МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»
(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа №1

по дисциплине: Исследование операций
тема: “Исследование множества опорных планов системы ограничений задачи линейного программирования (задачи ЛП) в канонической форме”

Выполнил: ст. группы ПВ-231
Столяров Захар

Проверил:
Вирченко Юрий Петрович

Белгород, 2025 г.

Лабораторная работа №1 «Исследование множества опорных планов системы ограничений задачи линейного программирования (задачи ЛП) в канонической форме.»

Цель работы: изучить метод Гаусса-Жордана и операцию замещения, а также освоить их применение к отысканию множества допустимых базисных видов системы линейных уравнений, и решению задачи линейного программирования простым перебором опорных решений.

Вариант 14

Оглавление

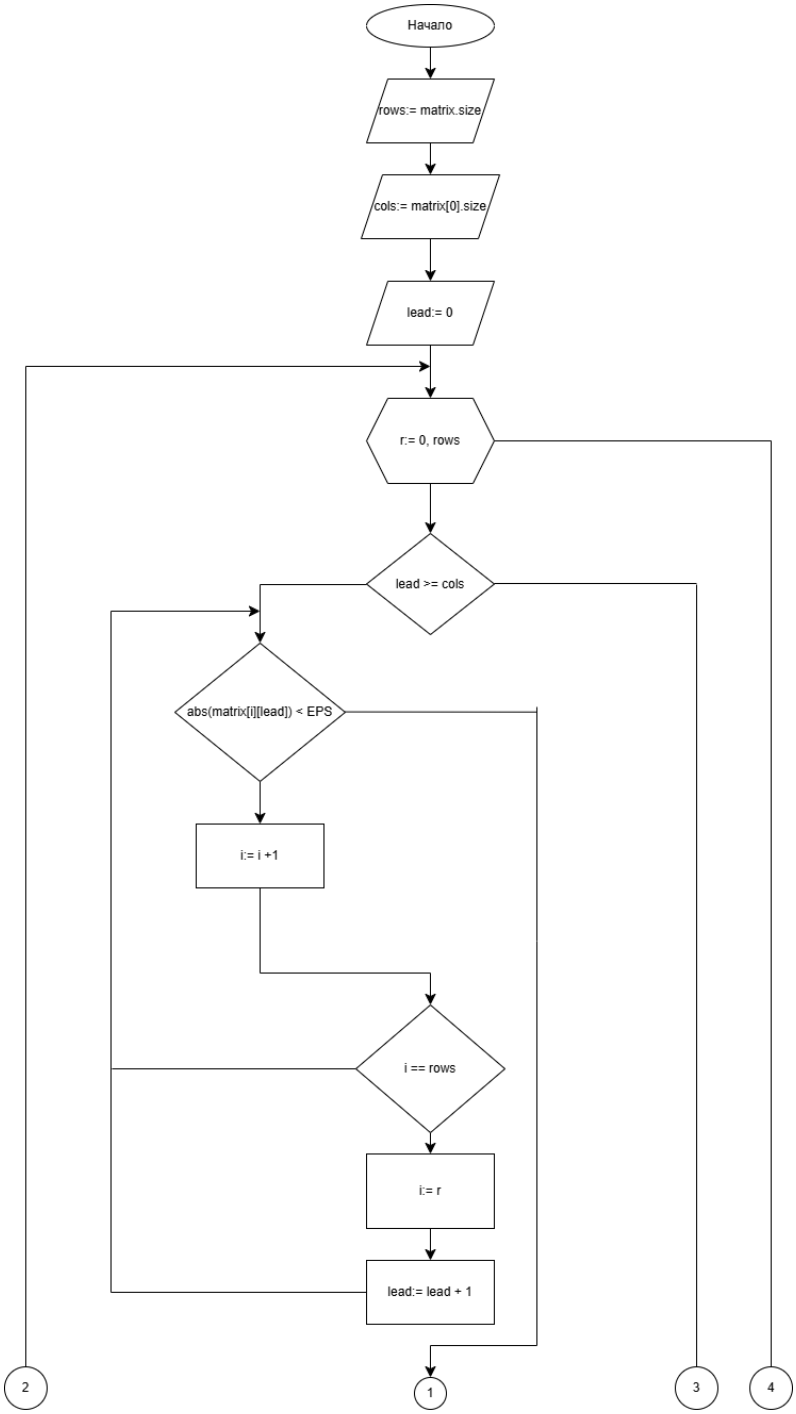
Система уравнений моего варианта

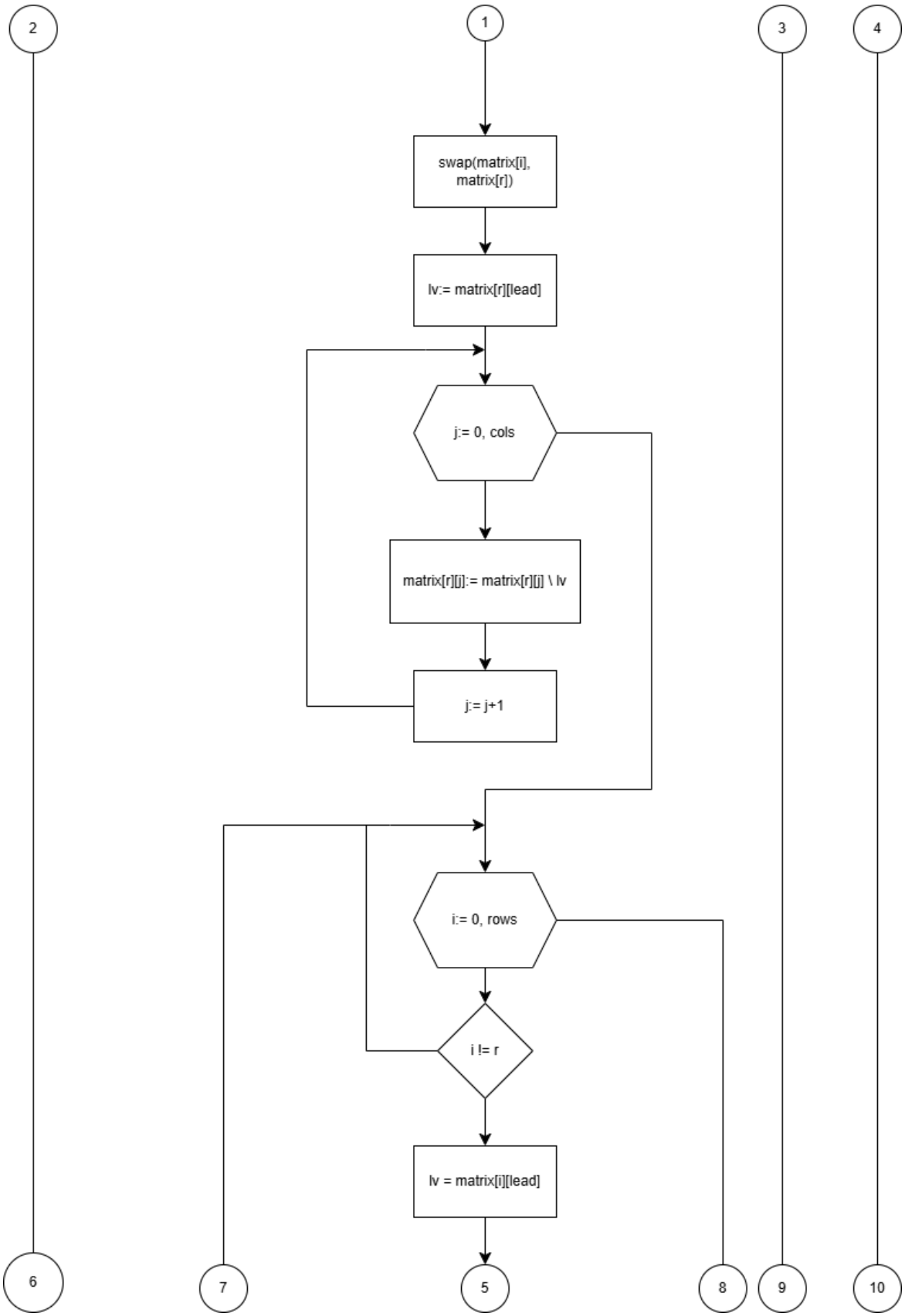
$$\begin{cases} -x_1 + 5x_2 - 4x_3 - 6x_4 + x_6 = -9 \\ 8x_1 + x_2 - x_3 + 2x_5 + 3x_6 = 8 \\ 4x_1 + 3x_2 - 2x_3 + 9x_4 + x_5 + 7x_6 = 1 \end{cases}$$

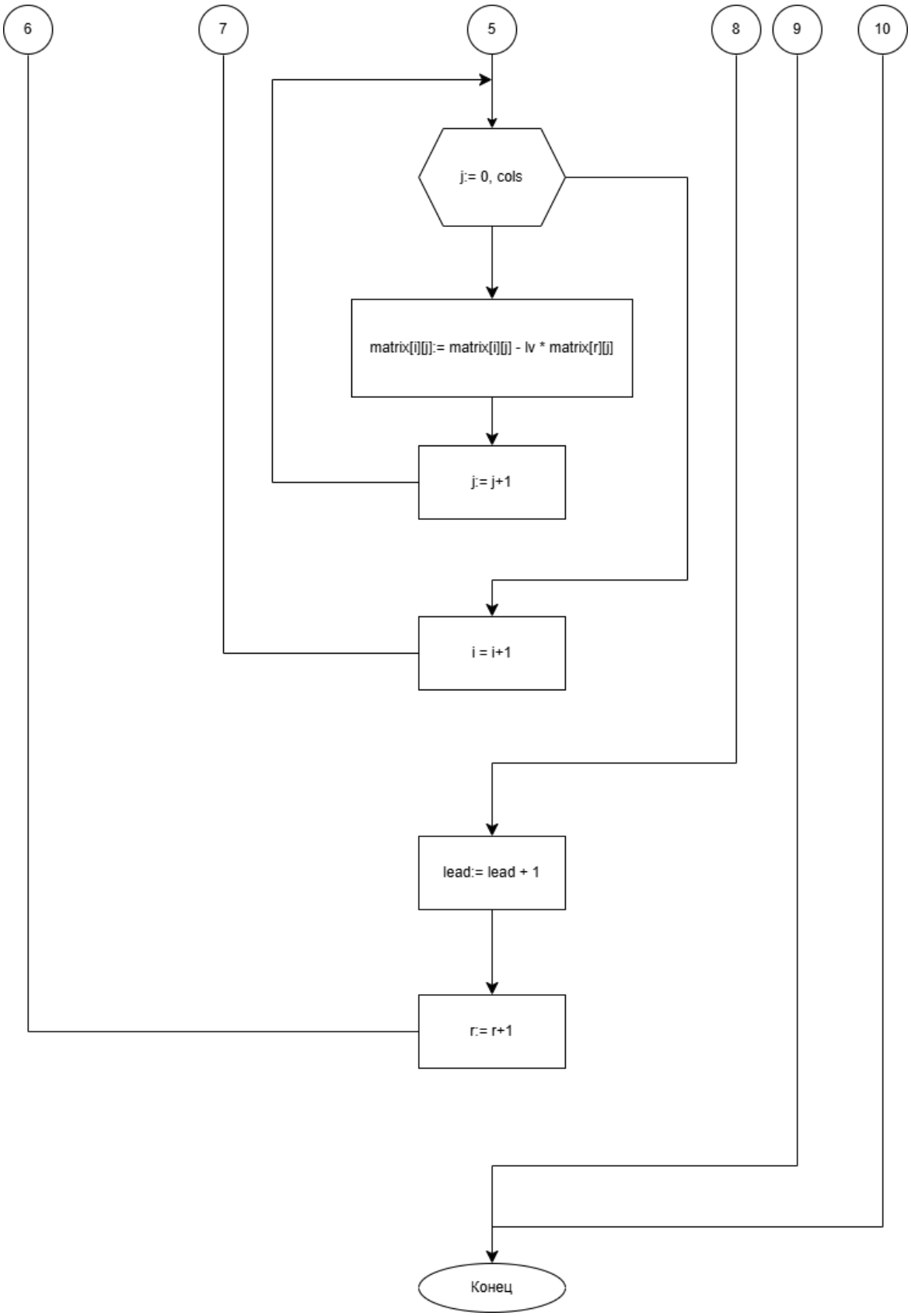
Задание 1

Составить программу для отыскания всех базисных видов системы линейных уравнений:

Блок-схема функции gauss_jordan




2
1
3
4
swap(matrix[i], matrix[r])
lv:= matrix[r][lead]
j:= 0, cols
matrix[r][j]:= matrix[r][j] \ lv
j:= j+1
i:= 0, rows
i != r
lv = matrix[i][lead]
6
7
5
8
9
10


6
7
5
8
9
10
j:= 0, cols
matrix[i][j]:= matrix[i][j] - lv * matrix[r][j]
j:= j+1
i = i+1
lead:= lead + 1
r:= r+1
Конец

Код программы

```cpp
#include <iostream>
#include <vector>
#include <cmath>
#include <algorithm>
#include <numeric>
#include <iomanip>

using namespace std;

const double EPS = 1e-9;

// Функция для перестановки столбцов в матрице
void rearrangeColumns(vector<vector<double>>& matrix, const vector<int>& basis) {
    int cols = matrix[0].size() - 1; // Исключаем столбец свободных членов
    for (auto& row : matrix) {
        vector<double> new_row;
        // Добавляем базисные столбцы
        for (int b : basis) new_row.push_back(row[b]);
        // Добавляем оставшиеся небазисные столбцы
        for (int c = 0; c < cols; ++c) {
            if (find(basis.begin(), basis.end(), c) == basis.end()) {
                new_row.push_back(row[c]);
            }
        }
        // Добавляем свободный член
        new_row.push_back(row.back());
        row = new_row;
    }
}

// Улучшенный метод Гаусса-Жордана с ведением журнала ведущих столбцов
void gauss_jordan(vector<vector<double>>& matrix, vector<int>& leading_columns) {
    int rows = matrix.size();
    int cols = matrix[0].size() - 1;
    leading_columns.clear();
    int lead = 0;

    for (int r = 0; r < rows && lead < cols; ++r) {
        // Поиск ненулевого элемента в столбце lead начиная с текущей строки
        int i = r;
        while (i < rows && abs(matrix[i][lead]) < EPS) ++i;
        if (i == rows) {
            ++lead;
            --r;
            continue;
```

```
        }

        swap(matrix[i], matrix[r]);
        double div = matrix[r][lead];
        for (int j = lead; j <= cols; ++j) matrix[r][j] /= div;

        for (i = 0; i < rows; ++i) {
            if (i != r) {
                double factor = matrix[i][lead];
                for (int j = lead; j <= cols; ++j)
                matrix[i][j] -= factor * matrix[r][j];
            }
        }
        leading_columns.push_back(lead);
        ++lead;
    }
}

// Функция для проверки, является ли комбинация допустимым базисом
bool is_valid_basis(const vector<int>& basis, const vector<int>& leading_columns) {
    if (leading_columns.size() != 3) return false;
    for (int lc : leading_columns)
    if (lc >= basis.size()) return false;
    return true;
}

// Функция для вывода решения с учетом переставленных столбцов
void print_solution(const vector<vector<double>>& matrix, const vector<int>& basis) {
    cout << "Базисные переменные: ";
    for (int b : basis) cout << "x" << b + 1 << " ";
    cout << "\nРешение:\n";

    vector<bool> is_basis_var(matrix[0].size() - 1, false);
    for (int b : basis) is_basis_var[b] = true;

    for (size_t r = 0; r < matrix.size(); ++r) {
        int lead_col = -1;
        for (size_t c = 0; c < basis.size(); ++c) {
            if (abs(matrix[r][c] - 1.0) < EPS) {
                lead_col = c;
                break;
            }
        }

        if (lead_col == -1) {
            if (abs(matrix[r].back()) > EPS)
            cout << "Система несовместна\n";
```

```
            continue;
        }

        int actual_var = basis[lead_col];
        cout << "x" << actual_var + 1 << " = " << fixed << setprecision(2) << matrix[r].back();

        for (size_t c = basis.size(); c < matrix[r].size() - 1; ++c) {
            if (abs(matrix[r][c]) > EPS) {
                cout << " - " << fixed << setprecision(2) << matrix[r][c]
                << "*x" << (c >= basis.size() ?
                (matrix[0].size() - 1 > c ? c - basis.size() + basis.size() : 0) : c) + 1;
            }
        }
        cout << endl;
    }
    cout << "-------------------------\n";
}

void find_basic_solutions(vector<vector<double>> original, int num_vars) {
    int num_eqs = original.size();
    vector<int> vars(num_vars);
    iota(vars.begin(), vars.end(), 0);

    vector<bool> selector(num_vars);
    fill(selector.begin(), selector.begin() + num_eqs, true);

    int count = 0;
    do {
        vector<int> basis;
        for (int i = 0; i < num_vars; ++i)
        if (selector[i]) basis.push_back(i);
        if (basis.size() != num_eqs) continue;

        vector<vector<double>> matrix = original;
        rearrangeColumns(matrix, basis);

        vector<int> leading_columns;
        gauss_jordan(matrix, leading_columns);

        cout << "Комбинация базиса " << ++count << ":\n";
        cout << "Преобразованная матрица:\n";
        for (const auto& row : matrix) {
            for (double val : row) cout << fixed << setprecision(2) << setw(8) << val;
            cout << endl;
        }

        if (is_valid_basis(basis, leading_columns)) {
```

```
            print_solution(matrix, basis);
        }
        else {
            cout << "Не является допустимым базисом\n";
            cout << "-------------------------\n";
        }

    } while (prev_permutation(selector.begin(), selector.end()));
}

int main() {
    setlocale(LC_ALL, "Russian");
    vector<vector<double>> matrix = {
        {-1, 5, -4, -6, 0, 1, -9},
        {8, 1, -1, 0, 2, 3, 8},
        {4, 3, -2, 9, 1, 7, 1}
    };
    find_basic_solutions(matrix, 6);
    return 0;
}
```

Результат работы программы:

```
Комбинация базиса 1:
Преобразованная матрица:
    1.00    0.00    0.00    0.71    0.24    0.57    1.24
    0.00    1.00    0.00   17.57   -0.14    7.86   -0.14
    0.00    0.00    1.00   23.29   -0.24    9.43    1.76
Базисные переменные: x1 x2 x3
Решение:
x1 = 1.24 - 0.71*x4 - 0.24*x5 - 0.57*x6
x2 = -0.14 - 17.57*x4 + 0.14*x5 - 7.86*x6
x3 = 1.76 - 23.29*x4 + 0.24*x5 - 9.43*x6
-------------------------
Комбинация базиса 2:
Преобразованная матрица:
    1.00    0.00    0.00   -0.03    0.25    0.28    1.18
    0.00    1.00    0.00   -0.75    0.04    0.74   -1.47
    0.00    0.00    1.00    0.04   -0.01    0.40    0.08
Базисные переменные: x1 x2 x4
Решение:
x1 = 1.18 + 0.03*x3 - 0.25*x5 - 0.28*x6
x2 = -1.47 + 0.75*x3 - 0.04*x5 - 0.74*x6
x4 = 0.08 - 0.04*x3 + 0.01*x5 - 0.40*x6
-------------------------
Комбинация базиса 3:
Преобразованная матрица:
    1.00    0.00    0.00    1.00   24.00   10.00    3.00
    0.00    1.00    0.00   -0.60    3.60    2.20   -1.20
    0.00    0.00    1.00   -4.20  -97.80  -39.60   -7.40
Базисные переменные: x1 x2 x5
Решение:
x1 = 3.00 - 1.00*x3 - 24.00*x4 - 10.00*x6
x2 = -1.20 + 0.60*x3 - 3.60*x4 - 2.20*x6
x5 = -7.40 + 4.20*x3 + 97.80*x4 + 39.60*x6
-------------------------
Комбинация базиса 4:
Преобразованная матрица:
    1.00    0.00    0.00   -0.06   -0.70    0.25    1.13
    0.00    1.00    0.00   -0.83   -1.83    0.06   -1.61
    0.00    0.00    1.00    0.11    2.47   -0.03    0.19
Базисные переменные: x1 x2 x6
Решение:
x1 = 1.13 + 0.06*x3 + 0.70*x4 - 0.25*x5
x2 = -1.61 + 0.83*x3 + 1.83*x4 - 0.06*x5
x6 = 0.19 - 0.11*x3 - 2.47*x4 + 0.03*x5
-------------------------
Комбинация базиса 5:
Преобразованная матрица:
    1.00    0.00    0.00   -0.04    0.24    0.25    1.24
    0.00    1.00    0.00   -1.33   -0.05   -0.98    1.95
    0.00    0.00    1.00    0.06   -0.01    0.45   -0.01
Базисные переменные: x1 x3 x4
Решение:
x1 = 1.24 + 0.04*x2 - 0.24*x5 - 0.25*x6
x3 = 1.95 + 1.33*x2 + 0.05*x5 + 0.98*x6
x4 = -0.01 - 0.06*x2 + 0.01*x5 - 0.45*x6
-------------------------
Комбинация базиса 6:
Преобразованная матрица:
    1.00    0.00    0.00    1.67   30.00   13.67    1.00
    0.00    1.00    0.00   -1.67   -6.00   -3.67    2.00
    0.00    0.00    1.00   -7.00 -123.00  -55.00    1.00
Базисные переменные: x1 x3 x5
Решение:
x1 = 1.00 - 1.67*x2 - 30.00*x4 - 13.67*x6
x3 = 2.00 + 1.67*x2 + 6.00*x4 + 3.67*x6
x5 = 1.00 + 7.00*x2 + 123.00*x4 + 55.00*x6
-------------------------
Комбинация базиса 7:
Преобразованная матрица:
    1.00    0.00    0.00   -0.07   -0.56    0.25    1.25
    0.00    1.00    0.00   -1.20    2.20   -0.07    1.93
    0.00    0.00    1.00    0.13    2.24   -0.02   -0.02
Базисные переменные: x1 x3 x6
Решение:
x1 = 1.25 + 0.07*x2 + 0.56*x4 - 0.25*x5
x3 = 1.93 + 1.20*x2 - 2.20*x4 + 0.07*x5
x6 = -0.02 - 0.13*x2 - 2.24*x4 + 0.02*x5
-------------------------
```

```
Комбинация базиса 8:
Преобразованная матрица:
    1.00    0.00    0.00   -6.67    5.00   -4.67   11.00
    0.00    1.00    0.00    0.28   -0.17    0.61   -0.33
    0.00    0.00    1.00   27.17  -20.50   20.17  -40.00
Базисные переменные: x1 x4 x5
Решение:
x1 = 11.00 + 6.67*x2 - 5.00*x3 + 4.67*x6
x4 = -0.33 - 0.28*x2 + 0.17*x3 - 0.61*x6
x5 = -40.00 - 27.17*x2 + 20.50*x3 - 20.17*x6
-------------------------
Комбинация базиса 9:
Преобразованная матрица:
    1.00    0.00    0.00   -0.38    0.26    0.23    1.74
    0.00    1.00    0.00   -0.55    0.45   -0.03    0.88
    0.00    0.00    1.00    1.35   -1.02    0.05   -1.98
Базисные переменные: x1 x4 x6
Решение:
x1 = 1.74 + 0.38*x2 - 0.26*x3 - 0.23*x5
x4 = 0.88 + 0.55*x2 - 0.45*x3 + 0.03*x5
x6 = -1.98 - 1.35*x2 + 1.02*x3 - 0.05*x5
-------------------------
Комбинация базиса 10:
Преобразованная матрица:
    1.00    0.00    0.00   -4.55    3.73    7.64    8.45
    0.00    1.00    0.00   18.00  -15.00  -33.00  -29.00
    0.00    0.00    1.00    0.45   -0.27    1.64   -0.55
Базисные переменные: x1 x5 x6
Решение:
x1 = 8.45 + 4.55*x2 - 3.73*x3 - 7.64*x4
x5 = -29.00 - 18.00*x2 + 15.00*x3 + 33.00*x4
x6 = -0.55 - 0.45*x2 + 0.27*x3 - 1.64*x4
-------------------------
Комбинация базиса 11:
Преобразованная матрица:
    1.00    0.00    0.00  -24.60   -6.00   -6.20  -30.60
    0.00    1.00    0.00  -32.60   -8.00   -9.20  -38.60
    0.00    0.00    1.00    1.40    0.33    0.80    1.73
Базисные переменные: x2 x3 x4
Решение:
x2 = -30.60 + 24.60*x1 + 6.00*x5 + 6.20*x6
x3 = -38.60 + 32.60*x1 + 8.00*x5 + 9.20*x6
x4 = 1.73 - 1.40*x1 - 0.33*x5 - 0.80*x6
-------------------------
Комбинация базиса 12:
Преобразованная матрица:
    1.00    0.00    0.00    0.60   18.00    8.20    0.60
    0.00    1.00    0.00    1.00   24.00   10.00    3.00
    0.00    0.00    1.00    4.20    3.00    2.40    5.20
Базисные переменные: x2 x3 x5
Решение:
x2 = 0.60 - 0.60*x1 - 18.00*x4 - 8.20*x6
x3 = 3.00 - 1.00*x1 - 24.00*x4 - 10.00*x6
x5 = 5.20 - 4.20*x1 - 3.00*x4 - 2.40*x6
-------------------------
Комбинация базиса 13:
Преобразованная матрица:
    1.00    0.00    0.00  -13.75    7.75   -3.42  -17.17
    0.00    1.00    0.00  -16.50   11.50   -4.17  -18.67
    0.00    0.00    1.00    1.75    1.25    0.42    2.17
Базисные переменные: x2 x3 x6
Решение:
x2 = -17.17 + 13.75*x1 - 7.75*x4 + 3.42*x5
x3 = -18.67 + 16.50*x1 - 11.50*x4 + 4.17*x5
x6 = 2.17 - 1.75*x1 - 1.25*x4 - 0.42*x5
-------------------------
Комбинация базиса 14:
Преобразованная матрица:
    1.00    0.00    0.00   -0.15   -0.75    0.70   -1.65
    0.00    1.00    0.00    0.04    0.04    0.42    0.12
    0.00    0.00    1.00    4.08   -0.12    1.15    4.83
Базисные переменные: x2 x4 x5
Решение:
x2 = -1.65 + 0.15*x1 + 0.75*x3 - 0.70*x6
x4 = 0.12 - 0.04*x1 - 0.04*x3 - 0.42*x6
x5 = 4.83 - 4.08*x1 + 0.12*x3 - 1.15*x6
```

```
Комбинация базиса 15:
Преобразованная матрица:
    1.00    0.00    0.00   -2.63   -0.67   -0.61   -4.59
    0.00    1.00    0.00   -1.43    0.09   -0.36   -1.62
    0.00    0.00    1.00    3.54   -0.11    0.87    4.20
Базисные переменные: x2 x4 x6
Решение:
x2 = -4.59 + 2.63*x1 + 0.67*x3 + 0.61*x5
x4 = -1.62 + 1.43*x1 - 0.09*x3 + 0.36*x5
x6 = 4.20 - 3.54*x1 + 0.11*x3 - 0.87*x5
-------------------------
Комбинация базиса 16:
Преобразованная матрица:
    1.00    0.00    0.00   -0.22   -0.82   -1.68   -1.86
    0.00    1.00    0.00    3.96   -0.24   -2.76    4.48
    0.00    0.00    1.00    0.10    0.10    2.40    0.30
Базисные переменные: x2 x5 x6
Решение:
x2 = -1.86 + 0.22*x1 + 0.82*x3 + 1.68*x4
x5 = 4.48 - 3.96*x1 + 0.24*x3 + 2.76*x4
x6 = 0.30 - 0.10*x1 - 0.10*x3 - 2.40*x4
-------------------------
Комбинация базиса 17:
Преобразованная матрица:
    1.00    0.00    0.00    0.20   -1.33   -0.93    2.20
    0.00    1.00    0.00    0.03    0.06    0.46    0.03
    0.00    0.00    1.00    4.10   -0.17    1.03    5.10
Базисные переменные: x3 x4 x5
Решение:
x3 = 2.20 - 0.20*x1 + 1.33*x2 + 0.93*x6
x4 = 0.03 - 0.03*x1 - 0.06*x2 - 0.46*x6
x5 = 5.10 - 4.10*x1 + 0.17*x2 - 1.03*x6
-------------------------
Комбинация базиса 18:
Преобразованная матрица:
    1.00    0.00    0.00    3.90   -1.48    0.90    6.81
    0.00    1.00    0.00   -1.77    0.13   -0.44   -2.22
    0.00    0.00    1.00    3.97   -0.16    0.97    4.94
Базисные переменные: x3 x4 x6
Решение:
x3 = 6.81 - 3.90*x1 + 1.48*x2 - 0.90*x5
x4 = -2.22 + 1.77*x1 - 0.13*x2 + 0.44*x5
x6 = 4.94 - 3.97*x1 + 0.16*x2 - 0.97*x5
-------------------------
Комбинация базиса 19:
Преобразованная матрица:
    1.00    0.00    0.00    0.27   -1.22    2.05    2.27
    0.00    1.00    0.00    4.02   -0.29   -2.27    5.02
    0.00    0.00    1.00    0.07    0.12    2.20    0.07
Базисные переменные: x3 x5 x6
Решение:
x3 = 2.27 - 0.27*x1 + 1.22*x2 - 2.05*x4
x5 = 5.02 - 4.02*x1 + 0.29*x2 + 2.27*x4
x6 = 0.07 - 0.07*x1 - 0.12*x2 - 2.20*x4
-------------------------
Комбинация базиса 20:
Преобразованная матрица:
    1.00    0.00    0.00    0.13   -0.60    0.49    1.11
    0.00    1.00    0.00    4.32   -1.64    1.11    7.54
    0.00    0.00    1.00   -0.21    1.43   -1.07   -2.36
Базисные переменные: x4 x5 x6
Решение:
x4 = 1.11 - 0.13*x1 + 0.60*x2 - 0.49*x3
x5 = 7.54 - 4.32*x1 + 1.64*x2 - 1.11*x3
x6 = -2.36 + 0.21*x1 - 1.43*x2 + 1.07*x3
-------------------------
```

Описание программы:

1. Ввод данных:
   - Программа запрашивает количество уравнений и переменных.
   - Затем вводятся коэффициенты системы уравнений.
2. Метод Гаусса-Жордана:
   - Матрица приводится к приведённому ступенчатому виду.
   - Ведущие элементы (первые ненулевые элементы в строках) становятся равными 1.
   - Все остальные элементы в столбцах ведущих элементов обнуляются.
3. Анализ решения:
   - Если в процессе преобразований обнаруживается строка вида 0 0 ... 0 | b, где b != 0, система не имеет решений.
   - Если система имеет бесконечно много решений, программа указывает свободные переменные.
   - Если система имеет единственное решение, программа выводит его.

Задание 2

Организовать отбор опорных планов среди всех базисных решений, а также нахождение оптимального опорного плана методом прямого перебора. Целевая функция выбирается произвольно.

Пусть в нашем случае целевая функция будет равна сумме свободных членов в системе.

Код программы:

```cpp
#include <iostream>
#include <vector>
#include <cmath>
#include <algorithm>
#include <numeric>
#include <iomanip>

using namespace std;

const double EPS = 1e-9;

// Функция для перестановки столбцов в матрице
void rearrangeColumns(vector<vector<double>>& matrix, const vector<int>& basis) {
    int cols = matrix[0].size() - 1; // Исключаем столбец свободных членов
    for (auto& row : matrix) {
        vector<double> new_row;
        // Добавляем базисные столбцы
        for (int b : basis) new_row.push_back(row[b]);
        // Добавляем оставшиеся небазисные столбцы
        for (int c = 0; c < cols; ++c) {
            if (find(basis.begin(), basis.end(), c) == basis.end()) {
                new_row.push_back(row[c]);
            }
        }
        // Добавляем свободный член
        new_row.push_back(row.back());
        row = new_row;
    }
}

// Улучшенный метод Гаусса-Жордана с ведением журнала ведущих столбцов
void gauss_jordan(vector<vector<double>>& matrix, vector<int>& leading_columns) {
    int rows = matrix.size();
    int cols = matrix[0].size() - 1;
    leading_columns.clear();
    int lead = 0;
```

```
37
38      for (int r = 0; r < rows && lead < cols; ++r) {
39          // Поиск ненулевого элемента в столбце lead начиная с текущей строки
40          int i = r;
41          while (i < rows && abs(matrix[i][lead]) < EPS) ++i;
42          if (i == rows) {
43              ++lead;
44              --r;
45              continue;
46          }
47
48          swap(matrix[i], matrix[r]);
49          double div = matrix[r][lead];
50          for (int j = lead; j <= cols; ++j) matrix[r][j] /= div;
51
52          for (i = 0; i < rows; ++i) {
53              if (i != r) {
54                  double factor = matrix[i][lead];
55                  for (int j = lead; j <= cols; ++j)
56                  matrix[i][j] -= factor * matrix[r][j];
57              }
58          }
59          leading_columns.push_back(lead);
60          ++lead;
61      }
62  }
63
64  // Функция для проверки допустимости решения (все переменные >= 0)
65  bool is_feasible(const vector<double>& solution) {
66      for (double x : solution) {
67          if (x < -EPS) return false;
68      }
69      return true;
70  }
71
72  // Целевая функция (сумма свободных членов)
73  double objective_function(const vector<double>& solution) {
74      double sum = 0.0;
75      for (double x : solution) sum += x;
76      return sum;
77  }
78
79  // Функция для проверки, является ли базис допустимым
80  bool is_valid_basis(const vector<int>& basis, const vector<int>& leading_columns) {
81      // Проверяем, что все ведущие столбцы находятся в пределах базиса
82      for (int lc : leading_columns) {
83          if (lc >= basis.size()) return false;
```

```
    }
    return true;
}

// Функция для вывода решения с учетом переставленных столбцов
void print_solution(const vector<vector<double>>& matrix, const vector<int>& basis, const vector<dou
    cout << "Базисные переменные: ";
    for (int b : basis) cout << "x" << b + 1 << " ";
    cout << "\nРешение:\n";

    // Создаем список небазисных переменных
    vector<int> non_basis;
    for (int i = 0; i < matrix[0].size() - 1; ++i) {
        if (find(basis.begin(), basis.end(), i) == basis.end()) {
            non_basis.push_back(i);
        }
    }

    for (size_t r = 0; r < matrix.size(); ++r) {
        int lead_col = -1;
        for (size_t c = 0; c < basis.size(); ++c) {
            if (abs(matrix[r][c] - 1.0) < EPS) {
                lead_col = c;
                break;
            }
        }

        if (lead_col == -1) {
            if (abs(matrix[r].back()) > EPS)
            cout << "Система несовместна\n";
            continue;
        }

        int actual_var = basis[lead_col];
        cout << "x" << actual_var + 1 << " = " << fixed << setprecision(2) << matrix[r].back();

        for (size_t c = basis.size(); c < matrix[r].size() - 1; ++c) {
            double coeff = matrix[r][c];
            if (abs(coeff) > EPS) {
                int var_index = non_basis[c - basis.size()];
                if (coeff < 0) {
                    cout << " + " << fixed << setprecision(2) << -coeff << "*x" << var_index + 1;
                }
                else {
                    cout << " - " << fixed << setprecision(2) << coeff << "*x" << var_index + 1;
                }
            }
```

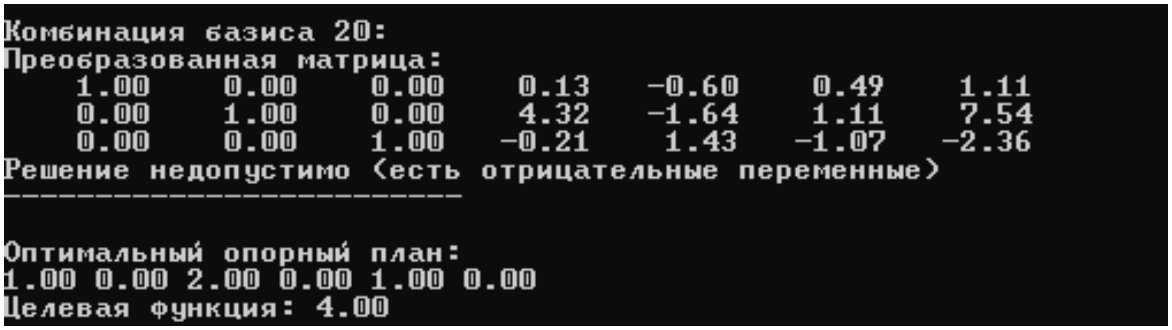
```
        }
        cout << endl;
    }
    cout << "-------------------------\n";
}

// Функция для нахождения всех опорных планов
void find_basic_solutions(vector<vector<double>> original, int num_vars) {
    int num_eqs = original.size();
    vector<int> vars(num_vars);
    iota(vars.begin(), vars.end(), 0);

    vector<bool> selector(num_vars);
    fill(selector.begin(), selector.begin() + num_eqs, true);

    vector<vector<double>> feasible_solutions;
    int count = 0;

    do {
        vector<int> basis;
        for (int i = 0; i < num_vars; ++i)
        if (selector[i]) basis.push_back(i);
        if (basis.size() != num_eqs) continue;

        vector<vector<double>> matrix = original;
        rearrangeColumns(matrix, basis);

        vector<int> leading_columns;
        gauss_jordan(matrix, leading_columns);

        cout << "Комбинация базиса " << ++count << ":\n";
        cout << "Преобразованная матрица:\n";
        for (const auto& row : matrix) {
            for (double val : row) cout << fixed << setprecision(2) << setw(8) << val;
            cout << endl;
        }

        if (is_valid_basis(basis, leading_columns)) {
            vector<double> solution(num_vars, 0.0);
            for (size_t r = 0; r < matrix.size(); ++r) {
                for (size_t c = 0; c < basis.size(); ++c) {
                    if (abs(matrix[r][c] - 1.0) < EPS) {
                        solution[basis[c]] = matrix[r].back();
                        break;
                    }
                }
            }
```

```
            if (is_feasible(solution)) {
                feasible_solutions.push_back(solution);
                print_solution(matrix, basis, solution);
                cout << "Целевая функция: " << objective_function(solution) << "\n";
                cout << "-------------------------\n";
            }
            else {
                cout << "Решение недопустимо (есть отрицательные переменные)\n";
                cout << "-------------------------\n";
            }
        }
        else {
            cout << "Не является допустимым базисом\n";
            cout << "-------------------------\n";
        }

    } while (prev_permutation(selector.begin(), selector.end()));

    // Поиск оптимального решения
    if (!feasible_solutions.empty()) {
        auto optimal_solution = *min_element(feasible_solutions.begin(), feasible_solutions.end(),
        [](const vector<double>& a, const vector<double>& b) {
            return objective_function(a) < objective_function(b);
        });

        cout << "\nОптимальный опорный план:\n";
        for (double x : optimal_solution) cout << fixed << setprecision(2) << x << " ";
        cout << "\nЦелевая функция: " << objective_function(optimal_solution) << "\n";
    }
    else {
        cout << "\nДопустимые решения отсутствуют!\n";
    }
}

int main() {
    setlocale(LC_ALL, "Russian");
    vector<vector<double>> matrix = {
        {-1, 5, -4, -6, 0, 1, -9},
        {8, 1, -1, 0, 2, 3, 8},
        {4, 3, -2, 9, 1, 7, 1}
    };
    find_basic_solutions(matrix, 6);
    return 0;
}
```

Результаты работы алгоритма:

```
Комбинация базиса 1:
Преобразованная матрица:
    1.00    0.00    0.00    0.71    0.24    0.57    1.24
    0.00    1.00    0.00   17.57   -0.14    7.86   -0.14
    0.00    0.00    1.00   23.29   -0.24    9.43    1.76
Решение недопустимо (есть отрицательные переменные)
-------------------------
Комбинация базиса 2:
Преобразованная матрица:
    1.00    0.00    0.00   -0.03    0.25    0.28    1.18
    0.00    1.00    0.00   -0.75    0.04    0.74   -1.47
    0.00    0.00    1.00    0.04   -0.01    0.40    0.08
Решение недопустимо (есть отрицательные переменные)
-------------------------
Комбинация базиса 3:
Преобразованная матрица:
    1.00    0.00    0.00    1.00   24.00   10.00    3.00
    0.00    1.00    0.00   -0.60    3.60    2.20   -1.20
    0.00    0.00    1.00   -4.20  -97.80  -39.60   -7.40
Решение недопустимо (есть отрицательные переменные)
-------------------------
Комбинация базиса 4:
Преобразованная матрица:
    1.00    0.00    0.00   -0.06   -0.70    0.25    1.13
    0.00    1.00    0.00   -0.83   -1.83    0.06   -1.61
    0.00    0.00    1.00    0.11    2.47   -0.03    0.19
Решение недопустимо (есть отрицательные переменные)
-------------------------
Комбинация базиса 5:
Преобразованная матрица:
    1.00    0.00    0.00   -0.04    0.24    0.25    1.24
    0.00    1.00    0.00   -1.33   -0.05   -0.98    1.95
    0.00    0.00    1.00    0.06   -0.01    0.45   -0.01
Решение недопустимо (есть отрицательные переменные)
-------------------------
Комбинация базиса 6:
Преобразованная матрица:
    1.00    0.00    0.00    1.67   30.00   13.67    1.00
    0.00    1.00    0.00   -1.67   -6.00   -3.67    2.00
    0.00    0.00    1.00   -7.00 -123.00  -55.00    1.00
Базисные переменные: x1 x3 x5
Решение:
x1 = 1.00 - 1.67*x2 - 30.00*x4 - 13.67*x6
x3 = 2.00 + 1.67*x2 + 6.00*x4 + 3.67*x6
x5 = 1.00 + 7.00*x2 + 123.00*x4 + 55.00*x6
-------------------------
Целевая функция: 4.00
-------------------------
Комбинация базиса 7:
Преобразованная матрица:
    1.00    0.00    0.00   -0.07   -0.56    0.25    1.25
    0.00    1.00    0.00   -1.20    2.20   -0.07    1.93
    0.00    0.00    1.00    0.13    2.24   -0.02   -0.02
Решение недопустимо (есть отрицательные переменные)
-------------------------
Комбинация базиса 8:
Преобразованная матрица:
    1.00    0.00    0.00   -6.67    5.00   -4.67   11.00
    0.00    1.00    0.00    0.28   -0.17    0.61   -0.33
    0.00    0.00    1.00   27.17  -20.50   20.17  -40.00
Решение недопустимо (есть отрицательные переменные)
-------------------------
Комбинация базиса 9:
Преобразованная матрица:
    1.00    0.00    0.00   -0.38    0.26    0.23    1.74
    0.00    1.00    0.00   -0.55    0.45   -0.03    0.88
    0.00    0.00    1.00    1.35   -1.02    0.05   -1.98
Решение недопустимо (есть отрицательные переменные)
-------------------------
Комбинация базиса 10:
Преобразованная матрица:
    1.00    0.00    0.00   -4.55    3.73    7.64    8.45
    0.00    1.00    0.00   18.00  -15.00  -33.00  -29.00
    0.00    0.00    1.00    0.45   -0.27    1.64   -0.55
Решение недопустимо (есть отрицательные переменные)
-------------------------
Комбинация базиса 11:
Преобразованная матрица:
    1.00    0.00    0.00  -24.60   -6.00   -6.20  -30.60
    0.00    1.00    0.00  -32.60   -8.00   -9.20  -38.60
    0.00    0.00    1.00    1.40    0.33    0.80    1.73
Решение недопустимо (есть отрицательные переменные)
-------------------------
```

```
-------------------------
Комбинация базиса 12:
Преобразованная матрица:
    1.00    0.00    0.00    0.60   18.00    8.20    0.60
    0.00    1.00    0.00    1.00   24.00   10.00    3.00
    0.00    0.00    1.00    4.20    3.00    2.40    5.20
Базисные переменные: x2 x3 x5
Решение:
x2 = 0.60 - 0.60*x1 - 18.00*x4 - 8.20*x6
x3 = 3.00 - 1.00*x1 - 24.00*x4 - 10.00*x6
x5 = 5.20 - 4.20*x1 - 3.00*x4 - 2.40*x6
-------------------------
Целевая функция: 8.80
-------------------------
Комбинация базиса 13:
Преобразованная матрица:
    1.00    0.00    0.00  -13.75    7.75   -3.42  -17.17
    0.00    1.00    0.00  -16.50   11.50   -4.17  -18.67
    0.00    0.00    1.00    1.75    1.25    0.42    2.17
Решение недопустимо (есть отрицательные переменные)
-------------------------
Комбинация базиса 14:
Преобразованная матрица:
    1.00    0.00    0.00   -0.15   -0.75    0.70   -1.65
    0.00    1.00    0.00    0.04    0.04    0.42    0.12
    0.00    0.00    1.00    4.08   -0.12    1.15    4.83
Решение недопустимо (есть отрицательные переменные)
-------------------------
Комбинация базиса 15:
Преобразованная матрица:
    1.00    0.00    0.00   -2.63   -0.67   -0.61   -4.59
    0.00    1.00    0.00   -1.43    0.09   -0.36   -1.62
    0.00    0.00    1.00    3.54   -0.11    0.87    4.20
Решение недопустимо (есть отрицательные переменные)
-------------------------
Комбинация базиса 16:
Преобразованная матрица:
    1.00    0.00    0.00   -0.22   -0.82   -1.68   -1.86
    0.00    1.00    0.00    3.96   -0.24   -2.76    4.48
    0.00    0.00    1.00    0.10    0.10    2.40    0.30
Решение недопустимо (есть отрицательные переменные)
-------------------------
Комбинация базиса 17:
Преобразованная матрица:
    1.00    0.00    0.00    0.20   -1.33   -0.93    2.20
    0.00    1.00    0.00    0.03    0.06    0.46    0.03
    0.00    0.00    1.00    4.10   -0.17    1.03    5.10
Базисные переменные: x3 x4 x5
Решение:
x3 = 2.20 - 0.20*x1 + 1.33*x2 + 0.93*x6
x4 = 0.03 - 0.03*x1 - 0.06*x2 - 0.46*x6
x5 = 5.10 - 4.10*x1 + 0.17*x2 - 1.03*x6
-------------------------
Целевая функция: 7.33
-------------------------
Комбинация базиса 18:
Преобразованная матрица:
    1.00    0.00    0.00    3.90   -1.48    0.90    6.81
    0.00    1.00    0.00   -1.77    0.13   -0.44   -2.22
    0.00    0.00    1.00    3.97   -0.16    0.97    4.94
Решение недопустимо (есть отрицательные переменные)
-------------------------
Комбинация базиса 19:
Преобразованная матрица:
    1.00    0.00    0.00    0.27   -1.22    2.05    2.27
    0.00    1.00    0.00    4.02   -0.29   -2.27    5.02
    0.00    0.00    1.00    0.07    0.12    2.20    0.07
Базисные переменные: x3 x5 x6
Решение:
x3 = 2.27 - 0.27*x1 + 1.22*x2 - 2.05*x4
x5 = 5.02 - 4.02*x1 + 0.29*x2 + 2.27*x4
x6 = 0.07 - 0.07*x1 - 0.12*x2 - 2.20*x4
-------------------------
Целевая функция: 7.37
-------------------------
```



Выводятся все базисные решения, но если они не являются опорными, то в рамках условия данной задачи их решение недопустимо. Для опорных планов выводится решение и значение целевой функции. Оптимальный опорный план выбирается по наименьшему значению целевой функции.

Описание программы:

1. Функция gauss_jordan:
   - Приводит расширенную матрицу системы к упрощённому ступенчатому виду.
2. Функция is_feasible:
   - Проверяет, является ли решение опорным планом (все переменные неотрицательны).
3. Функция objective_function:
   - Задаёт целевую функцию
4. Функция find_optimal_plan:
   - Находит все базисные решения.
   - Отбирает опорные планы.
   - Находит оптимальный опорный план, минимизирующий целевую функцию.

Задание 3

Решить систему линейных уравнений аналитически (подготовить тестовые данные)

$$\begin{cases} -x_1 + 5x_2 - 4x_3 - 6x_4 + x_6 = -9 \\ 8x_1 + x_2 - x_3 + 2x_5 + 3x_6 = 8 \\ 4x_1 + 3x_2 - 2x_3 + 9x_4 + x_5 + 7x_6 = 1 \end{cases}$$

- Так как n > m (переменных больше чем уравнений, то система имеет бесконечно много решений)
- Расширенная матрица:

$$\left(\begin{array}{cccccc|c} -1 & 5 & -4 & -6 & 0 & 1 & -9 \\ 8 & 1 & -1 & 0 & 2 & 3 & 8 \\ 4 & 3 & -2 & 9 & 1 & 7 & 1 \end{array}\right)$$

- Приведение матрицы:

1. a_{11} = -1
   - Делим первую строку на -1:

$$\left(\begin{array}{cccccc|c} 1 & -5 & 4 & 6 & 0 & -1 & 9 \\ 8 & 1 & -1 & 0 & 2 & 3 & 8 \\ 4 & 3 & -2 & 9 & 1 & 7 & 1 \end{array}\right)$$

   - Умножаем первую строку на 8 и вычитаем из второй:

$$\left(\begin{array}{cccccc|c} 1 & -5 & 4 & 6 & 0 & -1 & 9 \\ 0 & 41 & -33 & -48 & 2 & 11 & -64 \\ 4 & 3 & -2 & 9 & 1 & 7 & 1 \end{array}\right)$$

   - Умножаем первую строку на 4 и вычитаем из третьей:

$$\left(\begin{array}{cccccc|c} 1 & -5 & 4 & 6 & 0 & -1 & 9 \\ 0 & 41 & -33 & -48 & 2 & 11 & -64 \\ 0 & 23 & -18 & -15 & 1 & 11 & -35 \end{array}\right)$$

2. $a_{22} = 41$
   - Делим вторую строку на 41:

$$\left(\begin{array}{cccccc|c} 1 & -5 & 4 & 6 & 0 & -1 & 9 \\ 0 & 1 & -\frac{33}{41} & -\frac{48}{41} & \frac{2}{41} & \frac{11}{41} & -\frac{64}{41} \\ 0 & 23 & -18 & -15 & 1 & 11 & -35 \end{array}\right)$$

   - Умножаем вторую строку на -5 и вычитаем из первой:

$$\left(\begin{array}{cccccc|c} 1 & 0 & -\frac{1}{41} & \frac{6}{41} & \frac{10}{41} & \frac{14}{41} & \frac{49}{41} \\ 0 & 1 & -\frac{33}{41} & -\frac{48}{41} & \frac{2}{41} & \frac{11}{41} & -\frac{64}{41} \\ 0 & 23 & -18 & -15 & 1 & 11 & -35 \end{array}\right)$$

   - Умножаем вторую строку на 23 и вычитаем из третьей:

$$\left(\begin{array}{cccccc|c} 1 & 0 & -\frac{1}{41} & \frac{6}{41} & \frac{10}{41} & \frac{14}{41} & \frac{49}{41} \\ 0 & 1 & -\frac{33}{41} & -\frac{48}{41} & \frac{2}{41} & \frac{11}{41} & -\frac{64}{41} \\ 0 & 0 & \frac{21}{41} & \frac{489}{41} & -\frac{5}{41} & \frac{198}{41} & \frac{37}{41} \end{array}\right)$$

3. $a_{33} = \frac{21}{41}$
   - Делим третью строку на $\frac{21}{41}$:

$$\left(\begin{array}{cccccc|c} 1 & 0 & -\frac{1}{41} & \frac{6}{41} & \frac{10}{41} & \frac{14}{41} & \frac{49}{41} \\ 0 & 1 & -\frac{33}{41} & -\frac{48}{41} & \frac{2}{41} & \frac{11}{41} & -\frac{64}{41} \\ 0 & 0 & 1 & \frac{163}{7} & -\frac{5}{21} & \frac{66}{7} & \frac{37}{21} \end{array}\right)$$

   - Умножаем третью строку на $-\frac{33}{41}$ и вычитаем из второй:

$$\left(\begin{array}{cccccc|c} 1 & 0 & -\frac{1}{41} & \frac{6}{41} & \frac{10}{41} & \frac{14}{41} & \frac{49}{41} \\ 0 & 1 & 0 & -\frac{123}{7} & -\frac{1}{7} & \frac{55}{7} & -\frac{1}{7} \\ 0 & 0 & 1 & \frac{163}{7} & -\frac{5}{21} & \frac{66}{7} & \frac{37}{21} \end{array}\right)$$

   - Умножаем третью строку на $-\frac{1}{41}$ и вычитаем из первой:

$$\left(\begin{array}{cccccc|c} 1 & 0 & 0 & \frac{5}{7} & \frac{5}{21} & \frac{4}{7} & \frac{26}{21} \\ 0 & 1 & 0 & -\frac{123}{7} & -\frac{1}{7} & \frac{55}{7} & -\frac{1}{7} \\ 0 & 0 & 1 & \frac{163}{7} & -\frac{5}{21} & \frac{66}{7} & \frac{37}{21} \end{array}\right)$$

- Матрица в упрощенном ступенчатом виде:

$$\left(\begin{array}{cccccc|c} 1 & 0 & 0 & \frac{5}{7} & \frac{5}{21} & \frac{4}{7} & \frac{26}{21} \\ 0 & 1 & 0 & -\frac{123}{7} & -\frac{1}{7} & \frac{55}{7} & -\frac{1}{7} \\ 0 & 0 & 1 & \frac{163}{7} & -\frac{5}{21} & \frac{66}{7} & \frac{37}{21} \end{array}\right)$$

- После всех преобразований матрица имеет вид:

$$\left(\begin{array}{ccccccc} 1 & 0 & 0 & a_{14} & a_{15} & a_{16} & b_1 \\ 0 & 1 & 0 & a_{24} & a_{25} & a_{26} & b_3 \\ 0 & 0 & 1 & a_{34} & a_{35} & a_{36} & b_2 \end{array}\right)$$

- Базисные переменные: x_1, x_2, x_3. Выражаем их через свободные x_4, x_5, x_6:

$$\begin{cases} x_1 = b_1 - a_{14}x_4 - a_{15}x_5 - a_{16}x_6 \\ x_2 = b_2 - a_{24}x_4 - a_{25}x_5 - a_{26}x_6 \\ x_3 = b_3 - a_{34}x_4 - a_{35}x_5 - a_{36}x_6 \end{cases}$$

- Итоговое решение:

$$\begin{cases} x_1 = \frac{26}{21} - \frac{5}{7}x_4 - \frac{5}{21}x_5 - \frac{4}{7}x_6 \\ x_2 = -\frac{1}{7} - \frac{123}{7}x_4 + \frac{1}{7}x_5 - \frac{55}{7}x_6 \\ x_3 = \frac{37}{21} - \frac{163}{7}x_4 + \frac{5}{21}x_5 - \frac{66}{7}x_6 \end{cases}$$

Найдем один из опорных планов (Всего их $C_6^3 = 20$):

- Для базисных переменных x_1, x_2, x_4
- Расширенная матрица:

$$\left(\begin{array}{cccccc|c} -1 & 5 & -6 & -4 & 0 & 1 & -9 \\ 8 & 1 & 0 & -1 & 2 & 3 & 8 \\ 4 & 3 & 9 & -2 & 1 & 7 & 1 \end{array}\right)$$

1. a_{11} = -1
   - Делим первую строку на -1:

$$\left(\begin{array}{cccccc|c} 1 & -5 & 6 & 4 & 0 & -1 & 9 \\ 8 & 1 & 0 & -1 & 2 & 3 & 8 \\ 4 & 3 & 9 & -2 & 1 & 7 & 1 \end{array}\right)$$

   - Умножаем первую строку на 8 и вычитаем из второй:

$$\left(\begin{array}{cccccc|c} 1 & -5 & 6 & 4 & 0 & -1 & 9 \\ 0 & 41 & -48 & -33 & 2 & 11 & -64 \\ 4 & 3 & 9 & -2 & 1 & 7 & 1 \end{array}\right)$$

   - Умножаем первую строку на 4 и вычитаем из третьей:

$$\left(\begin{array}{cccccc|c} 1 & -5 & 6 & 4 & 0 & -1 & 9 \\ 0 & 41 & -48 & -33 & 2 & 11 & -64 \\ 0 & 23 & -15 & -18 & 1 & 11 & -35 \end{array}\right)$$

2. a_{22} = 41
   - Делим вторую строку на 41:

$$\left(\begin{array}{cccccc|c} 1 & -5 & 6 & 4 & 0 & -1 & 9 \\ 0 & 1 & -\frac{48}{41} & -\frac{33}{41} & \frac{2}{41} & \frac{11}{41} & -\frac{64}{41} \\ 0 & 23 & -15 & -18 & 1 & 11 & -35 \end{array}\right)$$

   - Умножаем вторую строку на -5 и вычитаем из первой:

$$\left(\begin{array}{cccccc|c} 1 & 0 & \frac{6}{41} & -\frac{1}{41} & \frac{10}{41} & \frac{14}{41} & \frac{49}{41} \\ 0 & 1 & -\frac{48}{41} & -\frac{33}{41} & \frac{2}{41} & \frac{11}{41} & -\frac{64}{41} \\ 0 & 23 & -15 & -18 & 1 & 11 & -35 \end{array}\right)$$

   - Умножаем вторую строку на 23 и вычитаем из третьей:

$$\left(\begin{array}{cccccc|c} 1 & 0 & \frac{6}{41} & -\frac{1}{41} & \frac{10}{41} & \frac{14}{41} & \frac{49}{41} \\ 0 & 1 & -\frac{48}{41} & -\frac{33}{41} & \frac{2}{41} & \frac{11}{41} & -\frac{64}{41} \\ 0 & 0 & \frac{489}{41} & \frac{21}{41} & -\frac{5}{41} & \frac{198}{41} & \frac{37}{41} \end{array}\right)$$

3. $a_{33} = \frac{21}{41}$
   - Делим третью строку на $\frac{489}{41}$:

$$\left(\begin{array}{ccccccc} 1 & 0 & \frac{6}{41} & -\frac{1}{41} & \frac{10}{41} & \frac{14}{41} & \left| \begin{array}{c} \frac{49}{41} \end{array} \right. \\ 0 & 1 & -\frac{48}{41} & -\frac{33}{41} & \frac{2}{41} & \frac{11}{41} & \left| \begin{array}{c} -\frac{64}{41} \end{array} \right. \\ 0 & 0 & 1 & \frac{7}{163} & -\frac{5}{489} & \frac{66}{163} & \left| \begin{array}{c} \frac{37}{489} \end{array} \right. \end{array}\right)$$

   - Умножаем третью строку на $-\frac{48}{41}$ и вычитаем из второй:

$$\left(\begin{array}{ccccccc} 1 & 0 & \frac{6}{41} & -\frac{1}{41} & \frac{10}{41} & \frac{14}{41} & \left| \begin{array}{c} \frac{49}{41} \end{array} \right. \\ 0 & 1 & 0 & -\frac{123}{163} & \frac{6}{163} & \frac{121}{163} & \left| \begin{array}{c} \frac{240}{163} \end{array} \right. \\ 0 & 0 & 1 & \frac{7}{163} & -\frac{5}{489} & \frac{66}{163} & \left| \begin{array}{c} \frac{37}{489} \end{array} \right. \end{array}\right)$$

   - Умножаем третью строку на $-\frac{6}{41}$ и вычитаем из первой:

$$\left(\begin{array}{ccccccc} 1 & 0 & 0 & \frac{1020}{6683} & -\frac{173}{6683} & \frac{2678}{6683} & \left| \begin{array}{c} \frac{8061}{6683} \end{array} \right. \\ 0 & 1 & 0 & -\frac{123}{163} & \frac{6}{163} & \frac{121}{163} & \left| \begin{array}{c} \frac{240}{163} \end{array} \right. \\ 0 & 0 & 1 & \frac{7}{163} & -\frac{5}{489} & \frac{66}{163} & \left| \begin{array}{c} \frac{37}{489} \end{array} \right. \end{array}\right)$$

- После всех преобразований матрица имеет вид:

$$\left(\begin{array}{ccccccc} 1 & 0 & 0 & a_{14} & a_{15} & a_{16} & \left| \begin{array}{c} b_1 \end{array} \right. \\ 0 & 1 & 0 & a_{24} & a_{25} & a_{26} & \left| \begin{array}{c} b_3 \end{array} \right. \\ 0 & 0 & 1 & a_{34} & a_{35} & a_{36} & \left| \begin{array}{c} b_2 \end{array} \right. \end{array}\right)$$

- Базисные переменные: x_1, x_2, x_3. Выражаем их через свободные x_4, x_5, x_6:

$$\begin{cases} x_1 = b_1 - a_{14}x_4 - a_{15}x_5 - a_{16}x_6 \\ x_2 = b_2 - a_{24}x_4 - a_{25}x_5 - a_{26}x_6 \\ x_3 = b_3 - a_{34}x_4 - a_{35}x_5 - a_{36}x_6 \end{cases}$$

- Итоговое решение:

$$\begin{cases} x_1 = \frac{8061}{6683} - \frac{1020}{6683}x_4 + \frac{173}{6683}x_5 - \frac{2678}{6683}x_6 \\ x_2 = \frac{240}{163} + \frac{123}{163}x_4 - \frac{6}{163}x_5 - \frac{121}{163}x_6 \\ x_3 = \frac{37}{489} - \frac{7}{163}x_4 + \frac{5}{489}x_5 - \frac{66}{163}x_6 \end{cases}$$

Вывод: в ходе выполнения лабораторной работы я составил программу для отыскания всех базисных решений системы уравнений с помощью метода Гаусса-Жордана, вывод которой совпал с ответом в моем аналитическом решении.