

Міністерство освіти і науки України  
Львівський національний університет імені Івана Франка

Звіт  
про виконання лабораторної роботи №10  
на тему:  
**«Розв'язування логічних задач за допомогою Prolog»**

Виконала:  
Студентка групи Фес-32  
Філь Дарина

Львів - 2025

### ***Мета роботи:***

Дослідження та практичне застосування мови Prolog для розв'язування логічних задач.

### ***Теоретичні відомості:***

Prolog (Programming in Logic) — це декларативна мова програмування, яка використовується для розв'язання задач, що потребують логічного висновку. Вона широко застосовується в штучному інтелекті, обробці природної мови, автоматизованому доведенні теорем та експертних системах.

Основні компоненти програми на Prolog:

1. Факти — відомі твердження про об'єкти та їх властивості.
2. Правила — логічні твердження, що встановлюють залежності між фактами.
3. Запити — питання до бази знань, на які Prolog намагається знайти відповідь.

#### **1. Уніфікація**

Уніфікація дозволяє зв'язати змінні з конкретними значеннями, щоб знайти відповідність між запитом і фактом чи правилом.

У нашому випадку, запит:

`suspicious_transaction(T)`. має знайти транзакції, які є підозрілими, а тому потрібно зв'язати змінну `T` з конкретними транзакціями, які відповідають певним умовам.

Процес уніфікації працює так:

1. Спочатку Prolog перевірить правила та факти для підозрілих транзакцій.

Наприклад, для правила:

`suspicious_transaction(T) :- transaction(T, _, Amount, night), Amount > 10000.`

2. Prolog намагається уніфікувати запит `suspicious_transaction(T)` з цим правилом. Він перевірить кожну транзакцію, яка відбувається вночі і має суму більшу за 10000.

- Транзакція 1: (1, john, 5000, night) — сума менша за 10000, не підходить.
  - Транзакція 2: (2, mary, 20000, day) — сума більша, але вона не відбулася вночі.
  - Транзакція 3: (3, john, 300, night) — сума менша за 10000.
  - Транзакція 4: (4, paul, 15000, night) — сума більша за 10000, це підозріло.
3. Уніфікація знайде підозрілу транзакцію `transaction(4, paul, 15000, night)` і прив'яже змінну `T` до значення 4.

## 2.Резолюція:

Резолюція в Prolog — це процес, коли ми комбінуємо факти та правила, щоб зробити висновки.

Вона працює наступним чином:

1. Перевіряються правила. Кожне правило має певні умови.
2. Пошук підстав для цих умов.
3. Якщо умови правил виконуються (підстави знайдені), програма робить логічний висновок, що об'єднує факти та правила в одне ціле.

У резолюції важливим є процес уніфікації, який дозволяє порівнювати вирази і знаходити, чи можна їх «підставити» (уніфікувати) так, щоб вони стали однаковими.

### *Хід роботи:*

1. Написати програму на мові prolog, яка визначає підозрілі платіжні транзакції за наступними параметрами: Користувач, Сума, Країна, де відбулась, транзакція, Час, Тип.

Задання фактів:

```
suspicious_account(jane).  
suspicious_account(ivan).  
  
suspicious_country('NorthKorea').  
suspicious_country('China').  
suspicious_country('Afghanistan').  
suspicious_country('Belarus').
```

Задання правил:

```
suspicious_transaction(T) :-  
    transaction(T, Account, Amount, Time, Country, Type),  
    (  
        suspicious_account(Account);  
        (Amount > 20000, Time = night);  
        suspicious_country(Country);  
        suspicious_type(Type)  
    ).
```

2. Використовуючи відповідні бібліотеки, зчитати дані з csv файлу, який містить відповідні транзакції та реалізувати факти та правила.

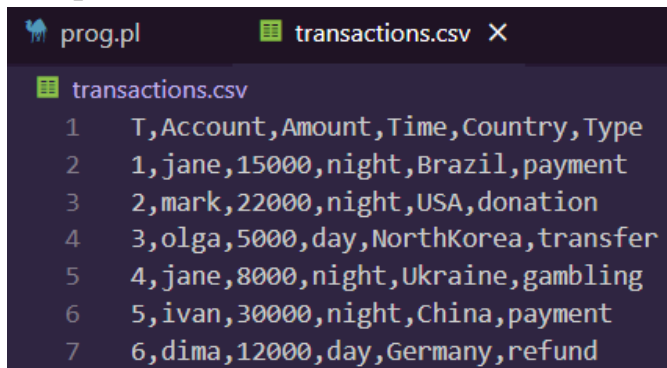
Імпорт бібліотеки csv:

```
:- use_module(library(csv)).
```

Функції зчитування даних з csv і перетворення їх у факти:

```
read_transactions(File) :-  
    csv_read_file(File, Rows, [functor(row)]),  
    exclude(is_header_row, Rows, DataRows),  
    maplist(convert_to_transaction, DataRows).  
  
is_header_row(row('T', _, _, _, _, _)).  
  
convert_to_transaction(row(T, Account, AmountRaw, Time, Country, Type)) :-  
    (    number(AmountRaw)  
    -> Amount = AmountRaw  
    ;    atom_number(AmountRaw, Amount)  
    ),  
    assertz(transaction(T, Account, Amount, Time, Country, Type)).
```

csv файл:



```
prog.pl transactions.csv X
transactions.csv
1 T,Account,Amount,Time,Country,Type
2 1,jane,15000,night,Brazil,payment
3 2,mark,22000,night,USA,donation
4 3,olga,5000,day,NorthKorea,transfer
5 4,jane,8000,night,Ukraine,gambling
6 5,ivan,30000,night,China,payment
7 6,dima,12000,day,Germany,refund
```

Вивід:

```
?- working_directory(_, 'E:/III/Lab10').
true.

?- consult('prog.pl').
true.

?- read_transactions('transactions.csv').
true.

?- suspicious_transaction(T).
T = 3 ;
T = 1 ;
T = 2 ;
T = 3 ;
T = 4 ;
T = 4 ;
T = 5 ;
T = 5 ;
T = 5 ;
false.
```

3. У висновку запропонувати інші підходи та вказати, які недоліки та переваги має цей підхід.

Prolog має низку переваг для реалізації подібних задач. Передусім, логіку в ньому можна формулювати досить природно завдяки використанню правил з оператором `:-`, що робить опис умов зрозумілим і лаконічним. Крім того, вбудований механізм уніфікації та логічного висновку дозволяє автоматично будувати висновки на основі заданих фактів. Особливо зручно, що нові правила можна легко додавати до існуючої програми без необхідності змінювати її основну структуру, що робить систему гнучкою та зручною для розширення.

Втім, у Prolog є й певні обмеження. Наприклад, робота з великими обсягами даних менш ефективна порівняно з інструментами на кшталт Python або SQL. До того ж, для повноцінної роботи в середовищі Prolog необхідне логічне мислення, а відсутність типізації та базової обробки помилок може ускладнити реалізацію складніших рішень.

Як альтернативу можна розглядати Python разом з бібліотекою Pandas, що дозволяє швидко зчитувати CSV-файли, зручно фільтрувати дані та вести облік транзакцій із гнучкими умовами.

### ***Висновок:***

У процесі виконання лабораторної роботи я створила програму, яка обробляє транзакції та перевіряє їх на підозрілість відповідно до заданих умов. Також я ознайомила з поняттями уніфікації та резолюції в Prolog і використала їх для формулювання логіки виводу. Це дало змогу краще зрозуміти принципи логічного програмування та переваги використання Prolog для задач, де важлива гнучкість у формулюванні правил.

### **Додаток**

#### **prog.pl**

```
:- use_module(library(csv)).
```

```
suspicious_account(jane).
```

```
suspicious_account(ivan).
```

```
suspicious_country('NorthKorea').
```

```
suspicious_country('China').
```

```
suspicious_country('Afghanistan').
```

```
suspicious_country('Belarus').
```

```
suspicious_type(gambling).
```

```
suspicious_transaction(T) :-
```

```
    transaction(T, Account, Amount, Time, Country, Type),
```

```
    (
```

```
        suspicious_account(Account);
```

```
        (Amount > 20000, Time = night);
```

```
        suspicious_country(Country);
```

```
        suspicious_type(Type)
```

```
    ).
```

```
read_transactions(File) :-
```

```
    csv_read_file(File, Rows, [functor(row)]),
```

```
    exclude(is_header_row, Rows, DataRows),
```

```
maplist(convert_to_transaction, DataRows).
```

```
is_header_row(row('T', _, _, _, _, _)).
```

```
convert_to_transaction(row(T, Account, AmountRaw, Time, Country, Type)) :-
```

```
  ( number(AmountRaw)
```

```
  -> Amount = AmountRaw
```

```
  ; atom_number(AmountRaw, Amount)
```

```
  ),
```

```
  assertz(transaction(T, Account, Amount, Time, Country, Type)).
```

### **transactions.csv**

```
T,Account,Amount,Time,Country,Type
```

```
1,jane,15000,night,Brazil,payment
```

```
2,mark,22000,night,USA,donation
```

```
3,olga,5000,day,NorthKorea,transfer
```

```
4,jane,8000,night,Ukraine,gambling
```

```
5,ivan,30000,night,China,payment
```

```
6,dima,12000,day,Germany,refund
```