Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра автоматизированных систем управления (АСУ)

ЗАДАЧИ НА ГЕНЕРАЦИЮ НЕСКОЛЬКИХ ВАРИАНТОВ РЕШЕНИЙ

Лабораторная работа №5

по дисциплине

«Функциональное и логическое программирование»

Студент гр. 430-2

_____А.А. Лузинсан

«____»_____2023 г.

Руководитель

Ассистент кафедры АСУ

_____ А.В. Анфиловьев

«____»_____2023 г.

Томск 2023

Оглавление

Введение

Цель: получить навыки логического программирования для решения неоднозначных задач.

Задание: написать программу в соответствии с вариантом.

Задание по варианту №4:

Генератор позиций после 3-х шагов шахматного коня, без повторений (для исключения повторений можно использовать assert).

1 ХОД РАБОТЫ

В ходе выполнения работы был реализован механизм, который представляет собой модель движения трёх ходов коня на шахматной доске.

Первая строка устанавливает факт horse_position/2 как динамический, что означает, что его можно изменять в процессе выполнения программы. Этот факт предназначен для хранения всех позиций коня на доске. Правило check_board/2 проверяет, находится ли конь в пределах шахматной доски размером 8x8. Далее следуют восемь правил move_horse/2, которые определяют различные варианты ходов коня по шахматной доске. Каждый из них проверяет, находится ли новая позиция (NewX, NewY) в пределах доски, не встречалась ли ранее, и если эти условия выполнены — добавляет в базу фактов с помощью assert и выводит информацию о ходе. В свою очередь правило generate_3movements_horse/2 представляет собой последовательность из трех ходов коня, в которой используется форматированный вывод для отображения информации о текущем ходе. Строка horse_position(SecondX, SecondY) и аналогичная ей horse_position(ThirdX, ThirdY) заносят в соответствующие переменные данные из внутренней базы данных. И далее вызывается правило horse_movement/2, которое пробует для каждой такой пары все варианты перемещений.

Листинг кода лабораторной работы представлен в приложении А.1.

2 ТЕСТИРОВАНИЕ ПРОГРАММЫ

Таким образом, тестируя с начальной позицией (7,1), мы получаем результат, часть которого представлена на рисунке 2.1.

```
Новый мув
←←↓
Шаг влево вниз.
Новая позиция (6,2)
true ;
←←↑
Шаг влево вверх.
Новая позиция (6,4)
true ;

Новый мув
→→↑
Шаг вправо вверх.
Новая позиция (7,3)
true ;
←←↓
Шаг влево вниз.
Новая позиция (3,1)
true ;
←←↑
Шаг влево вверх.
Новая позиция (3,3)
true ;
```

Рисунок 2.1 — Часть результата выполнения программы

Заключение

В результате выполнения лабораторной работы я получила навыки логического программирования для решения неоднозначных задач.

Приложение А
(обязательное)
Листинг программы

Листинг А.1 — Программа, реализующая генератор позиций после 3-х шагов шахматного коня

```
:-dynamic horse_position/2.

% проверка на выход за пределы шахматной доски
check_board(X, Y) :- X > 0, X < 9,
   Y > 0, Y < 9.

% Шаг вверх влево ↑↑←
move_horse(X, Y) :- NewX is X - 1, NewY is Y + 2,
        check_board(NewX, NewY),
        not(horse_position(NewX, NewY)),
        format("↑↑←~nШаг вверх влево.~nНовая позиция (~w,~w)~n",
              [NewX, NewY]),
        assert(horse_position(NewX, NewY)).

% Шаг вверх вправо ↑↑→
move_horse(X, Y) :- NewX is X + 1, NewY is Y + 2,
        check_board(NewX, NewY),
        not(horse_position(NewX, NewY)),
        format("↑↑→~nШаг вверх вправо.~nНовая позиция (~w,~w)~n",
              [NewX, NewY]),
        assert(horse_position(NewX, NewY)).

% Шаг вправо вверх →→↑
move_horse(X, Y) :- NewX is X + 2, NewY is Y + 1,
        check_board(NewX, NewY),
        not(horse_position(NewX, NewY)),
        format("→→↑~nШаг вправо вверх.~nНовая позиция (~w,~w)~n",
              [NewX, NewY]),
        assert(horse_position(NewX, NewY)).

% Шаг вправо вниз →→↓
move_horse(X, Y) :- NewX is X + 2, NewY is Y - 1,
        check_board(NewX, NewY),
        not(horse_position(NewX, NewY)),
        format("→→↓~nШаг вправо вниз.~nНовая позиция (~w,~w)~n",
              [NewX, NewY]),
        assert(horse_position(NewX, NewY)).
```

```
% Шаг вниз вправо ↓↓→
move_horse(X, Y) :- NewX is X + 1, NewY is Y - 2,
        check_board(NewX, NewY),
        not(horse_position(NewX, NewY)),
        format("↓↓→~nШаг вниз вправо.~nНовая позиция (~w,~w)~n",
                [NewX, NewY]),
        assert(horse_position(NewX, NewY)).

% Шаг вниз влево ↓↓←
move_horse(X, Y) :- NewX is X - 1, NewY is Y - 2,
        check_board(NewX, NewY),
        not(horse_position(NewX, NewY)),
        format("↓↓←~nШаг вниз влево.~nНовая позиция (~w,~w)~n",
                [NewX, NewY]),
        assert(horse_position(NewX, NewY)).

% Шаг влево вниз ←←↓
move_horse(X, Y) :- NewX is X - 2, NewY is Y - 1,
        check_board(NewX, NewY),
        not(horse_position(NewX, NewY)),
        format("←←↓~nШаг влево вниз.~nНовая позиция (~w,~w)~n",
                [NewX, NewY]),
        assert(horse_position(NewX, NewY)).

% Шаг влево вверх ←←↑
move_horse(X, Y) :- NewX is X - 2, NewY is Y + 1,
        check_board(NewX, NewY),
        not(horse_position(NewX, NewY)),
        format("←←↑~nШаг влево вверх.~nНовая позиция (~w,~w)~n",
                [NewX, NewY]),
        assert(horse_position(NewX, NewY)).


horse_movement(X, Y) :-
        format("~nНовый мув~n"),
        move_horse(X, Y).



% ↑
% ←
% →
% ↓
```

```
generate_3movements_horse(FirstX, FirstY) :-
        format("~nНачальная позиция (~w,~w)~n", [FirstX, FirstY]),
        check_board(FirstX, FirstY),
        assert(horse_position(FirstX, FirstY)),

        % Расчёт первого хода
        format("~nПервый ход:~n"),
        move_horse(FirstX, FirstY);

        % Расчёт второго хода
        format("~nВторой ход:~n"),
        horse_position(SecondX, SecondY),
        horse_movement(SecondX, SecondY);

        % Расчёт третьего хода
        format("~nТретий ход:~n"),
        horse_position(ThirdX, ThirdY),
        horse_movement(ThirdX, ThirdY).
```