Київський національний університет імені Тараса Шевченка

**Лабораторна робота**

**З навчального курсу «Парадигми та технології програмування»**

**«Розділ 6 (Prolog). Розробка простої експертної системи.  
Варіант «n»**

Виконав:

студент 3 курсу

факультету кібернетики

спеціальність «Комп’ютерні науки»

групи ТТП-32

Чебан Богдан Володимирович

**Київ 2024**

**Постановка задачі:**

1. Обов’язкова вимога: кількість об’єктів, що ідентифікуються, має бути не менше 8, їх назви мають починатись з літер, що відповідають варіанту (\*) – оцінка в межах 8 балів.

2. При використанні класифікаторів, узагальнюючих понять тощо - додатково 1 бал. Наприклад, гриби є пластинчастими, трубчастими, їстівними, неїстівними.

3. Підтримка вибору з трьох або більше альтернатив - додатково 1 бал. Наприклад, при визначенні одного з трьох можливих кольорів: червоного, синього чи зеленого; при визначенні одного з кількох (більше двох) можливих діапазонів даних тощо.



**Хід роботи:**

Для розробки моєї експертної системи на Prolog, яка допоможе ідентифікувати автомобілі за різними параметрами, я планую дотримуватись наступного підходу:

**Аналіз вимог**: Спочатку я визначу, що від системи очікують і які характеристики автомобілів потрібно враховувати. Це допоможе мені створити чітке розуміння цілей проєкту.

**Створення бази знань**: Я планую розробити детальну базу знань, яка буде включати інформацію про автомобілі, як-от марка, модель, рік випуску, потужність двигуна, тип кузова, та інші релевантні характеристики.

**Реалізація інтерактивності**: Я створю логіку для інтерактивного спілкування з користувачем, задаватиму питання і збиратиму відповіді, щоб зрозуміти потреби користувача і допомогти йому знайти найкращий автомобіль.

**Динамічна взаємодія**: Моя система буде адаптуватися до відповідей користувача, змінюючи набір питань або параметри пошуку, щоб забезпечити максимальну релевантність результатів.

**Управління невизначеністю**: Я інтегрую можливість вводити невизначені відповіді, наприклад "не знаю", так, щоб система могла пропонувати варіанти, навіть коли не всі параметри вказані чітко.

**Тестування та верифікація**: Я буду ретельно тестувати систему, щоб переконатися в її надійності та точності відповідей.

**Ітерація та поліпшення**: На основі отриманого зворотного зв'язку я буду постійно вдосконалювати систему, додаючи нові функції або вносячи корективи для покращення продуктивності та користувацького досвіду.

**Розробка користувацького інтерфейсу**: Хоча Prolog традиційно використовується для бекенду, я подбаю про те, щоб інтерфейс був дружній і зрозумілий для користувача.

**Код:**

% database

car(nissan\_leaf, electric, small, hatchback, 2019, 150, front).

car(nissan\_qashqai, petrol, medium, suv, 2021, 190, all).

car(nissan\_altima, petrol, medium, sedan, 2020, 180, front).

car(nissan\_xterra, diesel, large, suv, 2018, 210, all).

car(nissan\_juke, petrol, small, hatchback, 2019, 160, front).

car(nissan\_maxima, petrol, large, sedan, 2020, 300, front).

car(nissan\_murano, diesel, large, suv, 2021, 250, all).

car(nissan\_note, electric, small, hatchback, 2022, 130, front).

% q

ask(Question, Options, Answer) :-

writeln(Question),

(Options = [] -> read(Answer) % якщо немає заздалегідь визначених опцій, читаємо відповідь

; (member(Option, Options), format('~w ', [Option]), fail; nl, read(Answer))), % показує опції та читає відповідь

(Options = [] -> true; member(Answer, Options); Answer = 'xz').

% ідентифікація автомобіля з урахуванням усіх параметрів

identify\_car :-

ask('What is the power source?', [petrol, diesel, electric, 'xz'], Power),

ask('What is the size?', [small, medium, large, 'xz'], Size),

ask('What is the body type?', [sedan, hatchback, suv, 'xz'], BodyType),

ask('What is the year of manufacture?', [], Year),

ask('What is the horsepower?', [], Horsepower),

ask('What is the drive type?', [front, all, 'xz'], DriveType),

findall(Name, car(Name, Power, Size, BodyType, Year, Horsepower, DriveType), Matches),

(Matches = [] -> writeln('No car matches those criteria.')

; writeln('Matching cars:'), print\_matches(Matches)).

print\_matches([]).

print\_matches([Name|T]) :-

car(Name, Power, Size, BodyType, Year, Horsepower, DriveType),

format('~w: ~w, ~w, ~w, year ~w, ~w HP, drive: ~w~n', [Name, Power, Size, BodyType, Year, Horsepower, DriveType]),

print\_matches(T).

% main

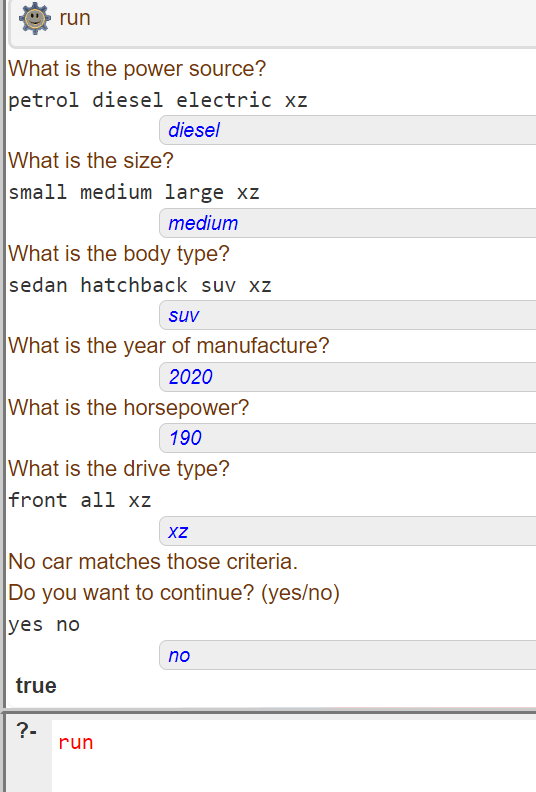
run :-

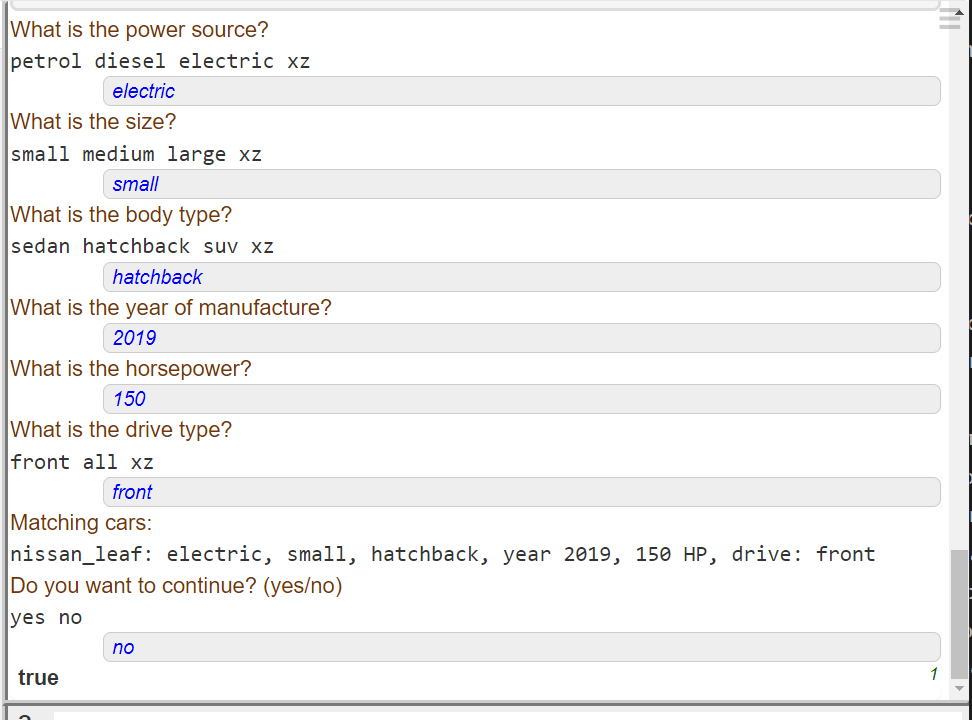
repeat,

identify\_car,

ask('Do you want to continue? (yes/no)', [yes, no], Continue),

Continue = no, !.





Цей код Prolog представляє собою експертну систему для ідентифікації автомобілів за різними параметрами:

**База даних car**: Включає записи про автомобілі з детальною інформацією, такою як марка, тип двигуна, розмір, тип кузова, рік випуску, потужність двигуна та тип приводу.

**Функція ask**: Взаємодіє з користувачем, запитуючи інформацію та читаючи відповіді. Якщо для питання визначені певні опції, вона виводить ці опції; в іншому випадку дозволяє ввести будь-яку відповідь. Опція "xz" використовується як плейсхолдер для невизначеності.

**Процедура identify\_car**: Веде діалог з користувачем, запитуючи про різні характеристики автомобіля (наприклад, джерело живлення, розмір, тип кузова тощо), і збирає ці дані для ідентифікації автомобілів, які відповідають цим критеріям.

**Функція print\_matches**: Виводить деталі знайдених автомобілів, які відповідають заданим критеріям.

**Основна програма run**: Керує циклом виконання, дозволяючи користувачу продовжувати ідентифікацію автомобілів або завершити роботу програми, вводячи "yes" або "no".

**Висновок:**

Я навчився розробляти інтерактивну експертну систему на Prolog, що ідентифікує автомобілі за різними параметрами. Цей процес включав розуміння того, як структурувати базу знань, використовуючи факти та правила в Prolog, щоб зберігати та обробляти інформацію про автомобілі. Я навчився створювати динамічні запити до користувача через процедуру **ask**, що дозволяє отримувати специфічні дані та пристосовуватися до відповідей користувача.

Також я зрозумів, як важливо обробляти невизначеність і забезпечувати гнучкість у відповідях користувачів, дозволяючи їм виражати невизначеність у своїх відповідях. Рекурсивні функції, як **print\_matches**, демонструють силу Prolog у обробці списків і здійсненні ітеративних завдань.

Завершення вивчення циклічної логіки через **run**, яка керує потоком виконання програми, надала мені глибше розуміння інтерактивних програмних патернів у Prolog. Цей досвід покращив мої навички в розробці логічних систем та зрозумінні інтерактивної програмної логіки.