Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет информационных технологий и управления

ОТЧЕТ

по лабораторной работе № 1

на тему: «Проектирование компонента решателя задач на основе декларативных моделей решения задач»

Выполнил:

Магистрант группы: 426401

Гуменный Н. А.

Проверил:  
Ковалёв М. В.

Минск 2025

**Выбор модели**

В рамках лабораторной работы была выбрана задача криптарифметики, в которой требуется найти цифровое представление букв так, чтобы соблюдалось заданное арифметическое равенство. Данная задача относится к классу *CSP* (*Constraint Satisfaction Problem*) и идеально решается средствами декларативного программирования.

В качестве инструмента был выбран язык *Prolog* и библиотека ограничений *CLPFD* (*Constraint Logic Programming over Finite Domains*).

Такой подход позволяет описывать ограничения напрямую, без необходимости явно указывать порядок поиска решений. Вместо пошагового перебора программа просто описывает требования к допустимому решению, а Prolog с *CLPFD* находит решение самостоятельно.

**Постановка задачи**

Требуется найти такое соответствие между буквами и цифрами, чтобы выполнялось выражение:

КОТ + ПЁС = ДРУГ

При этом:

- каждая буква обозначает уникальную цифру от 0 до 9;

- первые буквы слов (К, П, Д) не равны нулю;

- полученные числа из слов должны удовлетворять указанному равенству.

Например:

1) КОТ = 374

2) ПЁС = 629

3) ДРУГ = 1003

**Алгоритм решения**

Для реализации задачи была построена декларативная модель с применением следующих элементов:

1. Определение переменных и их доменов

Используется список переменных: [К, О, Т, П, Ё, С, Д, Р, У, Г], каждая из которых принимает значение от 0 до 9:

Буквы ins 0..9

2. Ограничение уникальности значений

Все буквы должны соответствовать различным цифрам:

all\_different(Буквы)

3. Ограничение на лидирующие цифры

Буквы К, П и Д не могут быть равны 0, поскольку они являются первыми в числе:

К #\= 0, П #\= 0, Д #\= 0

4. Преобразование букв в числа

Создан предикат *буквы\_в\_число/3*, который рекурсивно строит число на основе списка букв и их значений. Например:

буквы\_в\_число([К, О, Т], [К, О, Т], КОТ)

5. Задание основного уравнения

Основное равенство устанавливается декларативно:

КОТ + ПЁС #= ДРУГ

6. Поиск решения

Запускается *labeling([ff], Буквы)* , где *ff* означает стратегию *first-fail*: приоритетно выбираются переменные с наименьшим числом возможных значений.

7. Маскирование решения

После нахождения полного решения к нему применяется маска вида [1,0,1, 1,0,1, 1,0,0,0]. Значения с маской 1 сохраняются, остальные заменяются на '\_'.

8. Повторное восстановление решения

С использованием маски и известных цифр задача решается заново, при этом скрытые буквы подбираются так, чтобы уравнение снова выполнялось.

**Структура проекта**

Код реализован в одном файле kriptarif.pl и содержит следующие основные компоненты:

– буквы\_в\_число/3 преобразует список букв в числовое значение;

– решить\_пример/1 находит полное решение задачи;

– маска/1 и применить\_маску/4 реализуют механизм

частичного скрытия значений;

– решить\_с\_подсказками/2 восстанавливает полное решение из частичного;

– запустить\_решатель/0 основной предикат, запускаемый автоматически.

**Полный код файла kriptarif.pl с пояснением в «комментариях»**

:- encoding(utf8).

:- use\_module(library(clpfd)).

% Преобразование слова в число по правилам: [К, О, Т] => 100\*К + 10\*О + Т

буквы\_в\_число([], [], 0).

буквы\_в\_число([Б|Буквы], [Цифра|Цифры], Число) :-

буквы\_в\_число(Буквы, Цифры, Ост),

length([Б|Буквы], Длина),

Степень is 10^(Длина - 1),

Число #= Цифра \* Степень + Ост.

% Основное криптарифметическое уравнение: КОТ + ПЁС = ДРУГ

решить\_пример(Буквы) :-

format("Начинаем поиск решения...\n"),

% Инициализация переменных

Буквы = [К, О, Т, П, Ё, С, Д, Р, У, Г],

Буквы ins 0..9,

format("Установлены домены переменных: ~w\n", [Буквы]),

% Ограничение уникальности

all\_different(Буквы),

format("Установлено ограничение уникальности\n"),

% Ограничение первых цифр

К #\= 0, П #\= 0, Д #\= 0,

format("Установлены ограничения на первые цифры: К=~w, П=~w, Д=~w\n", [К, П, Д]),

% Формирование чисел с помощью буквы\_в\_число

буквы\_в\_число([К, О, Т], [К, О, Т], КОТ),

буквы\_в\_число([П, Ё, С], [П, Ё, С], ПЁС),

буквы\_в\_число([Д, Р, У, Г], [Д, Р, У, Г], ДРУГ),

format("Сформированы числа: КОТ=~w, ПЁС=~w, ДРУГ=~w\n", [КОТ, ПЁС, ДРУГ]),

% Базовые ограничения на числа

КОТ #>= 100, КОТ #< 1000,

ПЁС #>= 100, ПЁС #< 1000,

ДРУГ #>= 1000, ДРУГ #< 10000,

format("Установлены ограничения на числа\n"),

% Основное уравнение

КОТ + ПЁС #= ДРУГ,

format("Установлено основное уравнение\n"),

% Поиск решения с использованием first-fail стратегии

format("Начинаем перебор значений...\n"),

labeling([ff], Буквы),

% Вывод решения

format("\nНайдено решение:\n"),

format("КОТ = ~d, ПЁС = ~d, ДРУГ = ~d\n", [КОТ, ПЁС, ДРУГ]),

format("Буквы:\n К=~d, О=~d, Т=~d, П=~d, Ё=~d, С=~d, Д=~d, Р=~d, У=~d, Г=~d\n",

[К, О, Т, П, Ё, С, Д, Р, У, Г]).

% Маска, где 1 — показать букву, 0 — скрыть

маска([1,0,1, 1,0,1, 1,0,0,0]).

% Применение маски к списку значений с отслеживанием видимых и скрытых значений

применить\_маску([], [], [], []).

применить\_маску([Б|Бс], [М|Мс], [Видно|ВиднСл], [Б|Ост]) :- М =:= 1, !,

Видно = Б,

применить\_маску(Бс, Мс, ВиднСл, Ост).

применить\_маску([\_|Бс], [0|Мс], ['\_'|ВиднСл], Ост) :-

применить\_маску(Бс, Мс, ВиднСл, Ост).

% Печать маскированных значений

написать\_буквы([], []).

написать\_буквы([Б|Бс], [Ц|Цс]) :-

format("~w = ~w\n", [Б, Ц]),

написать\_буквы(Бс, Цс).

% Повторное решение с подсказками

решить\_с\_подсказками(Маска, Значения) :-

Буквы = [К, О, Т, П, Ё, С, Д, Р, У, Г],

Буквы ins 0..9,

all\_different(Буквы),

К #\= 0, П #\= 0, Д #\= 0,

буквы\_в\_число([К, О, Т], [К, О, Т], КОТ),

буквы\_в\_число([П, Ё, С], [П, Ё, С], ПЁС),

буквы\_в\_число([Д, Р, У, Г], [Д, Р, У, Г], ДРУГ),

КОТ + ПЁС #= ДРУГ,

применить\_подсказки(Буквы, Маска, Значения),

labeling([ff], Буквы),

format("\nВосстановленное решение:\n"),

format("КОТ = ~d, ПЁС = ~d, ДРУГ = ~d\n", [КОТ, ПЁС, ДРУГ]),

format("Буквы: К=~d О=~d Т=~d П=~d Ё=~d С=~d Д=~d Р=~d У=~d Г=~d\n",

[К, О, Т, П, Ё, С, Д, Р, У, Г]).

% Применение известных значений к переменным

применить\_подсказки([], [], []).

применить\_подсказки([Б|Бс], [1|Мс], [Б|Зс]) :-

применить\_подсказки(Бс, Мс, Зс).

применить\_подсказки([\_|Бс], [0|Мс], [\_|Зс]) :-

применить\_подсказки(Бс, Мс, Зс).

% Главный предикат

запустить\_решатель :-

format("Запуск решателя...\n"),

% Получаем решение

catch(

решить\_пример(Буквы),

Error,

(format("Ошибка при поиске решения: ~w\n", [Error]), fail)

),

format("\nПрименяем маску...\n"),

% Применяем маску к решению

маска(Маска),

применить\_маску(Буквы, Маска, Скрытые, \_),

format("\nЧастично известные буквы:\n"),

написать\_буквы(['К','О','Т','П','Ё','С','Д','Р','У','Г'], Скрытые),

% Решаем заново с теми же ограничениями, но частью переменных известны

решить\_с\_подсказками(Маска, Скрытые).

% Запуск автоматически

:- initialization(запустить\_решатель, main).

**Запуск программы**

1. Откройте терминал в папке с кодом (*PowerShell/Command Prompt).*

2. Запустите *Prolog* и сразу загрузите файл:

Swipl kriptarif.pl

После загрузки Prolog вы увидите вывод как на рисунке 1.

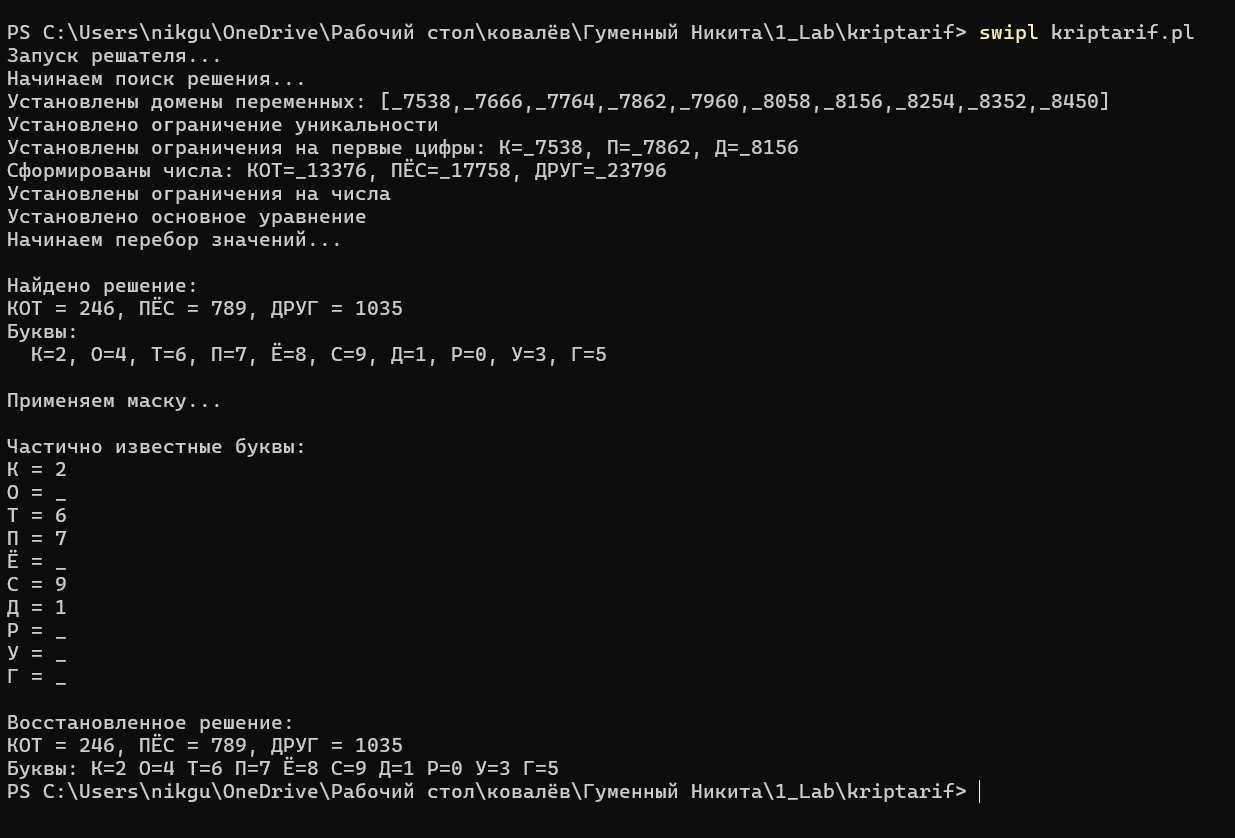


Рисунок 1 – Результат программы

**Вывод по лабораторной работе**

Разработанная декларативная модель криптарифметической задачи на *Prolog* демонстрирует высокую выразительность и эффективность. Благодаря использованию библиотеки *clpfd* удалось описать условия задачи в терминах ограничений и возложить поиск решения на встроенный механизм логического вывода.

Особенности реализации:

– простое и наглядное задание условий через *all\_different;*

– автоматическая генерация решения и его маскирование;

– повторное восстановление скрытых значений по частичной информации;

– пошаговый вывод для пользователя, облегчающий понимание происходящего.

Таким образом, лабораторная работа подтверждает применимость декларативного подхода и языка *Prolog* для решения задач удовлетворения ограничений, в частности криптарифметических выражений.