|  |  |
| --- | --- |
| |  | | --- | | МИНОБРНАУКИ РОССИИ  Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | |

Институт Информационных технологий

Кафедра МОСИТ

**ЛАБОРАТОРНАЯ РАБОТА №1**

по дисциплине «Функциональное и Логическое Программирование»

**Студент группы** ИКБО-12-17 Васильев И.С. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(подпись студента)*

**Руководитель работы** Смольянинова В.А. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(подпись руководителя)*

Москва, 2019

# Постановка задачи

Задание к примерам программ на языке Пролог.

* Наберите программу. Запустите на выполнение. Придумайте и задайте 5-6 вопросов к каждой программе.

ЗАДАНИЕ

1. Составьте программу Родственные отношения, которая кроме родственных отношений parent (родитель) и ancestor (предок) программа должна содержать одно или несколько из следующих отношений: brother (брат); sister (сестра); grand-father (дедушка); grand-mother (бабушка); uncle (дядя).
2. Составьте программу, используя отношения likes «нравится») и can\_buy («может купить»).
3. Составьте собственную программу, состоящую из фактов и правил. Проверьте ее работу.

# Ход выполнения работы

Первым примером была программа, описывающая схему родства, показанную на рисунке 1.

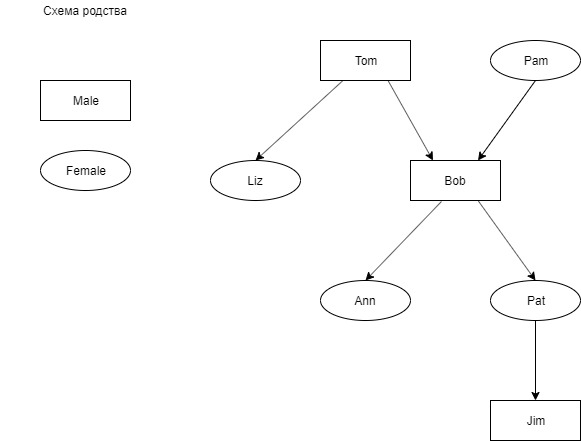


Рисунок 1. Схема родства

Код программы:

domains

s=symbol

predicates

nondeterm parent(s,s)

female(s)

male(s)

mother(s,s)

father(s,s)

nondeterm ancestor(s,s)

nondeterm child(s,s)

clauses

parent(pam,bob).

parent(tom,bob).

parent(tom,liz).

parent(bob,ann).

parent(bob,pat).

parent(pat,jim).

female(pam).

female(liz).

female(ann).

female(pat).

male(tom).

male(bob).

male(jim).

child(Y,X):- parent(X,Y).

mother(X,Y):- parent(X,Y),female(X).

father(X,Y):- parent(X,Y),male(X).

ancestor(X,Z):- parent(X,Z).

ancestor(X,Z):- parent(X,Y),ancestor(Y,Z).

Для данной программы были заданы следующие вопросы:

* Является ли pam женщиной?

goal

female(pam). /\* is pam female? \*/

* Перечислить родителей bob-а

goal

child(bob,Who). /\* find all people bob is a child of \*/

* Перечислить детей bob-а

goal

child(Who,bob). /\* find all children of bob \*/

* Является ли tom предком jim-а?

goal

ancestor(tom,jim). /\* is tom an ancestor of jim? \*/

* Перечислить всех потомков tom-а

goal

ancestor(tom,Who). /\* find all people tom is an ancestor of \*/

* Перечислить всех предков tom-а

goal

ancestor(Who,tom). /\* find all ancestors of tom \*/

Следующий пример рассматривал отношение «нравится»:

predicates

nondeterm likes(symbol,symbol)

clauses

likes(ellen, tennis).

likes(john, football).

likes(tom, baseball).

likes(eric, swimming).

likes(mark, tennis).

Были заданы следующие вопросы:

* Перечислить тех, кому нравится теннис

goal

likes(Who,tennis). /\* list all who like tennis \*/

* Перечислить тех, кому нравится футбол

goal

likes(Who,football). /\* list all who like football \*/

* Что нравится ellen?

goal

likes(ellen,What). /\* list what ellen likes \*/

* Нравится ли ellen теннис?

goal

likes(ellen,tennis). /\* does ellen like tennis? \*/

* Нравится ли eric-у плавание?

goal

likes(eric,swimming). /\* does eric like swimming? \*/

Третий пример рассматривал отношение «может купить»:

predicates

nondeterm can\_buy(symbol, symbol)

person(symbol)

nondeterm car(symbol)

likes(symbol, symbol)

for\_sale(symbol)

clauses

can\_buy(X, Y) :- person(X), car(Y),likes(X, Y), for\_sale(Y).

person(kelly).

person(judy).

car(lemon).

car(hot\_rod).

likes(kelly, hot\_rod).

likes(judy, pizza).

for\_sale(pizza).

for\_sale(lemon).

for\_sale(hot\_rod).

Были заданы следующие вопросы:

* Перечислить, что kelly может купить

goal

can\_buy(kelly, What). /\* list cars kelly can buy \*/

* Является ли пицца машиной?

goal

car(pizza). /\* is the pizza a car? \*/

* Нравится ли kelly hot\_rod?

goal

likes(kelly, hot\_rod). /\* does kelly like hot\_rod? \*/

* Продается ли hot\_rod?

goal

for\_sale(hot\_rod). /\* is the hot\_rod for sale? \*/

* Может ли kelly купить hot\_rod?

goal

can\_buy(kelly, hot\_rod). /\* can kelly buy hot\_rod? \*/

Четвертый пример рассматривал отношения «машина» и «грузовик»:

predicates

nondeterm car(symbol,real,integer,symbol,integer)

nondeterm truck(symbol,real,integer,symbol,integer)

clauses

car(chrysler,13000,3,red,12000).

car(ford,90000,4,gray,25000).

car(datsun,8000,1,red,30000).

truck(ford,80000,6,blue,8000).

truck(datsun,50000,5,orange,20000).

truck(toyota,25000,5,black,25000).

goal

/\* car(What1,What2,What3,red,What4). \*/ /\* list all red cars \*/

/\* truck(What1,What2,What3,black,What4). \*/ /\* list all black trucks \*/

/\* truck(ford,What1,What2,What3,What4). \*/ /\* list all ford trucks \*/

/\* car(ford,What1,What2,What3,What4). \*/ /\* list all ford cars \*/

car(chrysler,13000,3,red,12000). /\* does this car exist? \*/

Были заданы следующие вопросы:

* Перечислить красные машины

goal

car(What1,What2,What3,red,What4). /\* list all red cars \*/

* Перечислить черные грузовики

goal

truck(What1,What2,What3,black,What4). /\* list all black trucks \*/

* Перечислить грузовики марки ford

goal

truck(ford,What1,What2,What3,What4). /\* list all ford trucks \*/

* Перечислить машины марки ford

goal

car(ford,What1,What2,What3,What4). /\* list all ford cars \*/

* Существует ли указанная машина?

goal

car(chrysler,13000,3,red,12000). /\* does this car exist? \*/

Далее была составлена программа «Родственные отношения». За основу была взята программа из первого примера. Были добавлены отношения «брат», «сестра», «бабушка», «дедушка»:

domains

s=symbol

predicates

nondeterm parent(s,s)

female(s)

male(s)

nondeterm mother(s,s)

nondeterm father(s,s)

ancestor(s,s)

child(s,s)

nondeterm brother(s,s).

nondeterm sister(s,s).

nondeterm grand\_father(s,s).

nondeterm grand\_mother(s,s).

clauses

parent(pam,bob).

parent(tom,bob).

parent(tom,liz).

parent(bob,ann).

parent(bob,pat).

parent(pat,jim).

female(pam).

female(liz).

female(ann).

female(pat).

male(tom).

male(bob).

male(jim).

child(Y,X):- parent(X,Y).

mother(X,Y):- parent(X,Y),female(X).

father(X,Y):- parent(X,Y),male(X).

ancestor(X,Z):- parent(X,Z).

ancestor(X,Z):- parent(X,Y),ancestor(Y,Z).

brother(X,Y):- male(X), father(F,X), father(F,Y), mother(M,X), mother(M,Y), X<>Y.

sister(X,Y):- female(X), father(F,X), father(F,Y), mother(M,X), mother(M,Y), X<>Y.

grand\_father(X,Y):- male(X), parent(P,Y), parent(X,P).

grand\_mother(X,Y):- female(X), parent(P,Y), parent(X,P).

goal

grand\_father(tom,Who).

Следующим шагом была составлена программа, использующая отношения «нравится» и «может купить»:

predicates

likes(symbol,symbol).

for\_sale(symbol).

can\_afford(symbol,symbol).

can\_buy(symbol,symbol).

clauses

likes(ellen, onions).

likes(john, cheese).

likes(tom, pizza).

likes(tom, cheese).

likes(eric, potatoes).

for\_sale(onions).

for\_sale(cheese).

for\_sale(pizza).

can\_afford(tom,pizza).

can\_afford(eric, potatoes).

can\_buy(X,Y):- for\_sale(Y), likes(X,Y), can\_afford(X,Y).

goal

can\_buy(tom,pizza).

Наконец, мной была составлена собственная программа, использующая отношения «учитель», «ученик» и т. д.:

domains

s=symbol

predicates

nondeterm teacher(s).

nondeterm student(s).

teaches(s,s).

nondeterm wants\_to\_study(s,s).

nondeterm has\_students(s).

clauses

teacher(joe).

teacher(jim).

teacher(josh).

student(jorge).

student(joseph).

student(jack).

teaches(joe,astrology).

teaches(jim,cryptozoology).

teaches(josh,numerology).

wants\_to\_study(jorge,numerology).

wants\_to\_study(joseph,numerology).

wants\_to\_study(jack,cryptozoology).

has\_students(X):- teacher(X), student(Z), teaches(X,Y), wants\_to\_study(Z,Y).

goal

has\_students(jim).

# Выводы

Я научился создавать базовые программы на Прологе.