

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА  
(САМАРСКИЙ УНИВЕРСИТЕТ)»

Факультет Информатики   
Кафедра Программных систем

**ОТЧЕТ**  
  
по лабораторному практикуму по дисциплине

«Логическое программирование»

Лабораторная работа №2

Вариант № 8

Студент В. Д. Гижевская

Руководитель Д. С. Оплачко

Самара 2021

**Структура программы на языке Пролог.**

Программа, написанная на Прологе, состоит из пяти основных разделов:

* раздел описания доменов,
* раздел базы данных,
* раздел описания предикатов,
* раздел описания предложений,
* раздел описания цели.

Ключевые слова **domains, constants, database (facts), predicates, clauses** и **goal** отмечают начала соответствующих разделов.

Назначение этих разделов таково:

* раздел **domains** содержит определения доменов, которые описывают различные типы данных, используемых в программе;
* раздел **constants** используется для объявления символических констант, используемых в программе;
* раздел **database (facts)** содержит описания предикатов внутренней базы данных Пролога, если программа такой базы данных не требует, то этот раздел может быть опущен;
* раздел **predicates** служит для описания предикатов, не принадлежащих внутренней базе данных;
* в раздел **clauses** заносятся факты и правила самой программы;
* в разделе **goal** на языке Пролог формулируется назначение создаваемой программы.

**Пролог для решения логических задач**

ПРОЛОГ позволяет наиболее естественным образом решать логические задачи, моделируя процесс размышления человека с помощью правил. Многие логические задачи связаны с рассмотрением нескольких конечных множеств с одинаковым количеством элементов, между которыми устанавливается взаимно-однозначное соответствие. В ПРОЛОГе эти множества можно описывать как базы данных, а зависимости между объектами устанавливать с помощью правил. Операция деления списка на голову и хвост обозначается при помощи вертикальной черты (|):

[Head | Tail].

Head здесь является переменной для обозначения головы списка, переменная Tail обозначает хвост списка (для имен головы и хвоста списка пригодны любые допустимые Прологом имена). Данная операция также присоединяет элемент в начало списка, например, для того, чтобы присоединить X к списку S следует написать [X | S]. Отличительной особенностью описания списков является наличие звездочки (\*) после имени домена элементов.

**Задание.**

В одном театре работают четыре актёра: Смирнов, Снегов, Морев и Никитин. Один из них играет роль Отелло, другой – короля Лира, третий – Ромео, четвёртый – Гамлета. Смирнов – не Отелло и не Гамлет. Морев – не Ромео и не Отелло. Никитин – не Гамлет, не Отелло. Снегов не играет ни Гамлета, ни Ромео. Если Морев играет Гамлета, то Смирнов не играет короля Лира. Кто из актёров кого играет?

**Листинг программы.**

Domains

name,role=symbol

actor = c(name, role)

actors = actor\*

Predicates

nondeterm solve

name(actor ,name)

role(actor, role)

nondeterm in(actor, actors)

Clauses

name(c(N,\_), N).

role(c(\_,D), D).

in(X,[X,\_,\_,\_]).

in(X,[\_,X,\_,\_]).

in(X,[\_,\_,X,\_]).

in(X,[\_,\_,\_,X]).

solve:- in(C1,Actors),name(C1,смирнов),

in(C2,Actors),name(C2,снегов),

in(C3,Actors),name(C3,морев),

in(C4,Actors),name(C4,никитин),

in(C5,Actors),role(C5,гамлет),

in(C6,Actors),role(C6,оттело),

in(C7,Actors),role(C7,король\_лир),

in(C8,Actors),role(C8,ромео),

not(role(C1,оттело)),

not(role(C1,гамлет)),

not(role(C3,оттело)),

not(role(C3,ромео)),

not(role(C2,гамлет)),

not(role(C2,ромео)),

not(role(C4,гамлет)),

not(role(C4,оттело)),

role(C3,гамлет),

not(role(C1,король\_лир)),

write(Actors).

Goal

solve.

**Результат работы программы.**

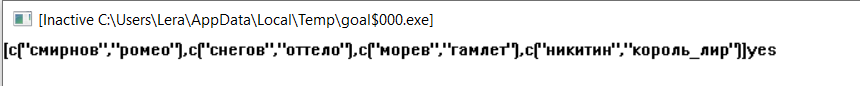


Рисунок 1 – Результат работы программы