Київський національний університет імені Тараса Шевченка
Факультет інформаційних технологій

Кафедра програмних систем і технологій

ЗВІТ

з лабораторної роботи № 1

Тема: “Автоматизація бізнес-процесів архітектурного 3Д-дизайну”

Дисципліна «Об’єктно-орієнтоване конструювання програм»

Підготував:
студент гр. ІПЗ-23(1)
Шевченко Андрій Віталійович
Дата : 18.05.22

Перевірила:
Зубик Л.В

Київ – 2022

Тема: Об’єктно-орієнтоване програмування у різних мовах програмування (COOL&CLIPS).

Мета роботи: ознайомитися з об'єктно орієнтованим підходом у різних мовах програмування, зокрема COOL&CLIPS.

Умова

1. Розробити програмний код для вирішення поставленої задачі. Середовище для розроблення – CLIPS. Мова програмування – COOL. Дозволяється використання онлайн-сервісів у процесі розроблення.

Варіант 11.

Жили-були три гноми: Ог, Гог і Магог. Були вони одружені, їх дружини звалися Ола, Гала і Мела. Кожна пара мала сина, їх імена – Ох, Ах, Ех. Відомо про них також таке: - Магога – не чоловік Гали і не батько Аха; - Мела – не жінка Ога і не мати Еха; - якщо батько Еха – Ог або Магог, то Гала – мати Оха; - якщо Гала – не жінка Ога, то Ола – не мати Еха. Назвіть імена кожного з членів трьох сімей гномів.

Хід роботи:

Для початку потрібно виділити сутності з якими ми будемо працювати, та початково їх проініціалізувати. Згідно умові ми можемо виділити такі сутності: Ог, Гог, Магог, Ох, Ах, Ех, Ола, Гала, Мела, можна відобразити це в програмному коді.

```
(deftemplate family
   (slot name (type SYMBOL) (default Unk))
   (slot wife (type SYMBOL) (default Unk))
   (slot son (type SYMBOL) (default Unk))
   (slot Ola (type SYMBOL) (default Unk))
   (slot Gala (type SYMBOL) (default Unk))
   (slot Mela (type SYMBOL) (default Unk))
   (slot Oh (type SYMBOL) (default Unk))
   (slot Ah (type SYMBOL) (default Unk))
   (slot Eh (type SYMBOL) (default Unk))
)

(deffacts data
   (family (name Og) (Mela false) (Ola false) (Gala true) (Eh false) (Ah false) (Oh true))
   (family (name Gog) (Ola false) (Gala false) (Mela true) (Eh false) (Oh false) (Ah true))
   (family (name Magog) (Mela false) (Gala false) (Ola true) (Ah false) (Oh false) (Eh true))
)
```

Після виконання даного кроку, можна зробити правило кінцевого виводу на екран користувачеві програми.

```
(defrule PrintInfo
   (family (name ?name) (wife ?wife) (son ?son))
   ?Og<-(family (name Og) (wife ~Unk) (son ~Unk))
   ?Gog<-(family (name Gog) (wife ~Unk) (son ~Unk))
   ?Magog<-(family (name Magog) (wife ~Unk) (son ~Unk))
   =>
   (printout t crlf ?name"'s data:" crlf
   "Wife is:"?wife crlf
   "Son is:"?son crlf crlf)
)
```

Далі після виконання попередніх кроків, потрібно написати правила перевірки кожної умови згідно завдання, для початку я логічно розв’язав цю задачу на листочку, щоб правильно і доцільно перетворити це на робочий код.

Спочатку я відтворив правила перевірки жінок, для кожного чоловіка.

```
(defrule IfOgWifeGala
    (family (name ?name))
    ?Family<-(family (name Og) (wife Gala))
    ?Gog<-(family (name Gog))
    ?Magog<-(family (name Magog))
    =>
    (modify ?Gog (Gala false))
    (modify ?Magog (Gala false))
)

(defrule IfGogWifeMela
    (family (name ?name))
    ?Family<-(family (name Gog) (wife Mela))
    ?Og<-(family (name Og))
    ?Magog<-(family (name Magog))
    =>
    (modify ?Og (Mela false))
    (modify ?Magog (Mela false))
)

(defrule IfMagogWifeOla
    (family (name ?name))
    ?Family<-(family (name Magog) (wife Ola))
    ?Og<-(family (name Og))
    ?Gog<-(family (name Gog))
    =>
    (modify ?Og (Ola false))
    (modify ?Gog (Ola false))
)
```

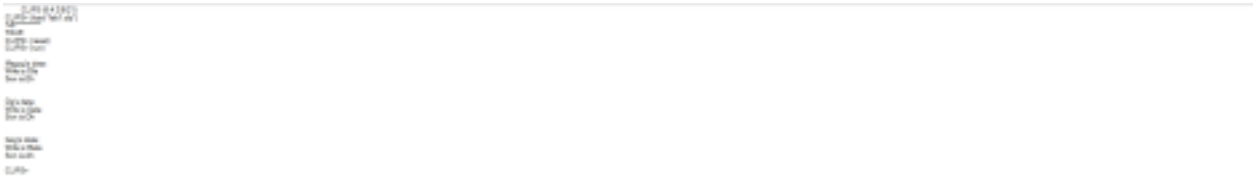
Після цього аналогічно відтворив правила для перевірки який син відповідає конкретному чоловікові.

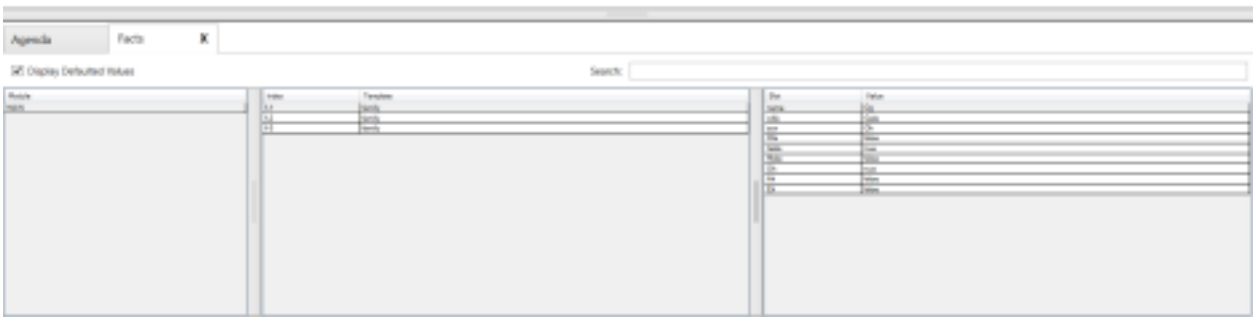
```
(defrule IfOgSonOh
   (family (name ?name))
   ?Family<-(family (name Og) (son Oh))
   ?Gog<-(family (name Gog))
   ?Magog<-(family (name Magog))
   =>
   (modify ?Gog (Oh false))
   (modify ?Magog (Oh false))
)

(defrule IfGogSonAh
   (family (name ?name))
   ?Family<-(family (name Gog) (son Ah))
   ?Og<-(family (name Og))
   ?Magog<-(family (name Magog))
   =>
   (modify ?Og (Ah false))
   (modify ?Magog (Ah false))
)

(defrule IfMagogSonEh
   (family (name ?name))
   ?Family<-(family (name Gog) (son Eh))
   ?Og<-(family (name Og))
   ?Gog<-(family (name Gog))
   =>
   (modify ?Og (Eh false))
   (modify ?Gog (Eh false))
)
```

Після цього я створив кінцеві правила, які будуть перевіряти всі умови та видавати кінцевий результат, який буде виводити правило print.

```
30  (defrule IfOgWifeGalaSonOh
31      (family (name ?name))
32      ?Family<-(family (name Og) (Mela false) (Ola false) (Eh false) (Ah false))
33      =>
34      (modify ?Family (wife Gala) (son Oh) (Gala true) (Oh true))
35  )
36
37  (defrule IfGogWifeMelaSonAh
38      (family (name ?name))
39      ?Family<-(family (name Gog) (Gala false) (Ola false) (Oh false) (Eh false))
40      =>
41      (modify ?Family (wife Mela) (son Ah) (Mela true) (Ah true))
42  )
43
44  (defrule IfMagogWifeOlaSonEh
45      (family (name ?name))
46      ?Family<-(family (name Magog) (Gala false) (Mela false) (Oh false) (Ah false))
47      =>
48      (modify ?Family (wife Ola) (son Eh) (Ola true) (Eh true))
49  )
```

Скріншот результатів:





Висновки:

Під час виконання цією лабораторної роботи я навчився використовувати мову програмування Cool з програмним середовищем Clips. Я розробив застосунок, що вирішує логічну задачу згідно варіанту заданого викладачем.