МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Кубанский государственный университет»

Кафедра вычислительной математики и информатики

ЛАБОРАТОРНАЯ РАБОТА

«Разработать программу " Служащие"»

по курсу

«Интеллектуальные системы и технологии в науке и образовании**»**

Выполнил:

Студент группы 102/1 Пасько Д. А.

подпись\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Проверил:

профессор каф. ВМиИ ФМиКН Вишняков Ю.М.

оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Краснодар 2020

1. **Вариант задания**

Разработать программу "Служащие" (номер 16 в списке <https://github.com/PasaOpasen/Old_Math_Projects/blob/master/prolog/Temy_individualnykh_zadaniy_IsiT_v_NO.pdf>). Программа демонстрирует метод отката после неудачи с предикатом fail. Программа выводит полный список служащих; выводит список мужчин; рассчитывает почасовую оплату.

1. **Теоретико-множественные положения**

Очень часто в программах необходимо выполнить одну и ту же задачу несколько раз. В программах на Prolog повторяющиеся операции обычно выполняются при помощи правил, которые используют *откат* или *рекурсию*. В лабораторной работе рассматриваются итерационные правила, а также общие способы их построения.

Правила повтора и рекурсии должны содержать средства управления их выполнением с тем, чтобы их использование было удобными. Встроенные предикаты Turbo Prolog fail и cut используются для управления откатами, а условия завершения используются для управления рекурсией.

Цели управляют программой на Prolog, обеспечивая выполнение последовательности определенных задач. Цели могут содержать подцели, которые, в свою очередь, могут содержать правила. Правила часто требуют, чтобы такие задачи, как поиск элементов в базе данных или вывод данных на экран выполнялись несколько раз.

Правила Prolog, выполняющие повторения, используют откат, а правила, выполняющие рекурсию, используют самовызов.

Вид правила, выполняющего повторение, следующий:

repetitive\_rule :- ***/\* правило повторения \*/***

<предикаты и правила>,

fail. ***/\* неудача \*/***

Конструкция <предикаты и правила> в теле правила обозначает предикаты, содержащие несколько утверждений, а также правила, определенные в программе. Встроенный предикат fail (неудача) вызывает откат, так что предикаты и правила выполняются еще раз.

1. **Результаты выполнения задания**

Задание выполнялось в среде Turbo Prolog (<http://old-dos.ru/index.php?page=files&mode=files&do=show&id=883>). Пришлось столкнуться с некоторыми трудностями; в частности, среда запускается только под Windows 32bit, а при использовании эмуляторов типа DOSBox (<https://www.dosbox.com>) нельзя вставлять скопированный текст или открывать файлы, расположенные на жёстких дисках. Вдобавок среда никак не реагирует на мышь и не раскрывается на весь экран (по крайней мере на моём нетбуке с Windows 8.1).

Написанный код:

***/\* Программа: Служащие \*/***

***/\* Назначение:***

***Демонстрация использования селектирующих правил на основе ОПН-метода***

***(ОНП = откат после неудачи)\*/***

**domains**

name, sex, department = symbol ***%имя, пол, место работы***

pay\_rate = real ***%ставка оплаты труда***

**predicates**

employee(name,sex,department,pay\_rate) ***% работник***

show\_employees

show\_employees\_male

**clauses**

***%используется очень мало записей, чтобы ответы поместились в окошко Turbo Prolog***

employee("Presley Perry","Male","ACCT",133.50).

employee("Noelle Carter","Female","DATA",145.00).

employee("Kye Coleman","Male","DATA",346.00).

employee("Sheila Burton","Female","ADVE",235.00).

***/\* Правило для генерации списка служащих любого пола \*/***

show\_employees :-

employee(Name, Sex, Dept, Pay\_rate),

write(Name," Sex: ",Sex," Department: ", Dept, " Pay by hour($): ", Pay\_rate),

nl,nl, ***% nl = переход на следующую строку***

fail.

***/\* Правило для генерации списка служащих мужского пола \*/***

show\_employees\_male :-

employee(Name, "Male", Dept, Pay\_rate),

write(Name," Department: ", Dept, " Pay($): ", Pay\_rate),

nl,nl,

fail.

**goal**

write("Employees with their pay rate:"),

nl, nl,

show\_employees,

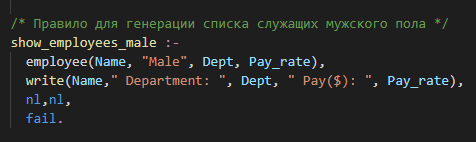
nl, nl,

write("Employees with their pay rate (only men):"),

nl, nl,

show\_employees\_male.

Метод отката демонстрируется, например, здесь:



Результаты выполнения цели (goal):

Изображение выглядит как снимок экрана, монитор

Автоматически созданное описание

Вывод всех служащих

Изображение выглядит как снимок экрана, монитор, экран, сидит

Автоматически созданное описание

Вывод служащих мужского пола

1. **Заключение**

В результате выполнения данной лабораторной работы была рассмотрена возможность использования повторяющихся операций через откат после неудачи (ОПТ). Повторяющиеся операции – это почти самые распространённые операции во многих языках программирования, поэтому во избежание дублирования кода их нужно использовать внутри функций или циклов или даже методов классов. Неудивительно, что Prolog в какой-то мере предоставляет подобный функционал, хоть и своеобразно.

1. **Список литературы и ресурсов**
2. <http://fevt.ru/load/povtorenija_i_rekursija/77-1-0-342>
3. Visual Prolog официальный сайт [Электронный ресурс] – режим доступа: <https://www.visual-prolog.com/>.
4. Повторение и рекурсия в визуальном прологе. Методические указания к лабораторному практикуму для “ИНТЕЛЛЕКТУАЛЬНЫЕ ИНФОРМАЦИОННЫЕ СИСТЕМЫ” (2004)
5. <http://www.verim.org/project/prolog/listing/biblioteka-2>
6. <https://habr.com/ru/post/49399/>
7. Цуканова Н. И., Дмитриева Т. А. Теория и практика логического программирования на языке Visual Prolog 7. Учебное пособие для вузов. – М.: Горячая линия – Телеком, 2013. – 232 с.
8. David Woods. Prolog Lists [статья] / D. Woods - Week 8 – HT – 4с