Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное

образовательное учреждение высшего образования

‹‹Уфимский университет науки и технологий››

Отчёт по лабораторной работе №2

по дисциплине «Логическое программирование»

Выполнил:

студент группы ПРО-331Б

Семенов Л.А.

Проверил:

старший преподаватель

Шакирзянов А.А.

Уфа – 2024

Ход работы.

Были добавлены к дереву родственных отношений из лабораторной работы №1, следующие предикаты для работы с динамический базой данных:

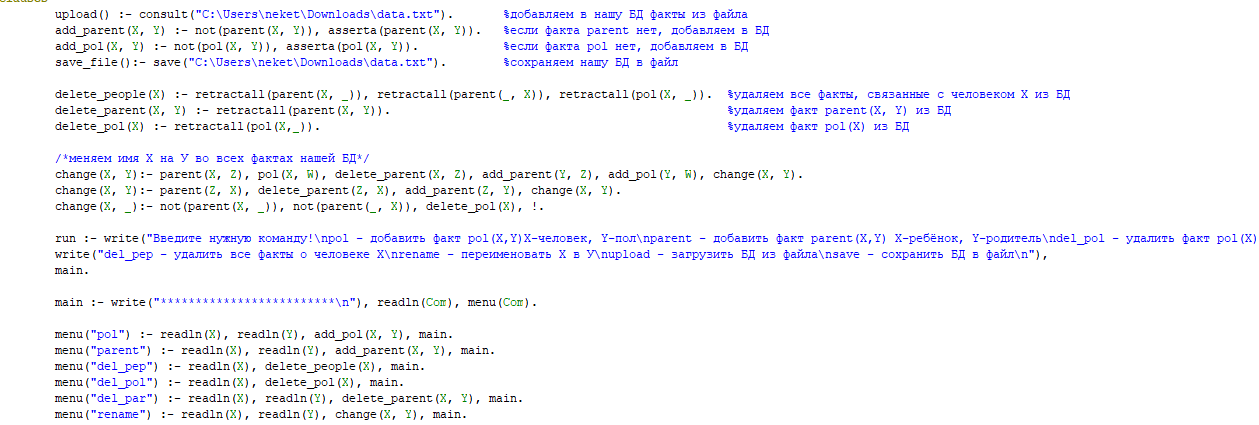


Рисунок 1. Предикаты для работы с динамической базой данных

Результаты работы программы:

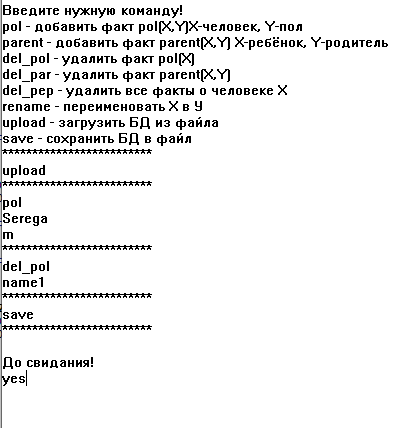


Рисунок 2. Сохранение данных в БД

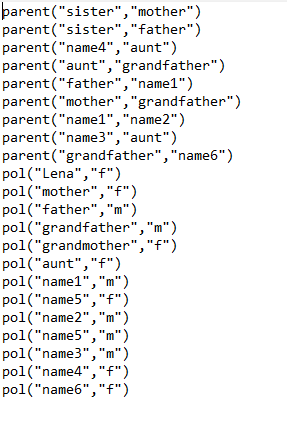


Рисунок 3 – Результат сохранения данных

Листинг программы.

database

parent(symbol, symbol)

pol(symbol, symbol)

domains

name = symbol

predicates

upload()

save\_file()

add\_parent(name, name)

add\_pol(name, name)

delete\_people(name)

delete\_parent(name, name)

delete\_pol(name)

nondeterm change(name, name)

nondeterm run

nondeterm main

nondeterm menu(name)

nondeterm man(name)

nondeterm woman(name)

nondeterm father(name)

nondeterm mother(name)

nondeterm son(name)

nondeterm daughter(name)

nondeterm brother(name, name)

nondeterm sister(name, name)

nondeterm grandfather(name, name)

nondeterm grandmother(name, name)

nondeterm uncle(name, name)

nondeterm aunt(name, name)

nondeterm nephew(name, name)

nondeterm niece(name, name)

nondeterm offspring(name, name)

nondeterm ancestor(name, name)

nondeterm generation(name, name)

nondeterm child(name)

nondeterm adult(name)

nondeterm grand(name, name).

nondeterm extreme(name, name).

nondeterm extreme\_(name, name).

clauses

upload() :- consult("C:\VIP52\BIN\WIN\32"). %добавляем в нашу БД факты из файла

add\_parent(X, Y) :- not(parent(X, Y)), asserta(parent(X, Y)). %если факта parent нет, добавляем в БД

add\_pol(X, Y) :- not(pol(X, Y)), asserta(pol(X, Y)). %если факта pol нет, добавляем в БД

save\_file():- save("C:\VIP52\BIN\WIN\32"). %сохраняем нашу БД в файл

delete\_people(X) :- retractall(parent(X, \_)), retractall(parent(\_, X)), retractall(pol(X, \_)). %удаляем все факты, связанные с человеком Х из БД

delete\_parent(X, Y) :- retractall(parent(X, Y)). %удаляем факт parent(X, Y) из БД

delete\_pol(X) :- retractall(pol(X,\_)). %удаляем факт pol(X) из БД

/\*меняем имя Х на У во всех фактах нашей БД\*/

change(X, Y):- parent(X, Z), pol(X, W), delete\_parent(X, Z), add\_parent(Y, Z), add\_pol(Y, W), change(X, Y).

change(X, Y):- parent(Z, X), delete\_parent(Z, X), add\_parent(Z, Y), change(X, Y).

change(X, \_):- not(parent(X, \_)), not(parent(\_, X)), delete\_pol(X), !.

run :- write("Введите нужную команду!\npol - добавить факт pol(X,Y)\nparent - добавить факт parent(X,Y)\ndel\_pol - удалить факт pol(X)\ndel\_par - удалить факт parent(X,Y)\n"),

write("del\_pep - удалить все факты о человеке X\nrename - переименовать Х в У\nupload - загрузить БД из файла\nsave - сохранить БД в файл\n"),

main.

main :- write("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n"), readln(Com), menu(Com).

menu("pol") :- readln(X), readln(Y), add\_pol(X, Y), main.

menu("parent") :- readln(X), readln(Y), add\_parent(X, Y), main.

menu("del\_pep") :- readln(X), delete\_people(X), main.

menu("del\_pol") :- readln(X), delete\_pol(X), main.

menu("del\_par") :- readln(X), readln(Y), delete\_parent(X, Y), main.

menu("rename") :- readln(X), readln(Y), change(X, Y), main.

menu("upload") :- upload(), main.

menu("save") :- save\_file(), main.

menu(\_) :- write("До свидания!\n"), !.

man(X) :- pol(X, m).

woman(X) :- pol(X, f).

father(X) :- parent(\_, X), pol(X, m).

mother(X) :- parent(\_, X), pol(X, f).

son(X) :- parent(X, \_), pol(X, m).

daughter(X) :- parent(X, \_), pol(X, f).

brother(X, Y) :- parent(X, Z), parent(Y, Z), man(X), X<>Y.

sister(X, Y) :- parent(X, Z), parent(Y, Z), woman(X), X<>Y.

grandfather(X, Y) :- man(X), parent(Y, Z), parent(Z, X), X<>Y.

grandmother(X, Y) :- woman(X), parent(Y, Z), parent(Z, X), X<>Y.

grand(X, Y) :- parent(Y, Z), parent(Z, X), X<>Y.

uncle(X, Y) :- parent(Y, Z), brother(X, Z).

aunt(X, Y) :- parent(Y, Z), sister(X, Z).

nephew(X, Y) :- man(X), parent(X, Z), parent(Z, W), parent(Y, W), Z<>Y.

niece(X, Y) :- woman(X), parent(X, Z), parent(Z, W), parent(Y, W), Z<>Y.

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

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

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

generation(X, Y) :- extreme(X, Y), X<>Y .

generation(X, Y) :- extreme\_(X, Y), X<>Y.

extreme(X, Y) :- parent(X, Z), parent(Y, Z); parent(X, W), parent(Y, A), extreme(W, A).

extreme\_(X, Y) :- parent(Z, X), parent(Z, Y).

extreme\_(X, Y) :- parent(W, X), parent(A, Y), extreme\_(W, A).

child(X) :- parent(X, \_), not(parent(\_, X)).

adult(X) :- parent(\_, X).

goal

run.

Вывод: в ходе лабораторной работы были изучены способы работы с динамическими базами данных в среде Visual Prolog. Реализована задача описания семейных отношений с использованием динамической базы данных, предусмотрены функции добавления, изменения и удаления данных, а также сохранения внесенных изменений. Полученные навыки могут быть применены для работы с другими типами динамических данных и расширения функциональности программ на языке Visual Prolog.