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 №2

по дисциплине: Исследование операций

тема: «Симплекс-метод в чистом виде»

Выполнил: ст. группы ПВ-233

Ситников Алексей Павлович

Проверил:

Вирченко Юрий Петрович

Белгород 2025 г.

Цель работы: изучение симплекс-метода для решения задачи линейного программирования с использованием симплекс-таблиц, получение навыков кодирования изученного алгоритма, отладки и тестирования соответствующих программ.

Вариант 13

Задания для подготовки к работе

1. Выяснить: какой вид должна иметь задача ЛП, чтобы можно было применять симплекс-метод в чистом виде, а также как составляется первая симплекс-таблица?

2. Изучить алгоритм перехода от одной симплекс-таблицы к другой при решении задачи симплекс-методом.

3. Запрограммировать и отладить изученный алгоритм. В рамках подготовки тестовых данных решить вручную следующую задачу:

13.

$$z = 5x_2 + 7x_4 + 3x_6 \to \max;$$

$$\begin{cases} x_1 - 3x_2 - 4x_4 - 2x_6 = 10, \\ 7x_2 + 5x_4 + x_5 + 4x_6 = 26, \\ 3x_2 + x_3 - 5x_4 - 4x_6 = 20, \end{cases}$$

$$x_i \geq 0 \left(i = \overline{1,6}\right).$$

Аналитическое решение:

Перезапишем целевую функцию:

$z - 5x_2 - 7x_4 - 3x_6 = 0$

Рисуем симплекс-таблицу:

Базисные переменные	Свободные члены	X_1	X_2	X_3	↓ X_4	X_5	X_6
X_1	10	1	-3	0	-4	0	-2
← X_5	26	0	7	0	5	1	4
X_3	20	0	3	1	-5	0	-4
z	0	0	-5	0	-7	0	-3

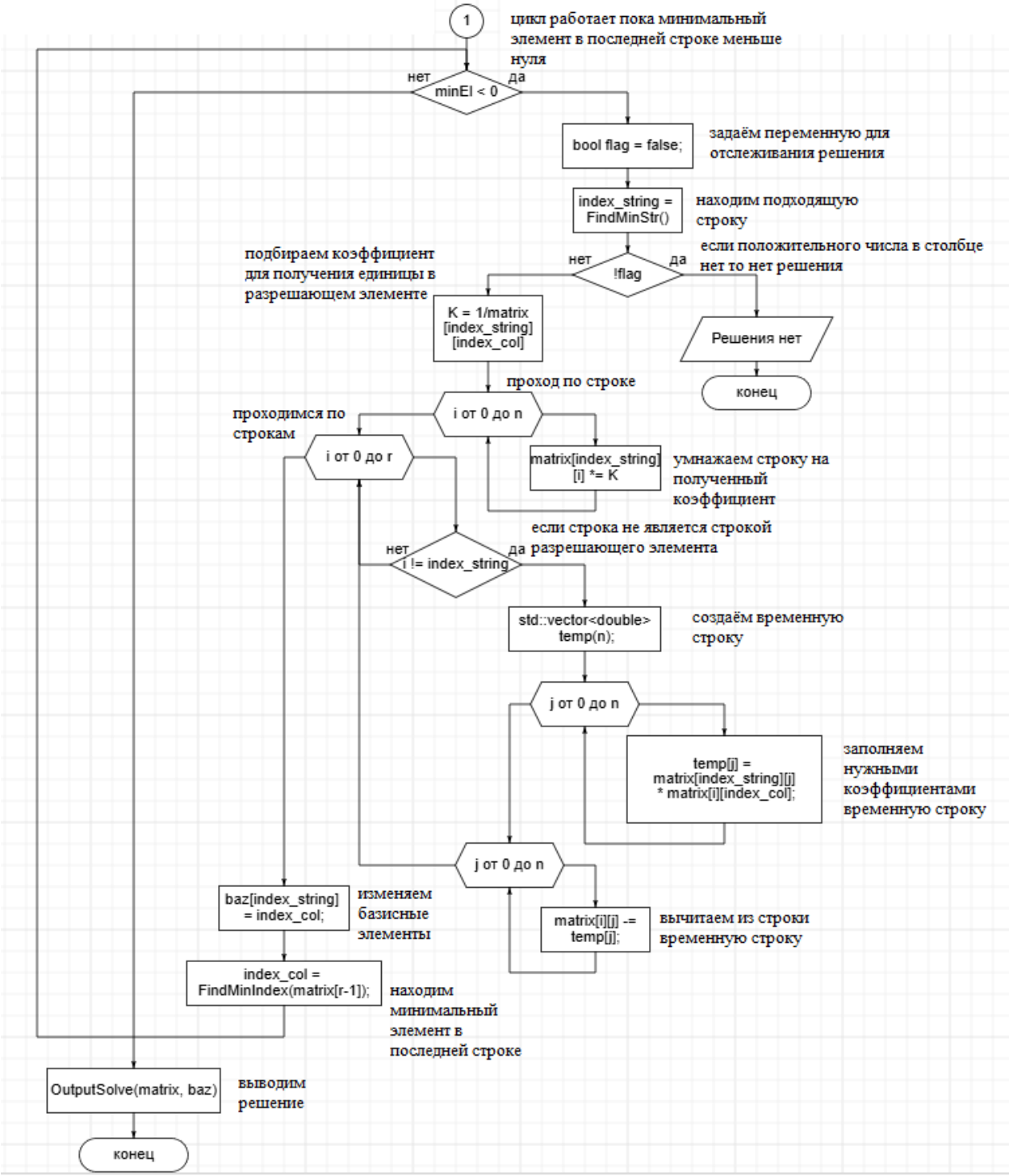
Переходим к новой симплекс-таблице

Базисные переменные	Свободные члены	X_1	X_2	X_3	X_4	X_5	X_6
X_1	$\frac{154}{5}$	1	$\frac{13}{5}$	0	0	$\frac{4}{5}$	$\frac{6}{5}$
X_4	$\frac{26}{5}$	0	$\frac{7}{5}$	0	1	$\frac{1}{5}$	$\frac{4}{5}$
X_3	46	0	10	1	0	1	0
z	$\frac{182}{5}$	0	$\frac{24}{5}$	0	0	$\frac{7}{5}$	$\frac{13}{5}$

$z_{max} = \frac{182}{5}$

Координаты точки максимума: $x_1 = \frac{154}{5}$, $x_2 = 0$, $x_3 = 46$, $x_4 = \frac{26}{5}$, $x_5 = 0$, $x_6 = 0$

Блок-схема модуля main



Код программы:

```
#include <iostream>
#include <vector>
#include <cmath>
#include <windows.h>

#define min_value 2.2e-10

//Находим минимальный коэффициент в строке целевой функции
int FindMinIndex(std::vector<double> v){
```

```
    double min = INT_MAX;//переменная, которая будет хранить минимальное
число
    int index;//переменная, которая будет хранить индекс минимального числа
    for(int i = 1; i < v.size(); i++){//пробегаемся по массиву
        if(min > v[i]){
            min = v[i];
            index = i;
        }
    }
    return index;//передаём индекс
}

//функция для ввода данных
std::vector<std::vector<double>> InputData(int n, int r){
    std::vector<std::vector<double>> matrix(r, std::vector<double>(n));
    for (int i = 0; i < r - 1; i++) {//пробегаемся по строкам
        std::cout << "Введите коэффициенты для уравнения " << i + 1 <<
std::endl;
        for (int j = 1; j < n; j++) {//пробегаемся по столбцам
            std::cin >> matrix[i][j];
        }
        std::cout << "Введите свободный член для уравнения " << i + 1 <<
std::endl;
        std::cin >> matrix[i][0];
    }
    std::cout << "Введите коэффициенты для целевой функции\n";
    for (int i = 1; i < n; i++) {
        std::cin >> matrix[r - 1][i];
        matrix[r - 1][i] *= -1;
    }
    return matrix;
}

//находим базисные элементы
void CreatBazis(int *baz, std::vector<std::vector<double>> matrix){
    int count = 0;//счётчик базисных элементов для оптимизации программы
    for(int i = 1; i < matrix[0].size(); i++){//пробегаемся по столбцам
        double sum = 0;//для базисного элемента сумма будет 1
        int index;// переменная для индекса строки
        for(int j = 0; j < matrix.size()-1; j++){//пробегаемся по строкам
            if(fabs(matrix[j][i]) > min_value){//проверка чтоб не было после
вычитания чисел с плавающей точкой невероятно маленьких иначе они считаются
за 0
                sum+=matrix[j][i];//суммируем столбец
                if(matrix[j][i] == 1){
                    index = j;//запоминаем индекс строки базисного элемента
                }
            }
        }
        if(sum == 1){
            baz[index] = i;//базисный элемент найден
            count++;
            if(count == matrix.size()-1){//если все базисные элементы
найдены, то конец функции
                return;
            }
        }
    }
}
//находим нужную строку
int FindMinStr(int index_col, std::vector<std::vector<double>> matrix, bool
*flag){
    int index_string;
    double min_val = INT_MAX;
```

```
    for(int i = 0; i < matrix.size()-1; i++){
        if(matrix[i][index_col] > 0){//число должно быть > 0
            *flag = true;//хотя бы одно число найдено, вероятно решение есть
            if(matrix[0][index_col]/matrix[i][index_col] < min_val){//находим
минимальное значение
                min_val = matrix[0][index_col]/matrix[i][index_col];
                index_string = i;
            }
        }
    }
    return index_string;//индекс найденной строки
}
//вывод решения
void OutputSolve(std::vector<std::vector<double>> matrix, const int *baz){
    std::cout << "Максимум функции = " << matrix[matrix.size()-1][0] <<
std::endl;
    std::cout << "Координаты точки максимума:" << std::endl << '<';
    for(int i = 1; i < matrix[0].size() ; i++){
        int f = -1;
        for(int j = 0; j < matrix.size()-1; j++){
            if(baz[j] == i){
                f = j;
            }
        }
        if(f!=-1){
            std::cout << "x" << i << " = " << matrix[f][0];
        }
        else{
            std::cout << "x" << i << " = " << 0;
        }
        if(i != matrix[0].size()-1){
            std::cout << "; ";
        }
    }
    std::cout << ">\n";
}
int main() {
    SetConsoleOutputCP(CP_UTF8);
    int n, r;
    std::cout << "Введите через пробел количество неизвестных и количество
уравнений\n";
    std::cin >> n >> r;
    n++;
    r++;
    std::vector<std::vector<double>> matrix = InputData(n, r);//ввод данных
    int baz[r-1];
    CreatBazis(baz, matrix);//нахождение базисных элементов
    int index_col = FindMinIndex(matrix[r-1]);//нахождение минимального
элемента в последней строке
    while (matrix[r-1][index_col] < 0){//пока в последней строке минимальное
число меньше 0
        bool flag = false;// для проверки есть ли в столбце положительные
числа
        int index_string = FindMinStr(index_col, matrix, &flag);//находим
подходящую строку, (index_string, index_col) - разрешающий элемент
        if(!flag){//если в столбце все числа отрицательные
            std::cout << "целевая функция неограниченна на области допустимых
значений переменных, решения нет" << std::endl;
            return 1;
        }
        double K = 1/matrix[index_string][index_col];//коэффициент для
получения единицы в разрешающем элементе
        for(int i = 0; i < n; i++){
            matrix[index_string][i] *= K;//умножаем всю строку разрешающего
```

```
элемента на этот коэффициент
        }
        for(int i = 0; i < r; i++){//проходимся п строкам
            if(i != index_string){//если строка не та в которой разрешающий
элемент
                std::vector<double> temp(n);//временная строка
                for(int j = 0; j < n; j++){
                    temp[j] = matrix[index_string][j] *
matrix[i][index_col];//заполняем её разрешающей умноженной на коэффициент
который нужно обнулить
                }
                for(int j = 0; j < n; j++){
                    matrix[i][j] -= temp[j];//вычитаем строку
                }
            }
        }
        baz[index_string] = index_col;//меняем базисные элементы
        index_col = FindMinIndex(matrix[r-1]);//снова находим минимальный
элемент в последней строке
    }
    OutputSolve(matrix, baz);//выводим решение
    return 0;
}
```

Вывод программы с данными из примера:

```
C:\Users\admin\CLionProjects\untitled8\cmake-build-debug\untitled8.exe
Введите через пробел количество неизвестных и количество уравнений
6 3
Введите коэффициенты для уравнения 1
1 -3 0 -4 0 -2
Введите свободный член для уравнения 1
10
Введите коэффициенты для уравнения 2
0 7 0 5 1 4
Введите свободный член для уравнения 2
26
Введите коэффициенты для уравнения 3
0 3 1 -5 0 -4
Введите свободный член для уравнения 3
20
Введите коэффициенты для целевой функции
0 5 0 7 0 3
Максимум функции = 36.4
Координаты точки максимума:
<x1 = 30.8; x2 = 0; x3 = 46; x4 = 5.2; x5 = 0; x6 = 0>

Process finished with exit code 0
```

Решение программы совпало с аналитическим решением.

Вывод: в результате проделанной работы я узнал, что симплекс-метод позволяет найти оптимальное решение для задач линейного программирования, в которых нужно найти максимум целевой функции. Также алгоритмы в работе с симплекс-таблицей оказались простыми в реализации, значит этот метод может легко оптимизировать решение задач.