**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**
**КРИВОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**
**ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ**

# З В І Т

## Лабораторна робота №3
## з дисципліни
## «Компьютерні системи штучного інтелекту»

Виконавець:

студент групи КІ-22м Косей М.П.

Керівник:

викладач Саяпін В.Г.

2023

# Лабораторна робота №3

**Тема: Моделювання штучного логічного виведення**

**Мета: Одержати уміння та закріпити навички роботи з певним інтегрованим середовищем для логічного програмування мовою PROLOG**

**Завдання: Скласти структурну схему власного родоводу та реалізувати її на мові програмування Prolog**

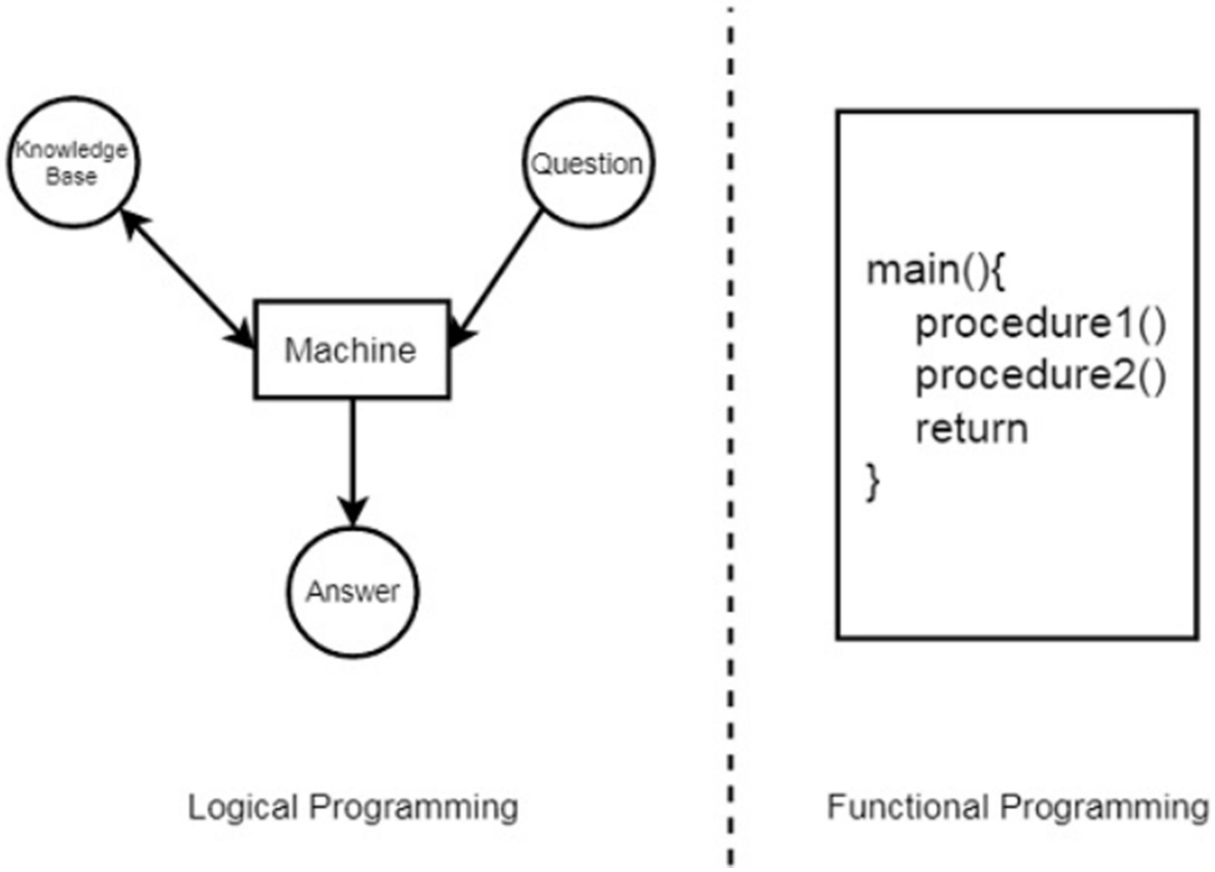
## ХІД РОБОТИ

**1) Ознайомитись з теоретичними відомостями до лабораторної роботи.**

## PROLOG

PROLOG - це мова логічного програмування, що базується на формальній системі, яка називається логікою першого порядку. Ця мова використовує набір правил та фактів для визначення взаємозв'язків та обмежень між об'єктами. Це робить PROLOG ідеальною мовою для завдань, таких як обробка природної мови, експертні системи та автоматизоване мислення.

Логічне програмування - це підхід до програмування, який базується на символьній логіці. У логічному програмуванні програми описуються умовами, які повинні бути виконані для досягнення певного результату. Це дозволяє програмістам визначати не тільки те, що програма повинна зробити, але й те, як програма повинна діяти у різних умовах.



Основи мови програмування PROLOG включають наступні концепції:

**Атоми**

У PROLOG атоми використовуються для представлення констант, таких як імена, значення або мітки.

Вони починаються з маленької літери, можуть складатися з літер, цифр і символу підкреслення ‘_’ і бути рядками символів в лапках

Приклади атомів в PROLOG:
***john, red, 'Hello, world!', b59, b_59, b_59AB, b_x25.***

**Змінні**

Змінні в PROLOG використовуються для представлення невідомих значень і позначаються великими літерами або символом підкреслення ‘_’на початку їх назв.

Можуть складатися з літер, цифр і символу підкреслення ‘_’.

Приклади змінних в PROLOG:
***X, Sum, Memer_name, Student_list, Shoppinglist, _a50, _15.***

**Факти(аксіоми, постулати)** - це базові знання, які можуть бути використані у програмі.

Це твердження, які є безумовно правдивими.

У PROLOG факти використовуються для визначення взаємозв'язків між об'єктами. Кожен факт складається зі співвідношення (***relation***) та його аргументів (***object1, object2, …***), які є об'єктами, що беруть участь у взаємозв'язку.

Синтаксис для фактів в PROLOG виглядає наступним чином:
***relation(object1, object2, ...).***

Назви співвідношень починаються з маленьких літер, можуть складатися з літер, цифр і символу підкреслення ‘_’.

Приклади фактів:
***cat(tom).***
***loves_to_eat(ivan,meat).***
***of_color(hair,black).***
***loves_to_play_games(alex).***

**Правила(теореми)** - це умови, які потрібно виконати для отримання результату.

Правило це предикат який є правдивим, якщо інші предикати від яких залежить правило теж правдиві.

Правило складається з назви правила (**rule_name**), його аргументів (***object1, object2, …***), імплікації «якщо» (**:-**) та фактів/правил(***fact/rule(object1, object2, ...)***) від правдивості яких залежить правило.

Синтаксис для правил в PROLOG виглядає наступним чином:
***rule_name(object1, object2, ...) :- fact/rule(object1, object2, ...).***

Факти/правила((***fact/rule(object1, object2, ...)***) можуть комбінуватись між собою за допомогою логічних операцій:

(**,**) - кон'юнкція або логічне «І»;
(**;**) - діз'юнкція або логічне «АБО»;
(**not()**) – заперечення або логічне «НЕ».

Приклади правил:
***happy(lili) :- dances(lili) ; songs(lili).***
***hungry(tom) :- search_for_food(tom).***
***friends(jack, bili) :- lovesCricket(jack), lovesCricket(bili).***

***goToPlay(ryan) :- not(opened(school)), free(ryan).***

**Терми** - це об'єкти, які можуть містити дані та використовуватися у програмі. Терми можуть бути фактами, правилами, змінними та константами.

**Запити** - це запити до бази даних у програмі.

Запит складається зі знаку питання (**?-**) та факту/правила (***fact/rule(object1, object2, ...)***) для якого потрібно виконати пошук у базі знань.

Синтаксис для правил в PROLOG виглядає наступним чином:

***?- fact/rule(object1, object2, ...).***

Приклади запитів:

***?- good(life).***

***?- loves(mary,X).***

***?- good(X).***

**2) Скласти структурну схему власного родоводу та реалізувати її на мові програмування Prolog.**

Сімейне дерево показано у вигляді орієнтованого графу, в вершинах графу - люди.
Ребра графа показують зв'язок між батьками та дітьми.
Синім кольором позначені – чоловіки, рожевим – жінки.

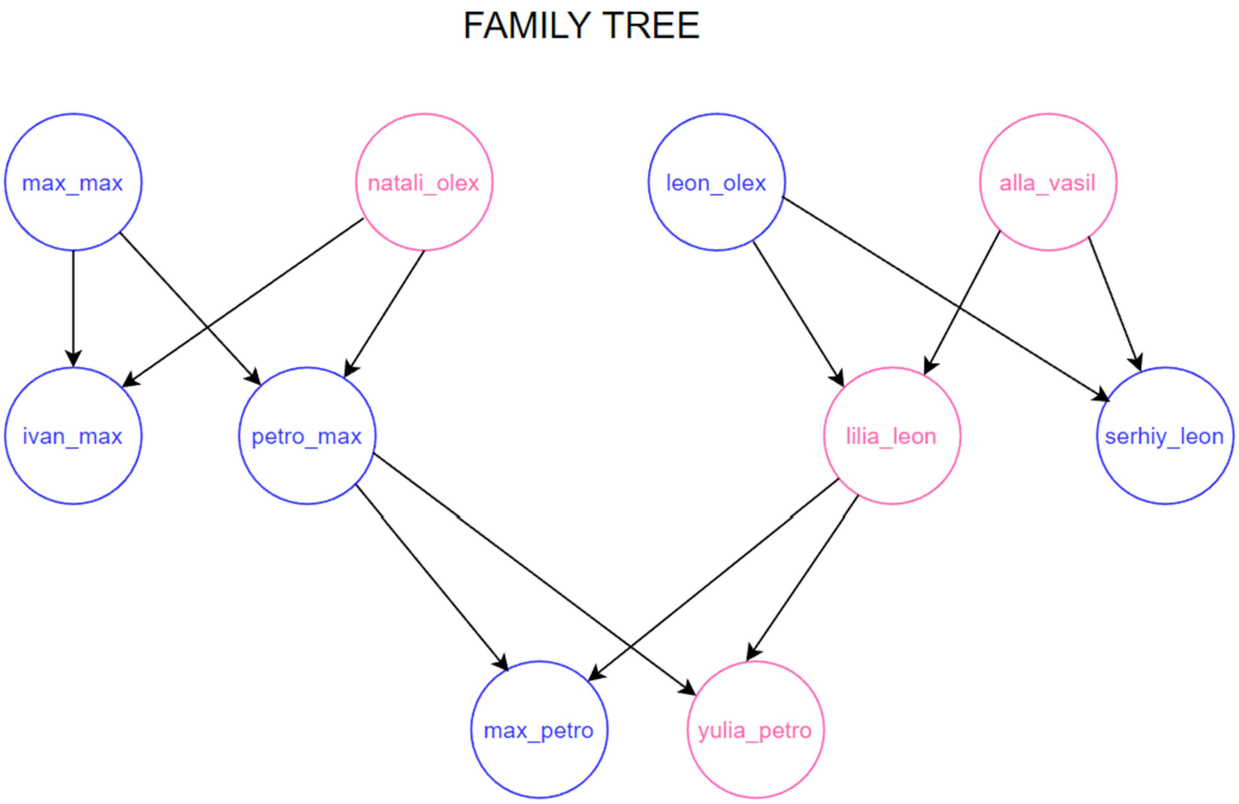


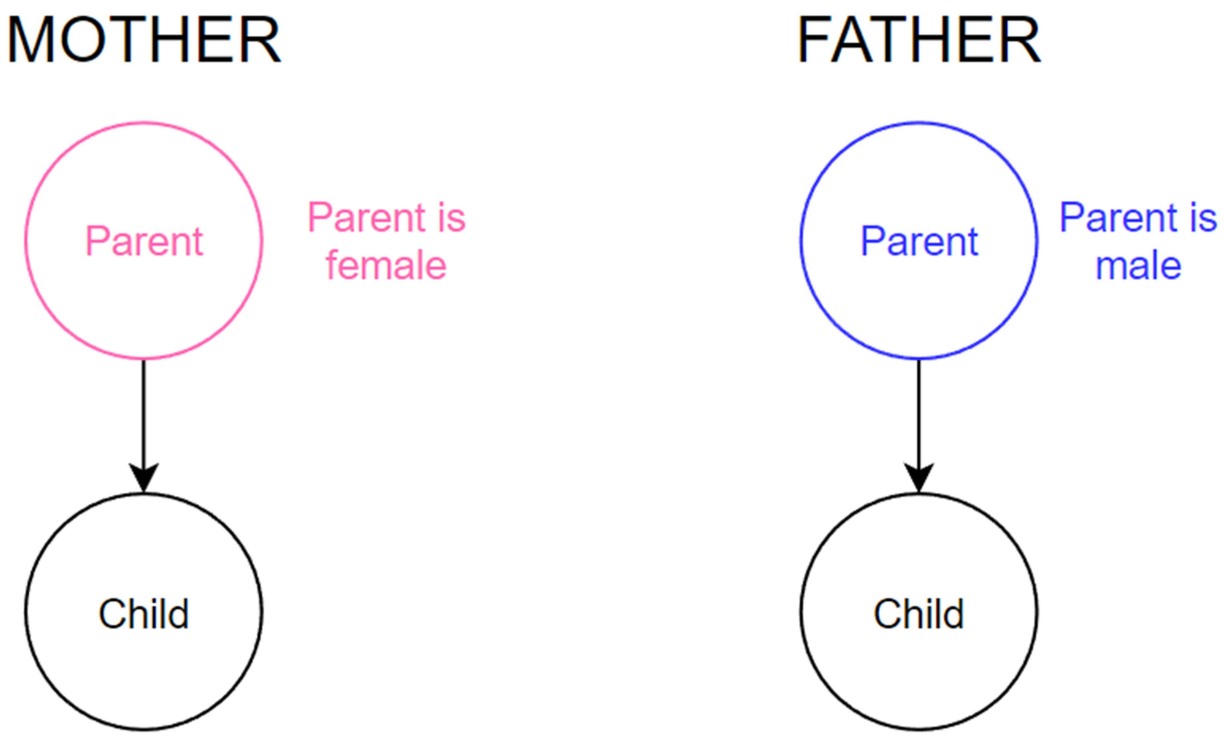

Схема власного родоводу реалізується за допомогою **SWI-Prolog**.
База даних фактів для сімейного дерева:

SWISH File Edit Examples Help

Program

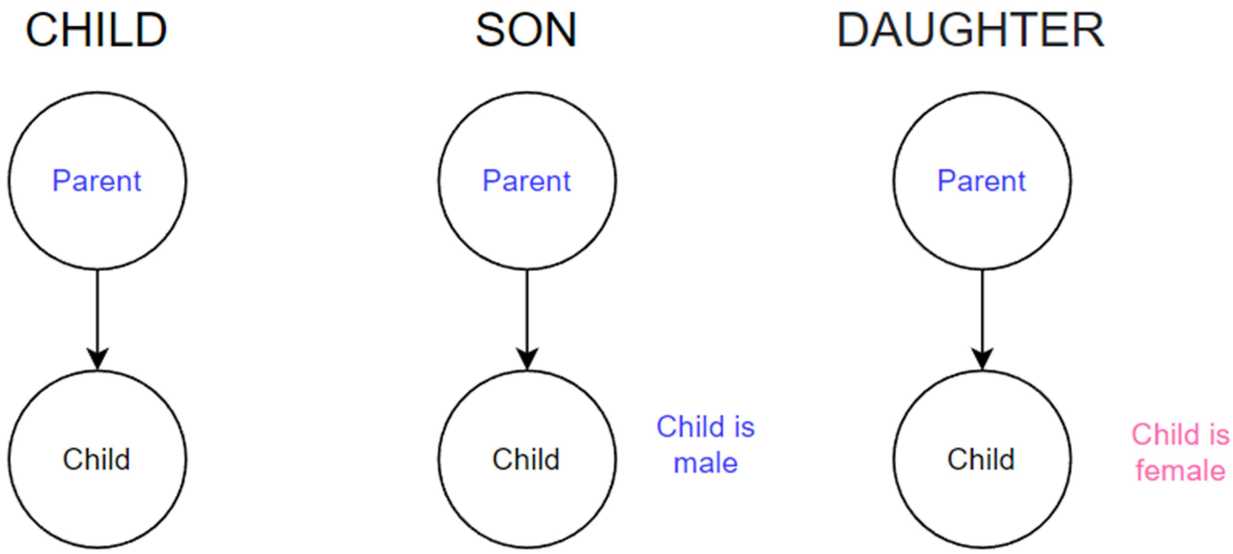
```
% Facts
male(max_max).
male(ivan_max).
male(petro_max).
male(leon_olex).
male(serhiy_leon).
male(max_petro).

female(natali_olex).
female(alla_vasil).
female(lilia_leon).
female(yulia_petro).

parent(max_max, ivan_max).
parent(max_max, petro_max).
parent(natali_olex, ivan_max).
parent(natali_olex, petro_max).

parent(leon_olex, lilia_leon).
parent(leon_olex, serhiy_leon).
parent(alla_vasil, lilia_leon).
parent(alla_vasil, serhiy_leon).

parent(petro_max, max_petro).
parent(petro_max, yulia_petro).
parent(lilia_leon, max_petro).
parent(lilia_leon, yulia_petro).

```

Правила для батьків



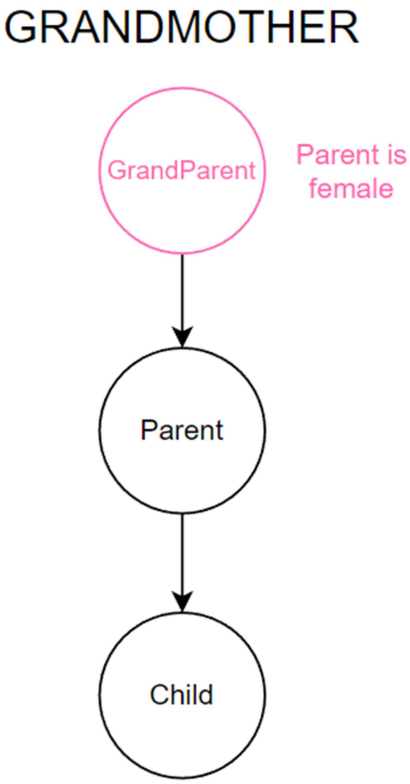

```
% Rules
father(Parent, Child) :- parent(Parent, Child), male(Parent).
mother(Parent, Child) :- parent(Parent, Child), female(Parent).
```

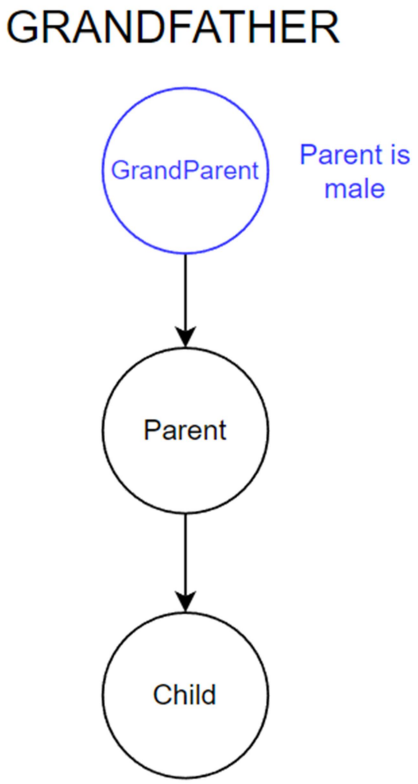
Правила для дітей




```
child(Child, Parent) :- parent(Parent, Child).
son(Child,Parent) :- child(Child, Parent), male(Child).
daugther(Child,Parent) :- child(Child, Parent), female(Child).
```

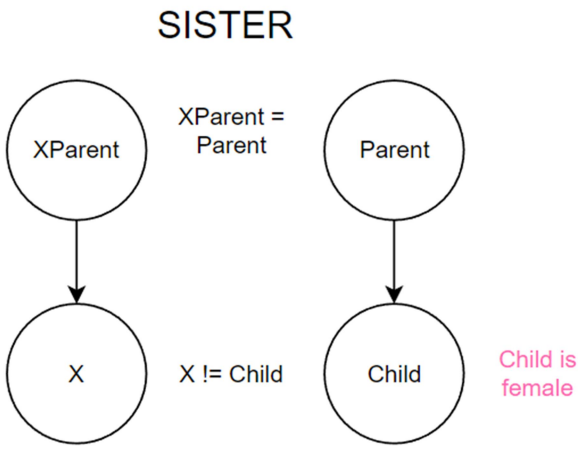
Правила для дідів, бабусь, онук та онуків.

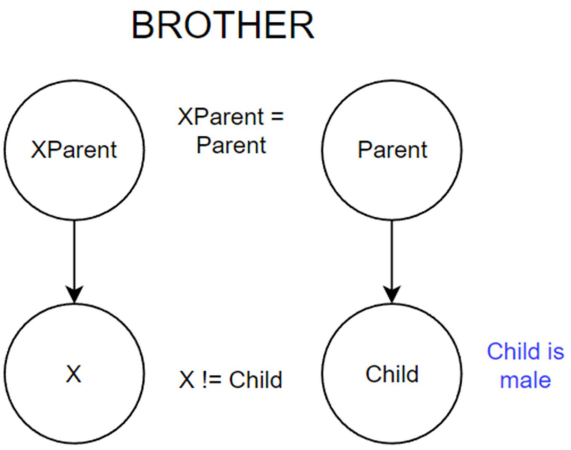



```
grandfather(GrandParent, Child) :- parent(GrandParent, Parent), parent(Parent, Child), male(GrandParent).
grandmother(GrandParent, Child) :- parent(GrandParent, Parent), parent(Parent, Child), female(GrandParent).
grandson(Child, GrandParent) :- parent(GrandParent, Parent), parent(Parent, Child), male(Child).
granddaugther(Child, GrandParent) :- parent(GrandParent, Parent), parent(Parent, Child), female(Child).
```

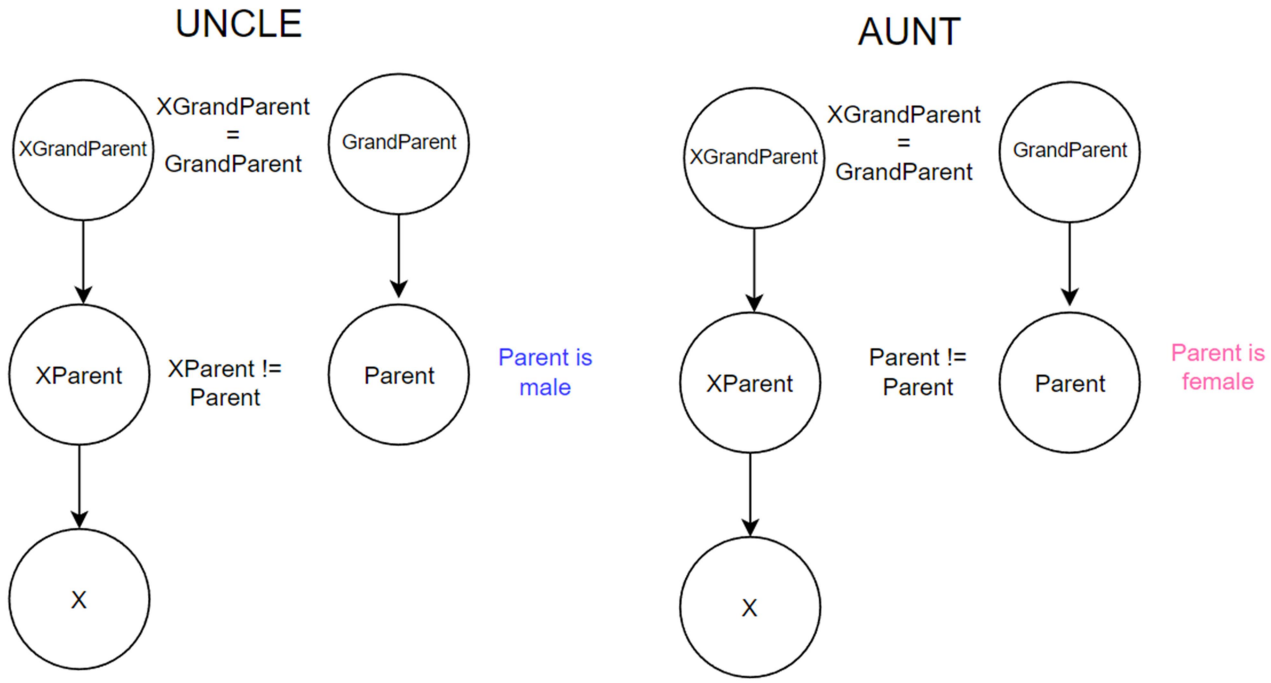
Правила для сестер і братів




```
siter(Child,X) :- parent(XParent, X), parent(Parent, Child), male(Child), XParent=Parent, not(Child = X).
brother(Child,X) :- parent(XParent, X), parent(Parent, Child), female(Child), XParent=Parent, not(Child = X).
```

Правила для дядьок та тіток




```
uncle(Uncle,X) :- parent(XParent, X), parent(XGrandParent, XParent), parent(GrandParent, Uncle), male(Uncle), XGrandParent=GrandParent, not(Uncle=XParent).
aunt(Aunt,X) :- parent(XParent, X), parent(XGrandParent, XParent), parent(GrandParent, Aunt), female(Aunt), XGrandParent=GrandParent, not(Aunt=XParent).
```

Складемо запити для людей із сімейного дерева:

- знайдемо усіх дітей та їх батьків

*child*(Child, Parent).

| Child | Parent | |
|---|---|---|
| ivan_max | max_max | 1 |
| petro_max | max_max | 2 |
| ivan_max | natali_olex | 3 |
| petro_max | natali_olex | 4 |
| lilia_leon | leon_olex | 5 |
| serhiy_leon | leon_olex | 6 |
| lilia_leon | alla_vasil | 7 |
| serhiy_leon | alla_vasil | 8 |
| max_petro | petro_max | 9 |
| yulia_petro | petro_max | 10 |
| max_petro | lilia_leon | 11 |
| yulia_petro | lilia_leon | 12 |

```
?- child(Child, Parent).
```

- знайдемо усіх онуків, онучок та їх дідів та бабусь

*grandfather*(GrandFather, GrandChild); *grandmother*(GrandMother, GrandChild).

| GrandFather | GrandChild | GrandMother | |
|---|---|---|---|
| max_max | max_petro | GrandMother | 1 |
| max_max | yulia_petro | GrandMother | 2 |
| leon_olex | max_petro | GrandMother | 3 |
| leon_olex | yulia_petro | GrandMother | 4 |
| GrandFather | max_petro | natali_olex | 5 |
| GrandFather | yulia_petro | natali_olex | 6 |
| GrandFather | max_petro | alla_vasil | 7 |
| GrandFather | yulia_petro | alla_vasil | 8 |

```
?- grandfather(GrandFather, GrandChild); grandmother(GrandMother, GrandChild).
```

- знайдемо дідусів та бабусь для **max_petro**

| GrandFather | GrandMother | |
|---|---|---|
| max_max | GrandMother | 1 |
| leon_olex | GrandMother | 2 |
| GrandFather | natali_olex | 3 |
| GrandFather | alla_vasil | 4 |

```
?- grandfather(GrandFather, max_petro); grandmother(GrandMother, max_petro).
```

- знайдемо дядьок та тіток для **max_petro**

| Uncle | Aunt | |
|---|---|---|
| ivan_max | Aunt | 1 |
| ivan_max | Aunt | 2 |
| serhiy_leon | Aunt | 3 |
| serhiy_leon | Aunt | 4 |

```
?- uncle(Uncle, max_petro); aunt(Aunt, max_petro).
```

## ВИСНОВКИ

**В результаті виконаної лабораторної роботи складена структурна схема власного родоводу та реалізовано на мові програмування Prolog.**

**Усі матеріали викладенні у репозіторії GitHub, за посиланням** https://github.com/Max11mus/Artifition-Intelect-Lab3**.**