alice, bmstu) % Yes
```

## Механизм унификации

**Механизм унификации** (подбор нужного решения). Поиск ответа на поставленный вопрос заключается в поиске нужного знания с помощью механизма унификации. Данный механизм встроен в систему и недоступен программисту.

**Процедуры** - совокупность правил, заголовки которых имеют одно и то же имя и одну и ту же аргументность.

**Предикат** - отношение, определяемое процедурой.

## Практическая часть л.р.1

**Задание 1.** Разработать свою программу - «Телефонный справочник». Протестировать работу программы.

```
PREDICATES
    phonebook(phone, name, school\_number).

CLAUSES
    phonebook("+79998881234", "Alice", 433).
    phonebook("+79998881235", "Pasha", 415).
    phonebook("+79998881236", "Nastya", 433).
    phonebook("+79998881237", "Ivan", 424).
    phonebook("+79998881238", "Dima", 123).
    phonebook("+79998881134", "Oleg", 123).
    phonebook("+79991881334", "Nikita", 25).
    phonebook("+79998391234", "Misha", 1024).

GOAL
    phonebook(Phone\_number, Name, 433).
```

## Практическая часть л.р.2

**Задание 2.** Составить программу – базу знаний, с помощью которой можно определить, например, множество студентов, обучающихся в одном ВУЗе. Студент может одновременно обучаться в нескольких ВУЗах. Привести примеры возможных вариантов вопросов и варианты ответов (не менее 3-х).

### DOMAINS

```
student_id = integer.
student_name = symbol.
student_surname = symbol.

university_id = integer.
university_name = symbol.
city = symbol.
```

### PREDICATES

```
study(student_id, university_id).
student(student_id, student_name, student_surname).
university(university_id, university_name, city).

students(university_id, student_name, student_surname).
```

### CLAUSES

```
student(0, "Alice", "Sukocheva").
student(1, "Nika", "Lilova").
student(2, "Masha", "Perestoronina").
student(3, "Pasha", "Perestoronin").
student(4, "Nastya", "Namestnik").
student(5, "Kirill", "Drovin").
student(6, "Tim", "Malov").

university(0, "Bauman_Moscow_State_Technical_University", "Moscow").
university(1, "Petersburg_State_University", "Petersburg").
university(2, "Kuban_State_University", "Krasnodar").

study(0, 0).
study(2, 0).
study(3, 0).
study(4, 0).

study(5, 2).
study(5, 0).

study(6, 0).
study(6, 1).
study(6, 2).

students(University_id, Name, Surname) :-
    study(Student_id, University_id),
    student(Student_id, Name, Surname).
```

### GOAL

```
% All infomation about students.
% student(Id, Name, Surname).

% All infomation about universities.
% university(Id, Name, City).

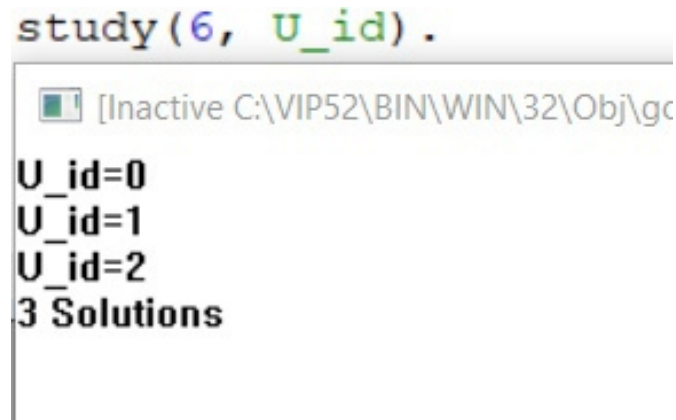
% All id students from the university.
% study(Id, 0). % 0, 2, 3, 4, 5, 6
```

```
% study(Id, 1). % 6
% study(Id, 2). % 5, 6

% All name and surname students from the university.
% students(0, Name, Surname).
% students(1, Name, Surname).
% students(2, Name, Surname).

% All universities where the student studies.
% study(0, U_id). % 0
% study(1, U_id). % study(0, U_id).
study(6, U_id). % 0, 1, 2
```

На рис. 1 продемонстрировано определение id вузов, в которых учится студент с id = 6.



```
study(6, U_id).
[Inactive C:\VIP52\BIN\WIN\32\Obj\gc
U_id=0
U_id=1
U_id=2
3 Solutions
```

Рис. 1: Результат работы 1.

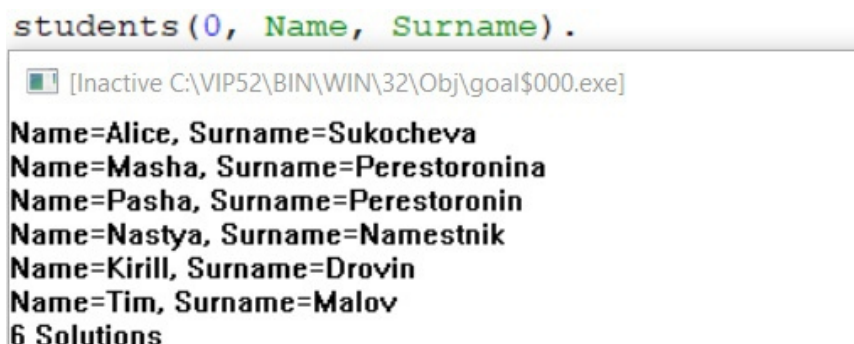
На рис. 2 продемонстрировано определение имени и фамилии студентов, обучающихся в вузе с id равным двум.



```
students(2, Name, Surname).
[Inactive C:\VIP52\BIN\WIN\32\Obj\goal$000.exe]
Name=Kirill, Surname=Drovin
Name=Tim, Surname=Malov
2 Solutions
```

Рис. 2: Результат работы 2.

На рис. 3 продемонстрировано определение имени и фамилии студентов, обучающихся в вузе с id равным 0.



```
students(0, Name, Surname).
[Inactive C:\VIP52\BIN\WIN\32\Obj\goal$000.exe]
Name=Alice, Surname=Sukocheva
Name=Masha, Surname=Perestoronina
Name=Pasha, Surname=Perestoronin
Name=Nastya, Surname=Namestnik
Name=Kirill, Surname=Drovin
Name=Tim, Surname=Malov
6 Solutions
```

Рис. 3: Результат работы 3.

На рис. 4 продемонстрировано определение id студентов, обучающихся в вузе с id равным 0.

```
study(Id, 0). % 0, 2, 3, 4, 5, 6
```



Рис. 4: Результат работы 4.

На рис. 4 продемонстрировано определение всех имеющихся в базе данных вузов.

```
university(Id, Name, City).
```



Рис. 5: Результат работы 5.

## Практическая часть л.р.3

**Задание 3.** Составить программу, т.е. модель предметной области – базу знаний, объединив в ней информацию – знания:

1. Телефонный справочник: Фамилия, Номер телефона, Адрес – структура (Город, Улица, Нодома, Нокв),
2. Автомобили: Фамилия владельца, Марка, Цвет, Стоимость, и др.,
3. Вкладчики банков: Фамилия, Банк, счет, сумма, др. Владелец может иметь несколько телефонов, автомобилей, вкладов (Факты). Используя правила, обеспечить возможность поиска:

1. а) По номеру телефона найти: Фамилию, Марку автомобиля, Стоимость автомобиля (может быть несколько), в) Используя сформированное в пункте а) правило, по номеру телефона найти: только Марку автомобиля (автомобилей может быть несколько),

2. Используя простой, не составной вопрос: по Фамилии (уникальна в городе, но в разных городах есть однофамильцы) и Городу проживания найти: Улицу проживания, Банки, в которых есть вклады и номер телефона.

```
DOMAINS
    surname = symbol.
    phone_number = symbol.
    address_struct = address(symbol, symbol, integer, integer).

    label = symbol.
```

```

color = symbol.
price = integer.

bank = symbol.
score = integer.
sum = integer.

```

#### PREDICATES

```

phonebook(surname, phone_number, address_struct).
car(surname, label, color, price).
bank_depositor(surname, bank, score, sum).

f1(surname, phone_number, label, price).
f1_2(phone_number, label).
f2(surname, symbol, phone_number, symbol, bank).

```

#### CLAUSES

```

phonebook("Tilov", "89999899999", address("Moscow", "2_baumanskaya",
57, 25)).
phonebook("Tilov", "89999899977", address("Moscow", "2_baumanskaya",
57, 25)).
phonebook("Alovik", "89999812999", address("Moscow", "3_baumanskaya",
50, 75)).

car("Tilov", "Buick", "black", 12000000).
car("Tilov", "Cadillac", "white", 22000000).

bank_depositor("Alovik", "sberbank", 10000, 25000).
bank_depositor("Alovik", "vtb", 20000, 35000).

f1(Surname, Phone_number, Label, Price) :-
    phonebook(Surname, Phone_number, _),
    car(Surname, Label, _, Price).

f1_2(Phone_number, Label) :-
    f1(_, Phone_number, Label, _).

f2(Surname, City, Phone_number, Street, Bank) :-
    phonebook(Surname, Phone_number, address(City, Street, _, _)),
    bank_depositor(Surname, Bank, _, _).

```

#### GOAL

```

f1(Surname, "89999899999", Label, Price).
% f1_2("89999899999", Label).
% f2("Alovik", "Moscow", Phone_number, Street, Bank).

```

## Ответы на вопросы

1. Что собой представляет программа на Prolog

Программа на Prolog представляет базу знаний и вопрос.

2. Какова ее структура.

Структура программы на Prolog.

1. директивы компилятора — зарезервированные символьные константы. (Пример ":-" — разделитель между заголовком и телом в правиле)

2. CONSTANTS — раздел описания констант.

3. DOMAINS — раздел описания доменов.

4. DATABASE — раздел описания предикатов внутренней базы данных.

5. PREDICATES — раздел описания предикатов.

6. CLAUSES — раздел описания предложений базы знаний.

7. GOAL — раздел описания внутренней цели (вопроса).

3. Как она реализуется.

Описывается база знаний и задается вопрос.

4. Как формируются результаты работы программы.

Система пытается найти такие значения переменных, при которых можно ответить на поставленный вопрос 'Да', используя базу знаний.