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.

2. 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.

3. 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.
```

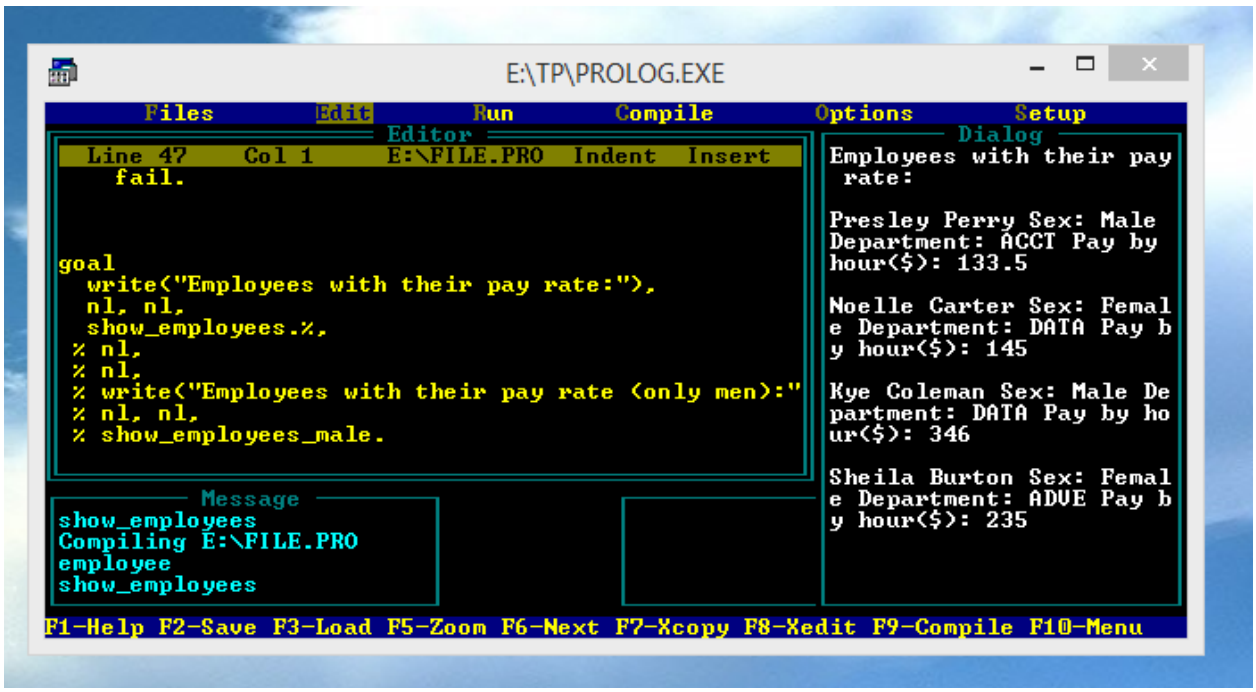
```
/* Правило для генерации списка служащих мужского пола */
show_employees_male :-
  employee(Name, "Male", Dept, Pay_rate),
  write(Name," Department: ", Dept, " Pay($): ", Pay_rate),
  nl,nl,
  fail.
```

```
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.
```

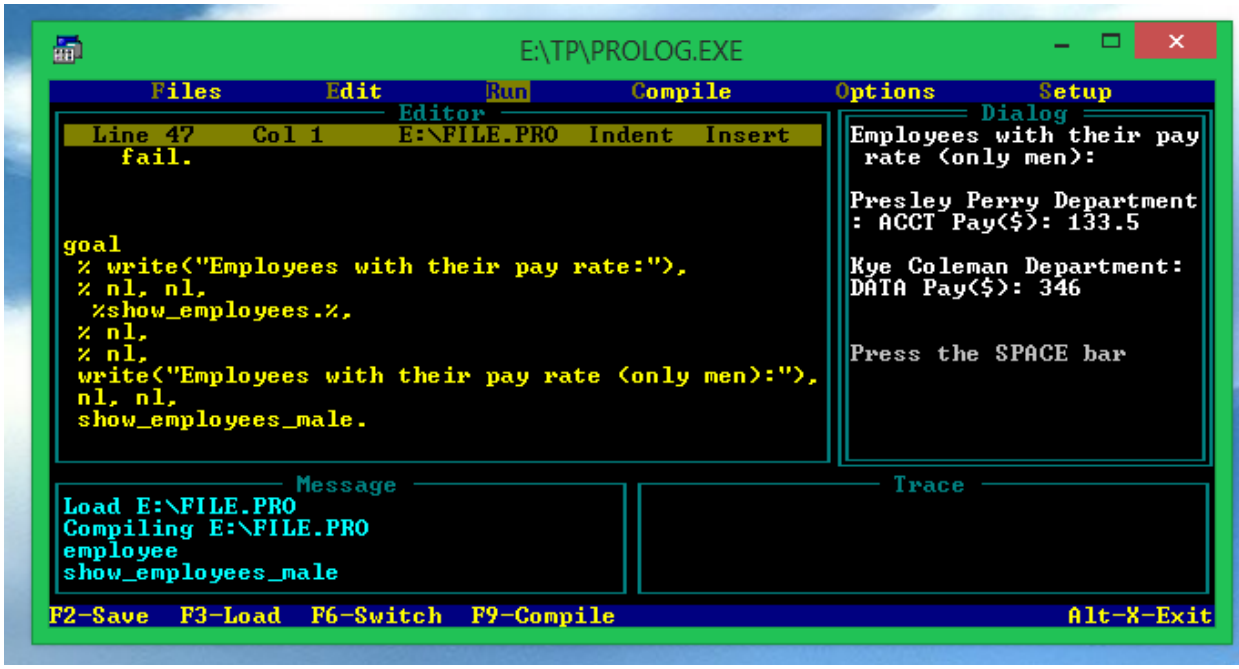
Метод отката демонстрируется, например, здесь:

```
/* 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.
```

Результаты выполнения цели (goal):



Вывод всех служащих



Вывод служащих мужского пола

4. 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.

5. Список литературы и ресурсов

1. http://fevt.ru/load/povtorenija_i_rekursija/77-1-0-342
2. Visual Prolog официальный сайт [Электронный ресурс] – *режим доступа:* https://www.visual-prolog.com/.
3. Повторение и рекурсия в визуальном прологе. Методические указания к лабораторному практикуму для “ИНТЕЛЛЕКТУАЛЬНЫЕ ИНФОРМАЦИОННЫЕ СИСТЕМЫ” (2004)
4. http://www.verim.org/project/prolog/listing/biblioteka-2
5. https://habr.com/ru/post/49399/
6. 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.
7. David Woods. Prolog Lists [статья] / D. Woods - Week 8 – HT – 4c