**Лабораторна робота №4**

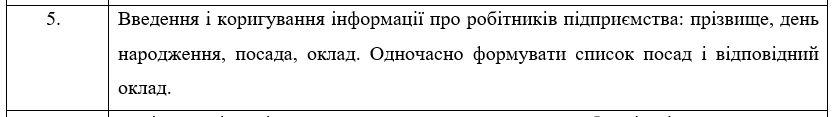
**Введення-виведення і робота з базою**

**Цілі роботи:**

* ознайомлення з предикатами введення-виведення у мові Prolog;
* вивчення та одержання практичних навичок з формуванням та змінні бази даних програми.

**Завдання:**

Завантажити з файлу список фактів, виконати дії відповідно завдання, а по закінченні сеансу роботи зберегти стан бази даних в файл.



Короткі теоретичні відомості

***Предикати введення-виведення***

Prolog-програма може читати дані з вхідного потоку і записувати в вихідний потік. У якості вхідного та вихідного потоку може бути файл або термінал (клавіатура і монітор відповідно). Дані, які введені з клавіатури і дані, які виведені на монітор, розглядаються як потік, якій має назву user. У будь який час існує один поточний вхідний і один вихідний потік. За замовчування поточним потоком є термінал, однак його можна змінити на файл, а після закінчення роботи з файлом відновити потік user.

Для зміни вхідного потоку використовується предикат

see(<імя файлу>)

Якщо вказаний файл існує, ціль see успішна, поточним вхідним потоком становиться файл і всі наступні операції читання здійснюються з файлу. Для відновлення потоку user необхідно застосувати предикат see(user) або предикат seen без параметрів.

***Предикати для роботи з базою даних програми***

У процесі виконання програми зміст бази даних може бути змінений. Для цього існують предикати (див. табл. 8.3) , які дозволяють додавати або вилучати твердження із бази даних.

|  |  |
| --- | --- |
| Таблиця 8.3 – Предикати для роботи з базою даних | |
| **Назва** | **Призначення** |
| assert(Х)  assertz(Х) | розміщує нове твердження X в кінці бази даних, або додає його до подібних за арністю та функтором тверджень за умови їх присутності |
| asserta(Х) | розміщує нове твердження X у початок бази даних |
| retract(Х) | вилучає задане твердження X із бази даних |
| abolish(Р/А) | вилучає всі подібні твердження з заданими функтором P і арністю A |

**Методичні вказівки**

***Особливості роботи з файлами***

Використання будь-якого предикату читання з файлу припускає, що файл існує. Щоб запобігти виникненню помилки у разі, якщо файл не існує, слід застосувати предикат exists\_file(F), де F – ім’я файлу. Предикат є вдалим, якщо файл F існує.

При читанні даних з файлу іноді необхідна перевірка кінця файлу. У мові Prolog існує атом end\_of\_file, що дає можливість виконати перевірку кінця файлу після кожного читання.

***Перевірка конкретизації змінної***

У програмі іноді важливо визначити стан конкретизації змінної виконання подальших дій. Предикат var(X) є вдалим, якщо X – неконкретизована змінна. Предикат nonvar(X) є вдалим, якщо X – конкретизована змінна. Приклад застосування цих предикатів представлений нижче.

**Рішення**

1. Аналіз

Задача, яку потрібно вирішити схожа на деякі класичні задачі з теорії баз даних. Тому є сенс оформити рішення у вигляді «бази даних».

Очевидно, що у найпростішому випадку нам потрібно дві таблиці:

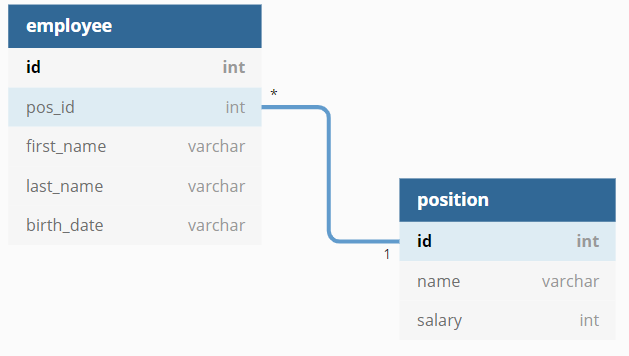
* employee (id, position\_id, first\_name, last\_name, birth\_date)
* position(id, name, salary)

Тобто id – внутрішній ключ, position\_id – зовнішній. Id у кожній таблиці має бути унікальним.

Для простоти дату народження будемо зберігати просто як текст у форматі «mm/dd/yyyy»

Для даної задачі будемо вважати, що оклад залежить лише від посади, тому ми можемо зберігати його у таблиці pisitions.

Дані усіх таблиць будемо зберігати у файлі db.txt



Реалізована програма підтримує такі можливості:

* зчитати дані з файлу
* зберегти дані у файл
* вивести співробітника по ідентифікатору (***при цьому, замість position\_id виводимо назву посади та оклад***)
* вивести усіх співробітників
* видалили співробітника по його ідентифікатору (*при цьому, якщо такого співробітника немає – має вивестись відповідне попередження*)
* додати співробітника (*якщо співробітник з таким ідентифікатором все існує – вивести повідомлення про помилку*)
* редагувати співробітника (*та вивести помилку, якщо такого співробітника немає*)
* вивести посаду по її ідентифікатору
* вивести всі посади
* видалити посаду по ідентифікатору (*вивести повідомлення про помилку, якщо немає такої посади або якщо є робітники що оймають дану посаду*)
* додати посаду *(вивести повідомлення про помилку, якщо вже є посада с таким ідентифікатором*)
* редагувати посаду (*вивести повідомлення про помилку, якщо немає посади с таким ідентифікатором*)

1. Текст програми

:-dynamic employee/5, position/3.

guide :- write('\* \* \* Prolog DB (v0.0.1) \* \* \*'), nl, nl,

   write('Available predicates:'), nl,

   write('?- empl(Id)'), nl,

   write('?- emplAll()'), nl,

   write('?- emplAdd(Id, PosId, FN, LN, Birthday)'), nl,

   write('?- emplEdit(Id, PosId, FM, LN, Birthday)'), nl,

   write('?- emplDel(Id)'), nl,

   write('?- pos(Id)'), nl,

   write('?- posAll()'), nl,

   write('?- posAdd(Id, Name, Salary) '), nl,

   write('?- posEdit(Id, Name, Salary)'), nl,

   write('?- posDel(Id)'), nl,

   write('?- guide()'), nl,

   write('?- save()'), nl.

% Run, Read, Save

run :- write('Loading data... '),

   (exists\_file('db.txt'); write('Can\'t load data. DB file not found'), nl, fail),

   see('db.txt'), read(X), readNext(X), seen(),

   nl, nl, guide, !.

readNext(end\_of\_file) :- done, nl, !.

readNext(X) :- assert(X), read(Y), readNext(Y).

save :- write('Saving... '),

   (exists\_file('db.txt'); write('Can\'t save data. DB file not found'), nl, fail),

   tell('db.txt'),

   findall(EmplId, employee(EmplId, \_, \_, \_, \_), EmplL), saveEmployees(EmplL),

   findall(PosId, position(PosId, \_, \_), PosL), savePositions(PosL),

   told(), done, nl, !.

saveEmployees(EmployeeIds) :- maplist(saveEmployee, EmployeeIds).

saveEmployee(Id) :- employee(Id, PosId, FN, LN, Birth),

   writeq(employee(Id, PosId, FN, LN, Birth)), write('.'), nl.

savePositions(PositionIds) :- maplist(savePosition, PositionIds).

savePosition(Id) :- position(Id, Name, Salary),

   writeq(position(Id, Name, Salary)), write('.'), nl.

% Employee

empl(Id) :- employee(Id, PosId, FN, LN, Birth),

   position(PosId, PosName, Salary),

   format('| ~a~t~5+ | ~a ~a~t~20+ | ~a~t~15+ | ~a~t~20+ | ~a~t~10+ |~n', [Id, FN, LN, Birth, PosName, Salary]).

emplDel(Id) :- (employee(Id, \_, \_, \_, \_); write('Employee not found')),

   retract(employee(Id, \_, \_, \_, \_)), !.

emplAdd(Id, PosId, FN, LN, Birth) :- (not(employee(Id, \_, \_, \_, \_)); write('Employee already exist. Use ?- emplEdit() instead'), nl, fail),

   (position(PosId, \_, \_); write('Position not found'), nl, fail),

   assert(employee(Id, PosId, FN, LN, Birth)),

   done, !.

emplEdit(Id, PosId, FN, LN, Birth) :- (employee(Id, \_, \_, \_, \_); write('Employee not found'), nl, fail),

   (position(PosId, \_, \_); write('Position not found'), nl, fail),

   emplDel(Id),

   assert(employee(Id, PosId, FN, LN, Birth)),

   done, !.

emplLine() :- format('|~`-t~71||~n').

emplAll() :- emplLine, emplHead, emplLine, empl(\_), emplLine, fail.

emplHead() :- format('| ~a~t~5+ | ~a~t~20+ | ~a~t~15+ | ~a~t~20+ | ~a~t~10+ |~n', ['Id', 'Name', 'Birthday', 'Position', 'Salary']).

% Position

pos(Id) :- position(Id, Name, Salary),

   format('| ~a~t~5+ | ~a~t~20+ | ~a~t~10+ |~n', [Id, Name, Salary]).

posDel(Id) :- (position(Id, \_, \_); write('Position not found'), fail),

   (not(employee(\_, Id, \_, \_, \_)); write('There are some employees on that position. Can\'t remove it.'), fail),

   retract(position(Id, \_, \_)), !.

posAdd(Id, Name, Salary) :- (not(position(Id, \_, \_)); write('Position already exist. Use ?- posEdit() instead'), nl, fail),

   assert(position(Id, Name, Salary)),

   done, !.

posEdit(Id, Name, Salary) :- ((position(Id, \_, \_); write('Position not found'), nl, fail)),

   retract(position(Id, \_, \_)),

   assert(position(Id, Name, Salary)),

   done, !.

posAll() :- posLine, posHead, posLine, pos(\_), posLine, fail.

posHead() :- format('| ~a~t~5+ | ~a~t~20+ | ~a~t~10+ |~n', ['Id', 'Name', 'Salary']).

posLine() :- format('|~`-t~36||~n').

% Other

done  :- write('Done!').

**./db.txt** (для лаконічності звіту представлена лише частина файлу, повний обсяг – 100 employees)

employee(1,7,'Oliver','Smith','6/21/1987').

employee(2,2,'George','Johnson','12/16/1987').

employee(3,4,'Harry','Williams','7/6/1997').

employee(33,5,'Tommy','Hill','6/13/1984').

employee(34,4,'Arlo','Scott','9/19/1998').

employee(35,5,'Reggie','Green','9/5/1981').

employee(36,6,'Edward','Adams','12/21/1985').

employee(37,6,'Jaxon','Baker','9/18/1985').

employee(38,2,'Adam','Gonzalez','6/10/1994').

employee(39,4,'Sebastian','Nelson','6/10/1980').

employee(40,2,'Rory','Carter','11/28/1990').

employee(41,3,'Riley','Mitchell','4/23/1997').

employee(42,5,'Dylan','Perez','2/22/1989').

employee(43,6,'Elijah','Roberts','11/26/1993').

employee(44,6,'Carter','Turner','1/21/1983').

employee(59,1,'Poppy','Murphy','9/4/1984').

employee(60,3,'Ella','Bailey','9/27/1982').

employee(61,3,'Lily','Rivera','4/14/1986').

employee(62,3,'Evie','Cooper','8/14/1993').

employee(63,4,'Isabella','Ross','10/13/1986').

employee(64,6,'Sophie','Cox','11/7/1984').

employee(65,3,'Ivy','Howard','2/20/1992').

employee(66,3,'Freya','Ward','8/9/1987').

employee(67,4,'Harper','Torres','11/30/1985').

employee(68,2,'Willow','Peterson','12/3/1996').

employee(69,3,'Charlotte','Gray','7/29/1990').

employee(70,4,'Jessica','Ramirez','1/9/1994').

employee(71,5,'Rosie','James','9/12/1997').

employee(72,5,'Daisy','Watson','4/27/1991').

employee(73,4,'Alice','Brooks','2/17/1980').

employee(74,3,'Elsie','Kelly','4/3/1998').

employee(91,6,'Maisie','Butler','1/20/1992').

employee(92,4,'Lucy','Simmons','11/10/1992').

employee(93,3,'Erin','Foster','8/24/1988').

employee(94,6,'Eliza','Gonzales','2/13/1986').

employee(95,4,'Ellie','Bryant ','1/15/1982').

employee(96,5,'Mila','Alexander','7/6/1986').

employee(97,3,'Imogen','Russell','8/27/1993').

employee(98,5,'Bella','Griffin ','5/10/1995').

employee(99,2,'Lola','Henderson','2/6/1996').

employee(100,3,'Molly','Hayes','7/17/1996').

position(1,'Web Developer',3000).

position(2,'IOS Developer',3100).

position(3,'QA',1500).

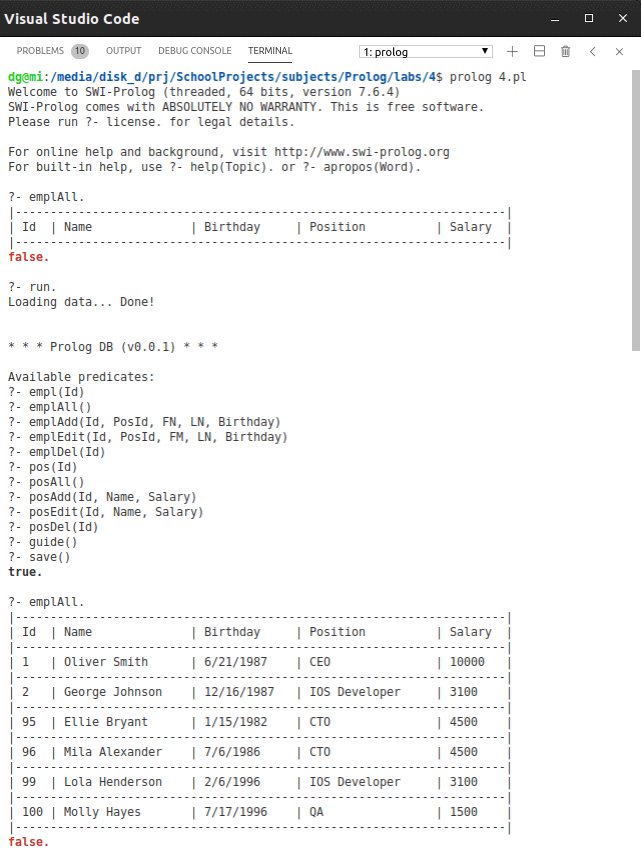
position(4,'HR',1350).

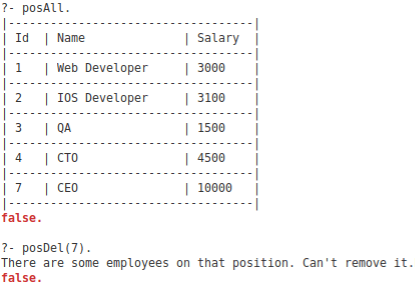
position(5,'PM',4500).

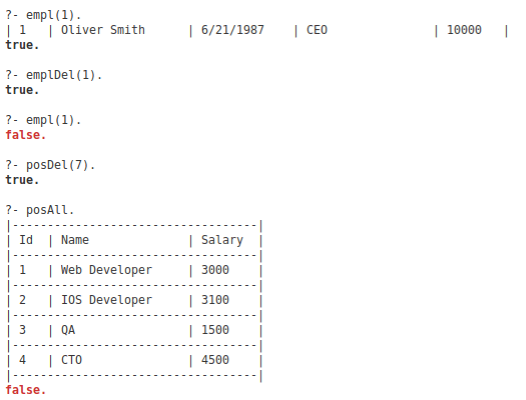
position(6,'CTO',4500).

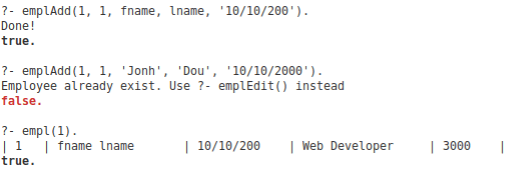
position(7,'CEO',10000).

1. Результат виконання програми

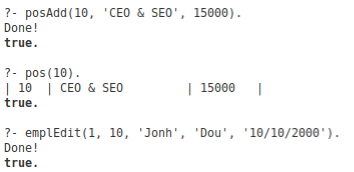


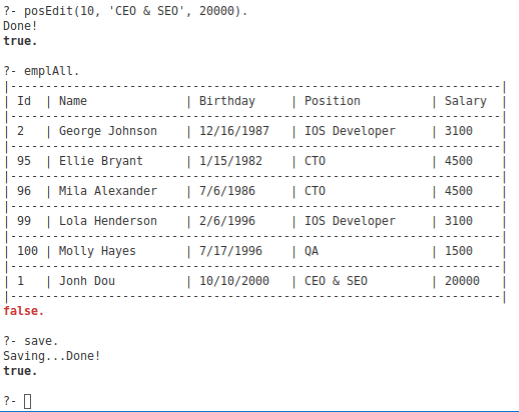








****

****

**Висновок:** на цій лабораторній роботі я ознайомився з предикатами введення-виведення у мові Prolog; вивчив та одержав практичні навички з формування та змінною бази даних програми.