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»

Лабораторная работа № 2

По дисциплине «Функциональное и логическое программирование»

Студент: Тимонин А. С.

Группа ИУ7-62б

Преподаватель Толпинская Н. Б.

Москва.
2020 г.

Практическая часть

Задание

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). Описать порядок формирования вариантов ответа.

- Исходную базу знаний сформировать с помощью только фактов;
- Исходную базу знаний сформировать, используя правила;
- Разработать свою базу знаний (содержание произвольно).

В программе присутствуют следующие переменные:

- **firstName** - переменная типа symbol отвечает за имя студента
- **lastName** - переменная типа symbol отвечает за имя фамилию
- **group** - переменная типа integer отвечает за группу студента
- **edu** - переменная типа symbol отвечает за институт, в котором обучается студент
- **averageMark** - переменная отвечающая за средний балл студента, округленная в число формата integer

Листинг

```
domains
      firstName = symbol.
      lastName = symbol.
      group = integer.
      edu = symbol.
      averageMark = integer.

predicates
      nondeterm student(firstName, lastName, group, edu, averageMark).
      nondeterm educational(firstName, lastName, edu).
      nondeterm grpNedu(lastName, group, edu).
      nondeterm praviloEDU(firstName, lastName).
      nondeterm excellentPupil(firstName, lastName, averageMark).
      nondeterm badPup(firstName, lastName, averageMark).

clauses
```

```
	student("Anton", "Timonin", 2, "BMSTU", 2).
	student("Egor", "Timonin", 2, "BMSTU", 2).
	student("Vasya", "Bobkov", 1, "BMSTU", 3).
	student("Oleg", "Gibadulin", 2, "BMSTU", 4).
	student("Sanya", "Stepanov", 3, "BMSTU", 5).
	student("Ivan", "Petrov", 1, "BMSTU", 3).
	student("Jenya", "Jekajio", 13, "MGSU", 3).
	student("Elizaveta", "Boyarko", 3, "MGSU", 5).
	student("Petr", "Ivanov", 1, "MSU", 3).
	student("K", "S", 1, "MSU", 5).
	student("Kek", "Lol", 14, "HSE", 3).
	student("Andrey", "Rodin", 4, "MEPI", 1).

	% FACTS
	educational(FirstName, LastName, Edu) :-
		student(FirstName, LastName, _, Edu, _).

	grpNedu(LastName, Group, Edu) :-
		student(_, LastName, Group, Edu, _).

	excellentPupil(FirstName, LastName, AverageMark) :-
student(FirstName, LastName, _, _, AverageMark).

	% PRAVILA
	praviloEDU(FirstName, LastName) :-
		student(FirstName, LastName, _, Edu, _),
		Edu = "BMSTU".

	badPup(FirstName, LastName, AverageMark) :-
		student(FirstName, LastName, _, _, AverageMark),
		AverageMark = 2;
		student(FirstName, LastName, _, _, AverageMark),
		AverageMark = 1.

goal
	% Facts: Check institute
	educational(FirstName, LastName, "BMSTU").

	% Facts: Check group and institute
	grpNedu(LastName, 2, "BMSTU").

	% Facts: Check excellent pupils
	excellentPupil(FirstName, LastName, 5).

	% Pravila: Check institute
	praviloEDU(FirstName, LastName).

	% Pravila: Check bad pupils
	badPup(FirstName, LastName, AverageMark).
```

Тестирование

Факт 1

```
educational(FirstName, LastName, "BMSTU").

% Вывод

FirstName=Anton, LastName=Timonin
FirstName=Egor, LastName=Timonin
FirstName=Vasya, LastName=Bobkov
FirstName=Oleg, LastName=Gibadulin
FirstName=Sanya, LastName=Stepanov
FirstName=Ivan, LastName=Petrov
6 Solutions
```

Факт 2

```
% Функция ищет людей из вторых групп

grpNedu(LastName, 2, "BMSTU").

% Вывод

LastName=Timonin
LastName=Gibadulin
3 Solutions
```

Факт 3

```
% Функция ищет отличников

excellentPupil(FirstName, LastName, 5).

% Вывод

FirstName=Sanya, LastName=Stepanov
FirstName=Elizaveta, LastName=Boyarko
FirstName=K, LastName=S
3 Solutions
```

Правило 1

```
% Функция ищет людей с институтом BMSTU

praviloEDU(FirstName, LastName).

% Вывод
```

```
FirstName=Anton, LastName=Timonin
FirstName=Egor, LastName=Timonin
FirstName=Vasya, LastName=Bobkov
FirstName=Oleg, LastName=Gibadulin
FirstName=Sanya, LastName=Stepanov
FirstName=Ivan, LastName=Petrov
6 Solutions
```

Правило 2

```
% Функция ищет плохих учеников

badPup(FirstName, LastName, AverageMark).

% Вывод

FirstName=Anton, LastName=Timonin, AverageMark=2
FirstName=Egor, LastName=Timonin, AverageMark=2
FirstName=Andrey, LastName=Rodin, AverageMark=1
3 Solutions
```

Теоретическая часть

Формирование результата. Если описывать формирование результата по следующему куску кода, тогда он будет выглядеть примерно так.

```
q(2). q(3).
икс(X) :- q(X), X>2.
```

Первым делом выполняется предикат q(X). Он унифицирует предикат и самый первый факт q(2). В стек помещается адрес факта q(3), переменная X связывается с значением 2. В этот момент в стеке лежит значение q(3). Проверяется условие 2>2, оно не успешно. Механизм поиска выталкивает из стека адрес факта q(3), в стек ничего не помещает, тк дальше нет предикатов. Переменная связывается с значением 3. Проверяется условие 3>2. Оно успешно. На этом доказательство предиката икс завершается успешно со значением икс(3).

Программа на декларативном языке представляет собой совокупность утверждений, описывающих фрагмент предметной области (знания о предметной области) или сложившуюся ситуацию, а не порядок поиска решения. Программа на Prolog представляет собой набор фактов и правил, которые формируют базу знаний о предметной области.

Факты представляют собой составные термы, с помощью которых фиксируется наличие истинностных отношений между объектами предметной области — аргументами терма.

Правила являются обобщенной формулировкой условия истинности знания – отношения между объектами предметной области (аргументами терма), которое записано в заголовке правила.

Программа на Prolog состоит из разделов. Каждый раздел начинается со своего заголовка.

Структура программы:

- директивы компилятора — зарезервированные символьные константы
- CONSTANTS — раздел описания констант
- **DOMAINS** — раздел описания доменов
- DATABASE — раздел описания предикатов внутренней базы данных
- PREDICATES — раздел описания предикатов
- CLAUSES — раздел описания предложений базы знаний
- **GOAL** — раздел описания внутренней цели (вопроса).

*В программе не обязательно должны быть все разделы.

Реализация программы

Поиск содержательного ответа на поставленный вопрос, с помощью имеющейся базы знаний, фактически заключается в поиске нужного знания, но какое знание понадобится – заранее неизвестно. Этот поиск осуществляется формально с помощью механизма **унификации**, встроенного в систему и не доступного программисту. Упрощенно, процесс унификации можно представить как формальный процесс сравнивания (сопоставления) терма вопроса с очередным термом знания. При этом, знания по умолчанию просматриваются сверху вниз, хотя такой порядок и не очевиден. В процессе сравнивания для переменных «подбираются», исходя из базы знаний, значения (для именованных переменных). И эти подобранные для переменных значения возвращаются в качестве побочного эффекта ответа на поставленный вопрос.