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-223
Игнатьев Артур Олегович

Проверил:
проф. Вирченко Юрий Петрович

Белгород 2024г.

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.

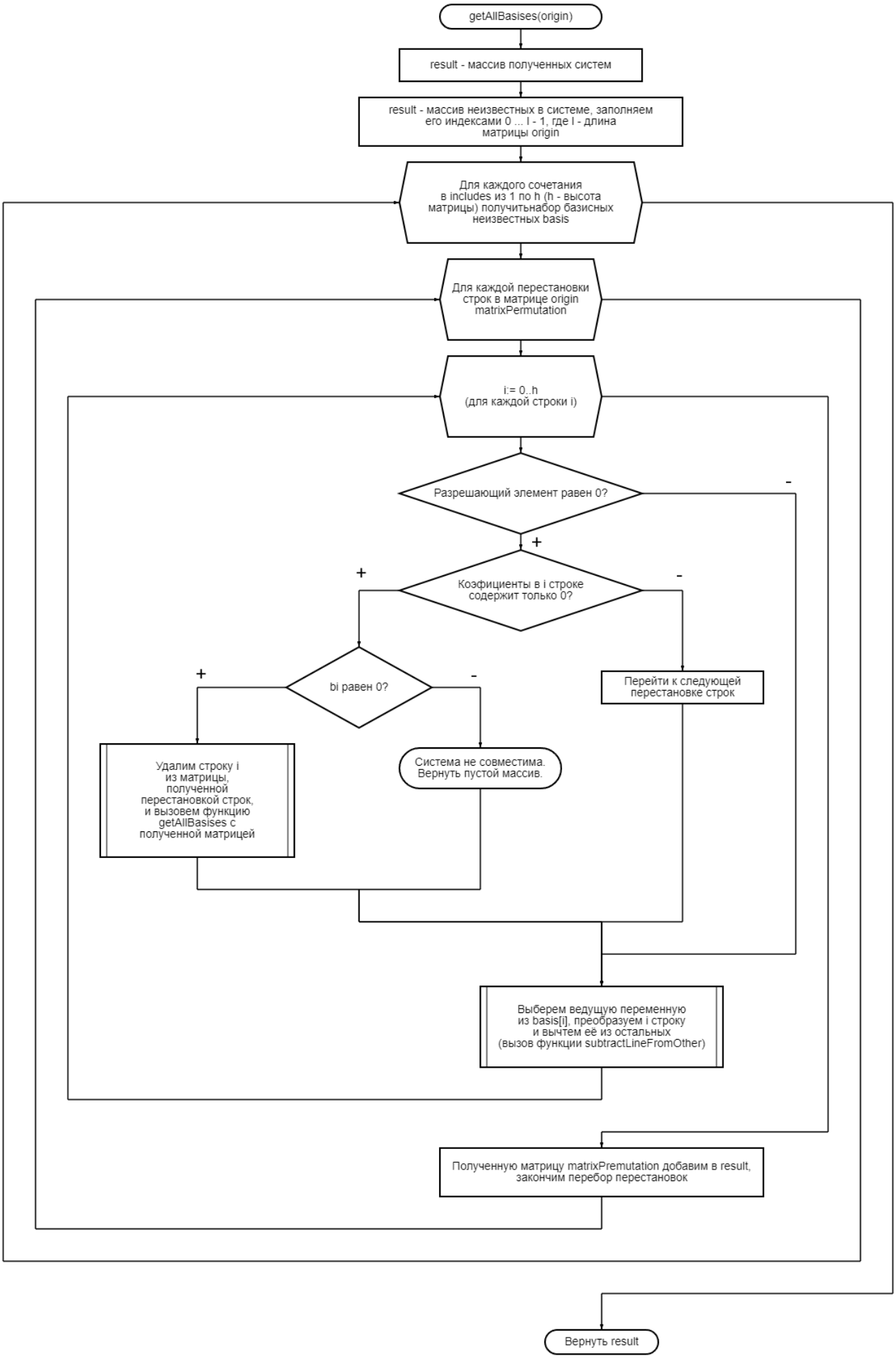
Вариант 3

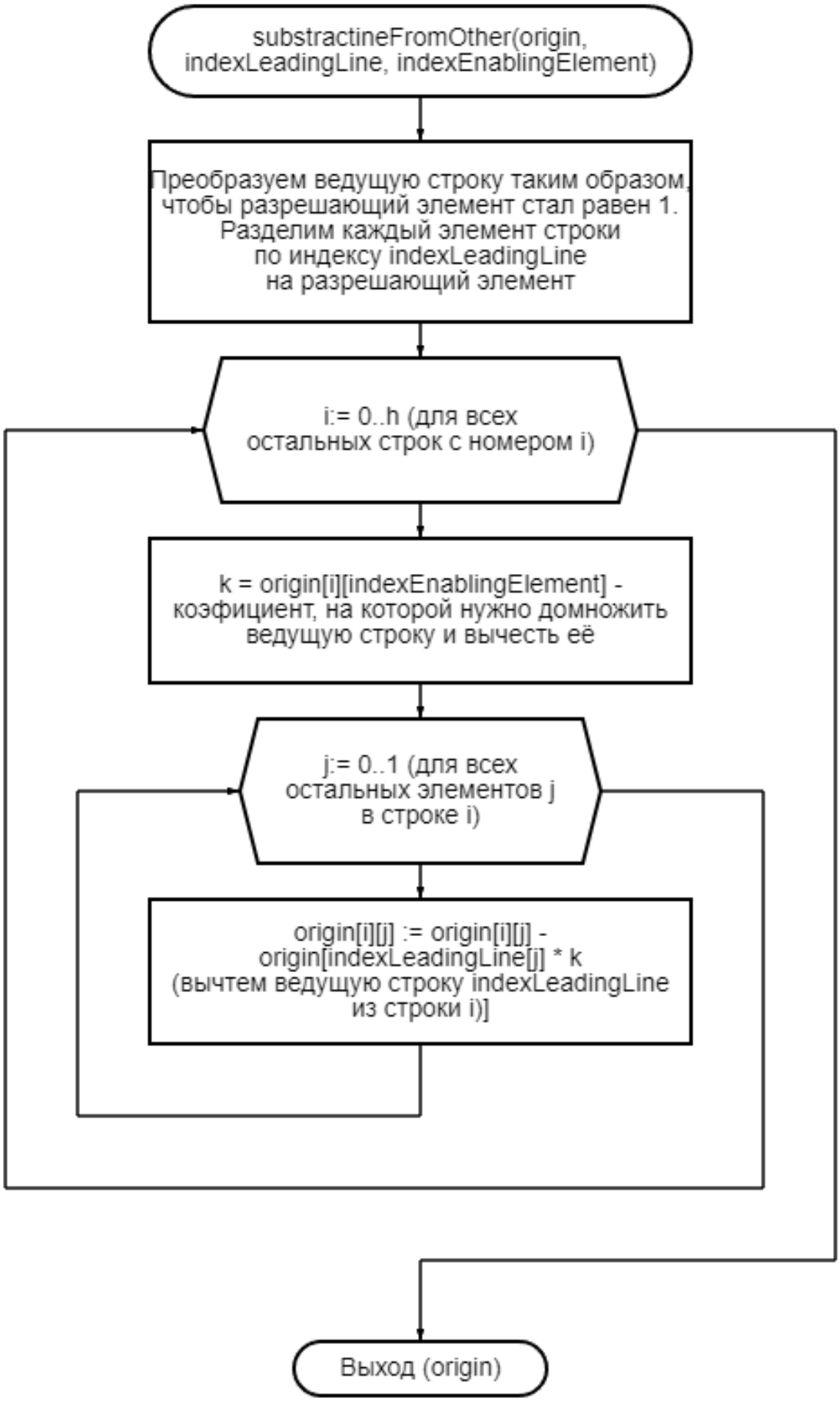
$$\begin{cases} 2x_1 - x_2 + 6x_3 - x_4 + 3x_5 = 12 \\ 3x_1 + 5x_2 + x_3 - 12x_4 + 2x_5 = 14 \\ -3x_1 + 6x_2 + 8x_3 + 7x_4 - 4x_5 = 18 \end{cases}$$

Ход выполнения лабораторной работы:

1. Составить программу для отыскания всех базисных видов системы линейных уравнений.

Блок-схемы


getAllBasises(origin)
result - массив полученных систем
result - массив неизвестных в системе, заполняем его индексами 0 ... l - 1, где l - длина матрицы origin
Для каждого сочетания в includes из 1 по h (h - высота матрицы) получитьнабор базисных неизвестных basis
Для каждой перестановки строк в матрице origin matrixPermutation
i:= 0..h (для каждой строки i)
Разрешающий элемент равен 0?
+
-
Коэфициенты в i строке содержит только 0?
+
-
bi равен 0?
+
-
Перейти к следующей перестановке строк
Удалим строку i из матрицы, полученной перестановкой строк, и вызовем функцию getAllBasises с полученной матрицей
Система не совместима. Вернуть пустой массив.
Выберем ведущую переменную из basis[i], преобразуем i строку и вычтем её из остальных (вызов функции subtractLineFromOther)
Полученную матрицу matrixPremutation добавим в result, закончим перебор перестановок
Вернуть result


substractineFromOther(origin, indexLeadingLine, indexEnablingElement)
Преобразуем ведущую строку таким образом, чтобы разрешающий элемент стал равен 1. Разделим каждый элемент строки по индексу indexLeadingLine на разрешающий элемент
i:= 0..h (для всех остальных строк с номером i)
k = origin[i][indexEnablingElement] - коэфициент, на которой нужно домножить ведущую строку и вычесть её
j:= 0..1 (для всех остальных элементов j в строке i)
origin[i][j] := origin[i][j] - origin[indexLeadingLine[j] * k (вычтем ведущую строку indexLeadingLine из строки i)]
Выход (origin)

Исходный код программы:

Файл task1.tpp

```
#pragma once

#include <vector>
#include <cstdlib>
#include <stdexcept>

#define EPS 0.00000000001

// Сочетания
template<typename T>
std::vector<std::vector<T>> getCombinations(std::vector<T> &baseSet,
std::vector<T> currentSet, size_t minIndex, size_t k, size_t count) {
    std::vector<std::vector<T>> resultCombs;

    // Если количество перестановок равно необходимому, мы достигли искомого
множества, возвращаем его
    if (count >= k)
        return {currentSet};

    for (size_t i = minIndex; i <= baseSet.size() - k + count; i++) {
        // Добавляем в текущее множество новый элемент, Ci = x
        std::vector<T> newCurrentSet(currentSet);
        newCurrentSet.push_back(baseSet[i]);

        // Вызываем следующий шаг итерации, сохраняем его результат в общий
массив множеств
        std::vector<std::vector<T>> combinations = getCombinations(baseSet,
newCurrentSet, i + 1, k, count + 1);
        resultCombs.insert(std::begin(resultCombs), std::begin(combinations),
std::end(combinations));
    }

    // Возвращаем массив множеств
    return resultCombs;
}

// Функция-обёртка для рекуррентной функции
template<typename T> std::vector<std::vector<T>> getCombinations(std::vec-
tor<T> &baseSet, size_t k) {
    return getCombinations(baseSet, {}, 0, k, 0);
}

// Перестановка
template<typename T>
std::vector<std::vector<T>> getPermutations(std::vector<T> baseSet, std::vec-
tor<T> currentSet) {
    std::vector<std::vector<T>> resultPerms;

    // Если элементов в изначальном множестве не осталось, получено искомое
множество
    if (baseSet.size() == 0) return {currentSet};

    for (size_t i = 0; i < baseSet.size(); i++) {
        // Удаляем из исходного массива x
        std::vector<T> newBaseSet(baseSet);
        newBaseSet.erase(std::begin(newBaseSet) + i);

        // Добавляем в текущее множество новый элемент
        std::vector<T> newCurrentSet(currentSet);
```

```
        newCurrentSet.push_back(baseSet[i]);

        // Выполняем следующий шаг итерации, сохраняем в итоговый массив мно-
жеств
        auto permutations = getPermutations(newBaseSet, newCurrentSet);
        resultPerms.insert(std::begin(resultPerms), std::begin(permutations),
std::end(permutations));
    }

    return resultPerms;
}

template<typename T>
std::vector<std::vector<T>> getPermutations(std::vector<T> &baseSet) {
    return getPermutations(baseSet, {});
}

template <std::size_t T>
void subtractLineFromOther(std::vector<std::array<double, T>>& origin, int
indexLeadingLine, int indexEnablingElement) {
    // Преобразуем ведущую строку таким образом, чтобы разрещающий элемент
стал равен 1.
    double originEnablingElement = origin[indexLeadingLine][indexEnablingEle-
ment];
    for (int i = 0; i < origin[indexLeadingLine].size(); i++)
        // Разделим каждый элемент строки по индексу indexLeadingLine
        // на разрешающий элемент
        origin[indexLeadingLine][i] /= originEnablingElement;

    // Для всех остальных строк с номером i
    for (int i = 0; i < origin.size(); i++) {
        if (i == indexLeadingLine) continue;

        // k - коэффициент, на который нужно домножить ведущую строку и вы-
честь её
        double k = origin[i][indexEnablingElement];

        // Для всех остальных элементов j в строке i
        for (int j = 0; j < origin[indexLeadingLine].size(); j++)
            // Вычесть ведущую строку indexLeadingLine из строки i
            origin[i][j] -= origin[indexLeadingLine][j] * k;
    }
}

// Вспомогательная СД
template <std::size_t T>
struct Basis {
    std::vector<int> indices;
    std::vector<std::array<double, T>> matrix;
};

template <std::size_t T>
std::vector<Basis<T>> getAllBasises(std::vector<std::array<double, T>>
origin) {
    // result - массив полученных систем
    std::vector<Basis<T>> result;

    // indices - массив неизвестных в системе, заполняем
    // его индексами 0 ... l - 1, где l - длина
    // матрицы origin
    std::vector<int> indices;
    for (int i = 0; i < origin[0].size() - 1; i++)
        indices.push_back(i);
```

```
    /*
    Для каждого сочетания в indices из l по h (h - высота матрицы)
    получим набор базисных неизвестных basis.
    */
    for (auto basis : getCombinations(indices, origin.size())) {
        // Для каждой перестановки строк в матрице origin matrixPermutation
        for (auto matrixPermutation : getPermutations(origin)) {
            auto copyMatrixPermutation = matrixPermutation;
            bool badPermutation = false;

            // Для каждой строки в матрице (ввод счётчика строк i)
            for (int i = 0; i < matrixPermutation.size(); i++) {
                // Разрешающий элемент равен 0?
                if (std::abs(matrixPermutation[i][basis[i]]) < EPS) {
                    bool allZeros = true;
                    for (int j = 0; (j < matrixPermutation[i].size() - 1) &&
allZeros; j++) {
                        if (std::abs(matrixPermutation[i][j]) > EPS) {
                            // Неудачное расположение строк, необходимо
                            // перейти к другой перестановке
                            badPermutation = true;
                            allZeros = false;
                            break;
                        }
                    }

                    // Коэффициенты в i строке содержат только 0?
                    if (!allZeros) {
                        // Перейти к следующей перестановке строк
                        break;
                    }

                    // b_i равен 0?
                    if (std::abs(matrixPermutation[i].back()) > EPS) {
                        // Система несовместима. Вернуть пустой массив.
                        return {};
                    }

                    /*
                    Удалить строку i из матрицы, полученной перестановкой
строк,
                    и вызвать функцию getAllBasises с полученной матрицей.
                    */
                    if (allZeros) {

copyMatrixPermutation.erase(copyMatrixPermutation.begin() + i);
                        return getAllBasises(copyMatrixPermutation);
                    }

                    break;
                }

                // Выберем ведущую переменную из basis[i], преобразуем i
строку и вычтем её из остальных
                subtractLineFromOther(matrixPermutation, i, basis[i]);
            }

            if (badPermutation) continue;

            // Полученную матрицу matrixPermutation добавим в result, закон-
чим перебор перестановок
            result.push_back({basis, matrixPermutation});
            break;
        }
```

```
    }

    return result;
}
```

Файл task1.cpp

```
#include <iostream>
#include <iomanip>
#include <array>
#include <windows.h>

#include "../../libs/alg/labs/lab1/task1.tpp"

int main() {
    SetConsoleOutputCP(CP_UTF8);
    // Инициализируем матрицу
    std::vector<std::array<double, 6>> matrix = {
            { 2, -1,  6,  -1,  3,  12},
            { 3,  5,  1, -12,  2,  14},
            {-3,  6,  8,   7, -4,  18}};;

    // Получаем базисные решения
    auto res = getAllBasises(matrix);

    // Выводим базисные решения
    std::cout <<
"=======================================================================
=======================================\n";
    for (auto& matrix : res) {
        std::cout << "Выбранные базисные переменные: ";
        for (auto& bas : matrix.indices) {
            std::cout << "x" << (bas + 1) << " ";
        }
        std::cout << "\n\nПолученная система: " << std::endl;


        for (int i = 0; i < matrix.matrix[0].size() - 1; i++) {
            std::stringstream buf;
            buf << "a" << (i + 1);
            std::cout << std::setw(15) << buf.str() << " ";
        }

        std::cout << std::setw(15) << "b" << std::endl;

        for (auto & line : matrix.matrix) {
            for (auto & element : line) {
                std::cout << std::setw(15) << element << " ";
            }

            std::cout << "\n";
        }

        std::cout <<
"=======================================================================
=======================================\n";
    }
}
```

Результат работы программы:

```
================================================================================================================
Выбранные базисные переменные: x3 x4 x5

Полученная система:
            a1             a2             a3             a4             a5              b
     0.0344828       0.517241              1              0              0        2.55172
     -0.156187      -0.636917              0              1              0       -1.20487
      0.545639       -1.58012              0              0              1       -1.50507
================================================================================================================
Выбранные базисные переменные: x2 x4 x5

Полученная система:
            a1             a2             a3             a4             a5              b
     0.0666667              1        1.93333              0              0        4.93333
     -0.113725              0        1.23137              1              0        1.93725
       0.65098              0         3.0549              0              1         6.2902
================================================================================================================
Выбранные базисные переменные: x2 x3 x5

Полученная система:
            a1             a2             a3             a4             a5              b
      0.245223              1              0       -1.57006              0        1.89172
    -0.0923567              0              1       0.812102              0        1.57325
      0.933121              0              0       -2.48089              1        1.48408
================================================================================================================
```

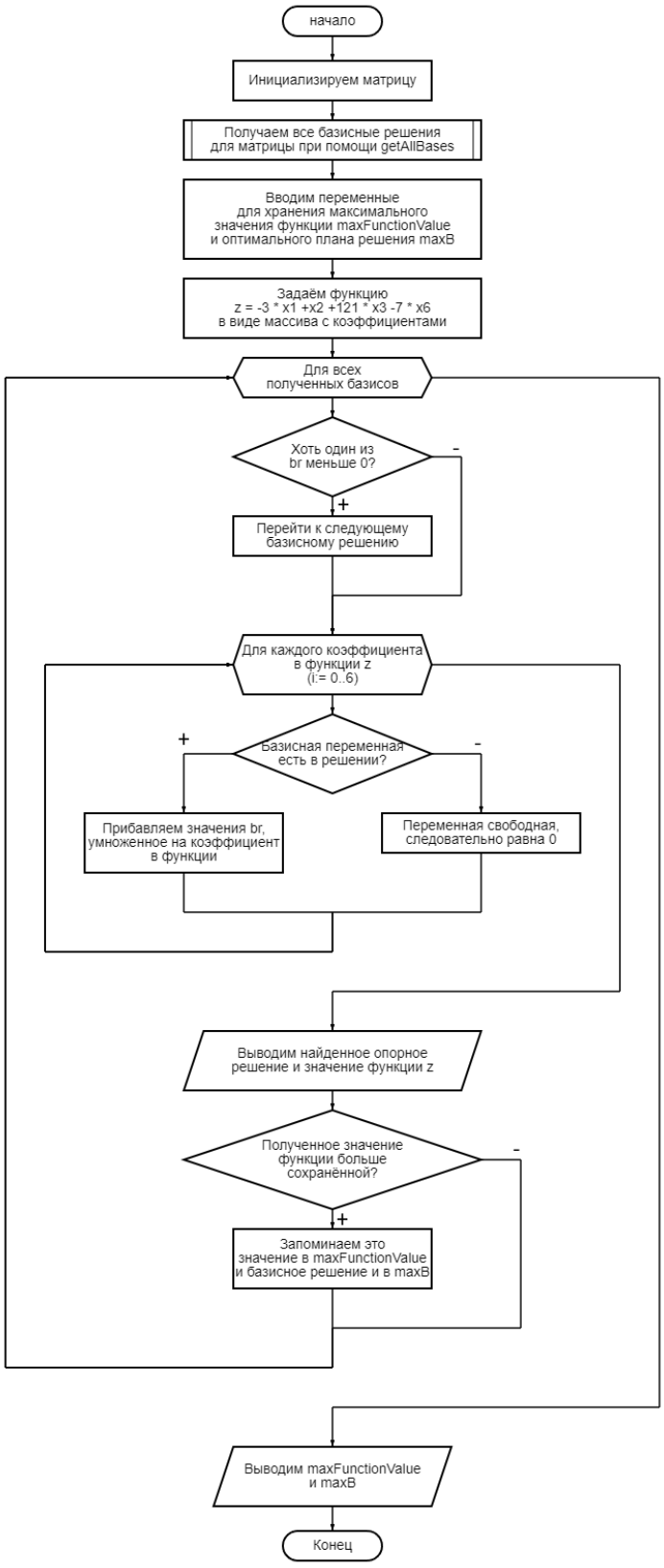
```
================================================================================================================
Выбранные базисные переменные: x2 x3 x4

Полученная система:
            a1             a2             a3             a4             a5              b
     -0.345315              1              0              0      -0.632863       0.952503
      0.213094              0              1              0       0.327343        2.05905
     -0.376123             -0             -0              1      -0.403081      -0.598203
================================================================================================================
Выбранные базисные переменные: x1 x4 x5

Полученная система:
            a1             a2             a3             a4             a5              b
             1             15             29              0              0             74
             0        1.70588        4.52941              1              0        10.3529
            -0       -9.76471       -15.8235             -0              1       -41.8824
================================================================================================================
Выбранные базисные переменные: x1 x3 x5

Полученная система:
            a1             a2             a3             a4             a5              b
             1        4.07792              0        -6.4026              0        7.71429
             0       0.376623              1       0.220779              0        2.28571
             0       -3.80519              0        3.49351              1       -5.71429
================================================================================================================
```

```
================================================================================================================
Выбранные базисные переменные: x1 x3 x4

Полученная система:
            a1             a2             a3             a4             a5              b
             1       -2.89591              0              0        1.83271       -2.75836
             0         0.6171              1              0      -0.063197        2.64684
             0       -1.08922              0              1       0.286245       -1.63569
================================================================================================================
Выбранные базисные переменные: x1 x2 x5

Полученная система:
            a1             a2             a3             a4             a5              b
             1              0       -10.8276       -8.7931              0       -17.0345
             0              1        2.65517       0.586207              0        6.06897
             0              0        10.1034        5.72414              1        17.3793
================================================================================================================
Выбранные базисные переменные: x1 x2 x4

Полученная система:
            a1             a2             a3             a4             a5              b
             1              0        4.69277              0        1.53614        9.66265
             0              1        1.62048              0      -0.10241        4.28916
             0              0        1.76506              1       0.174699        3.03614
================================================================================================================
```

```
================================================================================================================
Выбранные базисные переменные: x1 x2 x3

Полученная система:
            a1             a2             a3             a4             a5              b
             1              0              0       -2.6587        1.07167        1.59044
             0              1              0      -0.918089      -0.262799        1.50171
             0              0              1       0.566553      0.0989761        1.72014
================================================================================================================
```

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

Блок-схемы



Исходный код программы:

```
#include <iostream>
#include <iomanip>
#include <windows.h>
#include <limits>
#include <algorithm>
#include <array>

#include "../../libs/alg/labs/lab1/task1.tpp"

int main() {
    SetConsoleOutputCP(CP_UTF8);
    // Инициализируем матрицу
    std::vector<std::array<double, 6>> matrix = {
            { 2, -1,  6,  -1,  3,  12},
            { 3,  5,  1, -12,  2,  14},
            {-3,  6,  8,   7, -4,  18}};

    // Получаем все базисные решения для матрицы при помощи getAllBasises
    auto res = getAllBasises(matrix);

    // Вводим переменные для хранения максимального значения функции и опти-
мального плана решения
    double maxFunctionValue = std::numeric_limits<double>::min();
    std::vector<double> maxB;
    // Задаём функцию z = -10 * x1 + 2 * x2 + 75 * x3 - 18 * x6
    std::vector<double> function = {-10, 2, 75, 0, 0, -18};
    std::cout <<
"=============================================================================
=====================================\n";

    // Для всех полученных базисов
    for (auto &basis : res) {
        bool isAllBsMoreOrEqualToZero = true;
        for (int i = 0; i < basis.matrix.size() && isAllBsMoreOrEqualToZero;
i++) {
            if (basis.matrix[i].back() < EPS)
                isAllBsMoreOrEqualToZero = false;
        }

        // Хоть один из br меньше 0?
        if (!isAllBsMoreOrEqualToZero) {
            // Перейти к следующему базисному решению
            continue;
        }

        double z = 0;
        std::vector<double> B;
        // Для каждого коэффициента в функции z
        for (int i = 0; i < function.size(); i++) {
            // Базисная переменная есть в решении?
            if (std::find(basis.indices.begin(), basis.indices.end(), i) !=
basis.indices.end()) {
                for (int j = 0; j < basis.matrix.size(); j++) {
                    if (std::abs(basis.matrix[j][i] - 1.0) < EPS) {
                        // Прибавляем значение br, умноженное на коэффициент
в функции
                        z += function[i] * basis.matrix[j].back();
                        B.push_back(basis.matrix[j].back());
                        break;
                    }
```

```
                }
            } else {
                // Переменная свободная, следовательно равна 0
                B.push_back(0);
            }
        }

        // Выводим найденное опорное решение и значение функции z
        std::cout << "Обнаружено опорное решение: {";
        for (int i = 0; i < B.size(); i++) {
            std::cout << B[i] << "; ";
        }
        std::cout << "\b\b}\n\nЗначение функции z(B): " << z << "\n";

        std::cout <<
"=====================================================================================
====================================" << std::endl;

        // Полученное значение функции больше сохранённой?
        if (z > maxFunctionValue) {
            // Запоминаем его и базисное решение
            maxFunctionValue = z;
            maxB = B;
        }
    }

    // Выводим значение z и базисное решение
    std::cout << "\nZmax: " << maxFunctionValue << "\n\nОптимальный план: {";
    for (int i = 0; i < maxB.size(); i++) {
        std::cout << maxB[i] << "; ";
    }
    std::cout << "\b\b}" << std::endl;
}
```

Результат работы программы.

```
==========================================================================
Обнаружено опорное решение: {0; 4.93333; 0; 1.93725; 6.2902; 0}

Значение функции z(B): 9.86667
==========================================================================
Обнаружено опорное решение: {0; 1.89172; 1.57325; 0; 1.48408; 0}

Значение функции z(B): 121.777
==========================================================================
Обнаружено опорное решение: {9.66265; 4.28916; 0; 3.03614; 0; 0}

Значение функции z(B): -88.0482
==========================================================================
Обнаружено опорное решение: {1.59044; 1.50171; 1.72014; 0; 0; 0}

Значение функции z(B): 116.109
==========================================================================

Zmax: 121.777

Оптимальный план: {0; 1.89172; 1.57325; 0; 1.48408; 0}
```

3. Решить одну из следующих ниже задач вручную (подготовить тестовые данные).

Исходная система уравнений:

$$2x_1 - x_2 + 6x_3 - x_4 + 3x_5 = 12$$
$$3x_1 + 5x_2 + x_3 - 12x_4 + 2x_5 = 14$$
$$-3x_1 + 6x_2 + 8x_3 + 7x_4 - 4x_5 = 18$$

Построение расширенной матрицы:

2 -1 6 -1 3 | 12

3 5 1 -12 2 | 14

-3 6 8 7 -4 | 18

Вычтем из второй строки первую, умноженную на 1.5.

Вычтем из третьей строки первую, умноженную на -1.5.

2 -1 6 -1 3 | 12

0 7.5 -8 -13.5 -2.5 | -6

0 12 17 1 5.5 -0.5 | 36

Вычтем из третьей строки вторую.

2 -1 6 -1 3 | 12

0 1 -1.066 -1.8 -0.333 | -0.8

0 0 2.482 3.092 -0.291 | 3.8

Приведем второй ведущий элемент к 1 путем деления строки на 1.

Вычтем из первой строки шесть вторых строк, умноженных на 1.

Вычтем из третьей строки две вторых строки, умноженных на 2.482.

2 0 0 10 5 | 16

0 1 -1.066 -1.8 -0.333 | -0.8

0 0 1 1.246 -0.117 | 1.532

Приведем третий ведущий элемент к 0 путем вычитания второй строки, умноженной на 1.246.

Вычтем из первой строки 1.246 третьей строки.

2 0 0 8.708 5.291 | 14.468

0 1 0 -3.054 -0.463 | -1.422

0 0 1 1.246 -0.117 | 1.52

Теперь можем выразить базисные переменные через свободные переменные:

$$x_1 = 14.468 - 8.708x_4 - 5.291x_5$$

$$x_2 = -1.422 + 3.054x_4 + 0.463x_5$$

$$x_3 = 1.532 - 1.246x_4 + 0.117x_5$$

$$x_4, x_5 - \text{свободные переменные}$$

Вывод: в ходе лабораторной работы разработали и отладили программу, находящую базисные решения системы уравнений методом Гаусса-Жордана.