**Федеральное агентство по образованию РФ**

**Государственное образовательное учреждение высшего профессионального образования**

**Белгородский Государственный Технологический Университет им В. Г. Шухова**

Кафедра программного обеспечения вычислительной техники и

автоматизированных систем.

**Лабораторная работа № 8**

**Задача дробно-линейного программирования**

Выполнила:

Студент группы КБ-213

Белякова Мария Евгеньевна

Принял:

Вирченко Ю.П.

**Содержание отчета**

1. Выполнение работы. Аналитическое решение

2.Программная реализация. Блок-схемы алгоритмов

3.Листинг программы

4.Результат работы программы

5.Вывод

**Цель работы:** освоить метод сведения задачи ДЛП к задаче линейного программирования с помощью введения новых переменных. Изучить алгоритм решения задачи ДЛП и реализовать программно этот алгоритм.

#### **Задания для подготовки к работе**

1. Изучить постановку задачи ДЛП, а также подходы к ее решению.
2. Ознакомиться с введением новых переменных, в которых задача ДЛП превращается в задачу ЛП.
3. Изучить метод и алгоритм решения задачи ДЛП, составить и отладить программу решения этой задачи, используя в качестве тестовых данных одну из нижеследующих задач, решенную вручную.

**Вариант 2.**

****

**Выполнение работы**

**Аналитическое решение**

Сведем данную задачу к задаче линейного программирования.  
Знаменатель обозначим:  
x1+x2+x3+x4 = 1/d  
Получим:  
L = 9\*x1d+3\*x2d+2\*x3d+x4d → max  
4\*x1d+x2d+x3d+3\*x4d ≤ 250d  
x1d+3\*x3d+x4d ≤ 80d  
x1d+3\*x2d+x3d ≤ 340d  
x1d+x2d+x3d+x4d = 1  
Введем новые переменные:  
y1 = x1d, y2 = x2d, y3 = x3d, y4 = x4d, y5 = d  
Получаем:  
**4\*y1+y2+y3+3\*y4-250y5 ≤ 0  
y1+3\*y3+y4-80y5 ≤ 0  
y1+3\*y2+y3-340y5 ≤ 0  
y1+y2+y3+y4 = 1**  
Целевая функция:  
**L = 9\*y1+3\*y2+2\*y3+y4 → max  
y1 ≥ 0, y2 ≥ 0, y3 ≥ 0, y4 ≥ 0, y5 ≥ 0**  
Далее решаем симплекс-методом.

Решим прямую задачу линейного программирования симплексным методом, с использованием симплексной таблицы.  
Определим максимальное значение целевой функции F(X) = 9x1+3x2+2x3+x4 при следующих условиях-ограничений.  
4x1+x2+x3+3x4-250x5≤0  
x1+3x3+x4-80x5≤0  
x1+3x2+x3-340x5≤0  
x1+x2+x3+x4=1  
Для построения первого опорного плана систему неравенств приведем к системе уравнений путем введения дополнительных переменных (**переход к канонической форме**).  
В 1-м неравенстве смысла (≤) вводим базисную переменную x6. В 2-м неравенстве смысла (≤) вводим базисную переменную x7. В 3-м неравенстве смысла (≤) вводим базисную переменную x8.  
4x1+x2+x3+3x4-250x5+x6 = 0  
x1+3x3+x4-80x5+x7 = 0  
x1+3x2+x3-340x5+x8 = 0  
x1+x2+x3+x4 = 1  
Введем **искусственные переменные x**: в 4-м равенстве вводим переменную x9;  
4x1+x2+x3+3x4-250x5+x6 = 0  
x1+3x3+x4-80x5+x7 = 0  
x1+3x2+x3-340x5+x8 = 0  
x1+x2+x3+x4+x9 = 1  
Для постановки задачи на максимум целевую функцию запишем так:  
F(X) = 9x1+3x2+2x3+x4 - Mx9 → max

За использование искусственных переменных, вводимых в целевую функцию, накладывается так называемый штраф величиной М, очень большое положительное число, которое обычно не задается.  
Полученный базис называется искусственным, а метод решения называется методом искусственного базиса.  
Причем искусственные переменные не имеют отношения к содержанию поставленной задачи, однако они позволяют построить стартовую точку, а процесс оптимизации вынуждает эти переменные принимать нулевые значения и обеспечить допустимость оптимального решения.  
Из уравнений выражаем искусственные переменные:  
x9 = 1-x1-x2-x3-x4  
которые подставим в целевую функцию:  
F(X) = 9x1 + 3x2 + 2x3 + x4 - M(1-x1-x2-x3-x4) → max  
или  
F(X) = (9+M)x1+(3+M)x2+(2+M)x3+(1+M)x4+(-M) → max  
Матрица коэффициентов A = a(ij) этой системы уравнений имеет вид:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 4 | 1 | 1 | 3 | -250 | 1 | 0 | 0 | 0 |
| 1 | 0 | 3 | 1 | -80 | 0 | 1 | 0 | 0 |
| 1 | 3 | 1 | 0 | -340 | 0 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |

**Базисные переменные** это переменные, которые входят только в одно уравнение системы ограничений и притом с единичным коэффициентом.  
**Экономический смысл дополнительных переменных**: дополнительные переменные задачи ЛП обозначают излишки сырья, времени, других ресурсов, остающихся в производстве данного оптимального плана.  
Решим систему уравнений относительно базисных переменных: x6, x7, x8, x9  
Полагая, что **свободные переменные** равны 0, получим первый опорный план:  
X0 = (0,0,0,0,0,0,0,0,1)  
**Базисное решение** называется допустимым, если оно неотрицательно.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Базис | B | x1 | x2 | x3 | x4 | x5 | x6 | x7 | x8 | x9 |
| x6 | 0 | 4 | 1 | 1 | 3 | -250 | 1 | 0 | 0 | 0 |
| x7 | 0 | 1 | 0 | 3 | 1 | -80 | 0 | 1 | 0 | 0 |
| x8 | 0 | 1 | 3 | 1 | 0 | -340 | 0 | 0 | 1 | 0 |
| x9 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| F(X0) | -M | -9-M | -3-M | -2-M | -1-M | 0 | 0 | 0 | 0 | 0 |

Переходим к основному алгоритму симплекс-метода.  
**Итерация №0**.  
**1. Проверка критерия оптимальности**.  
Текущий опорный план неоптимален, так как в индексной строке находятся отрицательные коэффициенты.  
**2. Определение новой базисной переменной**.  
В качестве ведущего выберем столбец, соответствующий переменной x1, так как это наибольший коэффициент по модулю.  
**3. Определение новой свободной переменной**.  
Вычислим значения Di по строкам как частное от деления: bi / ai1  
и из них выберем наименьшее:  
min (0 : 4 , 0 : 1 , 0 : 1 , 1 : 1 ) = 0  
Следовательно, 1-ая строка является ведущей.  
Разрешающий элемент равен (4) и находится на пересечении ведущего столбца и ведущей строки.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Базис | B | x1 | x2 | x3 | x4 | x5 | x6 | x7 | x8 | x9 | min |
| x6 | 0 | 4 | 1 | 1 | 3 | -250 | 1 | 0 | 0 | 0 | 0 |
| x7 | 0 | 1 | 0 | 3 | 1 | -80 | 0 | 1 | 0 | 0 | 0 |
| x8 | 0 | 1 | 3 | 1 | 0 | -340 | 0 | 0 | 1 | 0 | 0 |
| x9 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| F(X1) | -M | -9-M | -3-M | -2-M | -1-M | 0 | 0 | 0 | 0 | 0 |  |

Поскольку в последнем столбце присутствует несколько минимальных элементов 0, то номер строки выбираем по **правилу Креко**.  
Метод Креко заключается в следующем. Элементы строк, имеющие одинаковые наименьшие значения min=0, делятся на предполагаемые разрешающие элементы, а результаты заносятся в дополнительные строки. За ведущую строку выбирается та, в которой раньше встретится наименьшее частное при чтении таблицы слева направо по столбцам.  
**4. Пересчет симплекс-таблицы**.  
Формируем следующую часть симплексной таблицы. Вместо переменной x6 в план 1 войдет переменная x1.  
Строка, соответствующая переменной x1 в плане 1, получена в результате деления всех элементов строки x6 плана 0 на разрешающий элемент РЭ=4. На месте разрешающего элемента получаем 1. В остальных клетках столбца x1 записываем нули.  
Таким образом, в новом плане 1 заполнены строка x1 и столбец x1. Все остальные элементы нового плана 1, включая элементы индексной строки, определяются по правилу прямоугольника.  
Для этого выбираем из старого плана четыре числа, которые расположены в вершинах прямоугольника и всегда включают разрешающий элемент РЭ.  
НЭ = СЭ - (А\*В)/РЭ  
СТЭ - элемент старого плана, РЭ - разрешающий элемент (4), А и В - элементы старого плана, образующие прямоугольник с элементами СТЭ и РЭ.

Представим расчет каждого элемента в виде таблицы:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| B | x1 | x2 | x3 | x4 | x5 | x6 | x7 | x8 | x9 |
| 0 : 4 | 4 : 4 | 1 : 4 | 1 : 4 | 3 : 4 | -250 : 4 | 1 : 4 | 0 : 4 | 0 : 4 | 0 : 4 |
| 0-(0\*1):4 | 1-(4\*1):4 | 0-(1\*1):4 | 3-(1\*1):4 | 1-(3\*1):4 | -80-(-250\*1):4 | 0-(1\*1):4 | 1-(0\*1):4 | 0-(0\*1):4 | 0-(0\*1):4 |
| 0-(0\*1):4 | 1-(4\*1):4 | 3-(1\*1):4 | 1-(1\*1):4 | 0-(3\*1):4 | -340-(-250\*1):4 | 0-(1\*1):4 | 0-(0\*1):4 | 1-(0\*1):4 | 0-(0\*1):4 |
| 1-(0\*1):4 | 1-(4\*1):4 | 1-(1\*1):4 | 1-(1\*1):4 | 1-(3\*1):4 | 0-(-250\*1):4 | 0-(1\*1):4 | 0-(0\*1):4 | 0-(0\*1):4 | 1-(0\*1):4 |
| (0)-(0\*(-9-M)):4 | (-9-M)-(4\*(-9-M)):4 | (-3-M)-(1\*(-9-M)):4 | (-2-M)-(1\*(-9-M)):4 | (-1-M)-(3\*(-9-M)):4 | (0)-(-250\*(-9-M)):4 | (0)-(1\*(-9-M)):4 | (0)-(0\*(-9-M)):4 | (0)-(0\*(-9-M)):4 | (0)-(0\*(-9-M)):4 |

Получаем новую симплекс-таблицу:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Базис | B | x1 | x2 | x3 | x4 | x5 | x6 | x7 | x8 | x9 |
| x1 | 0 | 1 | 0.25 | 0.25 | 0.75 | -62.5 | 0.25 | 0 | 0 | 0 |
| x7 | 0 | 0 | -0.25 | 2.75 | 0.25 | -17.5 | -0.25 | 1 | 0 | 0 |
| x8 | 0 | 0 | 2.75 | 0.75 | -0.75 | -277.5 | -0.25 | 0 | 1 | 0 |
| x9 | 1 | 0 | 0.75 | 0.75 | 0.25 | 62.5 | -0.25 | 0 | 0 | 1 |
| F(X1) | -M | 0 | -0.75-0.75M | 0.25-0.75M | 5.75-M | -562.5-62.5M | 2.25+M | 0 | 0 | 0 |

**Итерация №1**.  
**1. Проверка критерия оптимальности**.  
Текущий опорный план неоптимален, так как в индексной строке находятся отрицательные коэффициенты.  
**2. Определение новой базисной переменной**.  
В качестве ведущего выберем столбец, соответствующий переменной x5, так как это наибольший коэффициент по модулю.  
**3. Определение новой свободной переменной**.  
Вычислим значения Di по строкам как частное от деления: bi / ai5  
и из них выберем наименьшее:  
min (- , - , - , 1 : 62.5 ) = 0.016  
Следовательно, 4-ая строка является ведущей.  
Разрешающий элемент равен (62.5) и находится на пересечении ведущего столбца и ведущей строки.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Базис | B | x1 | x2 | x3 | x4 | x5 | x6 | x7 | x8 | x9 | min |
| x1 | 0 | 1 | 0.25 | 0.25 | 0.75 | -62.5 | 0.25 | 0 | 0 | 0 | - |
| x7 | 0 | 0 | -0.25 | 2.75 | 0.25 | -17.5 | -0.25 | 1 | 0 | 0 | - |
| x8 | 0 | 0 | 2.75 | 0.75 | -0.75 | -277.5 | -0.25 | 0 | 1 | 0 | - |
| x9 | 1 | 0 | 0.75 | 0.75 | 0.25 | 62.5 | -0.25 | 0 | 0 | 1 | 0.016 |
| F(X2) | -M | 0 | -0.75-0.75M | 0.25-0.75M | 5.75-M | -562.5-62.5M | 2.25+M | 0 | 0 | 0 |  |

**4. Пересчет симплекс-таблицы**.  
Формируем следующую часть симплексной таблицы. Вместо переменной x9 в план 2 войдет переменная x5.  
Строка, соответствующая переменной x5 в плане 2, получена в результате деления всех элементов строки x9 плана 1 на разрешающий элемент РЭ=62.5. На месте разрешающего элемента получаем 1. В остальных клетках столбца x5 записываем нули.  
Таким образом, в новом плане 2 заполнены строка x5 и столбец x5. Все остальные элементы нового плана 2, включая элементы индексной строки, определяются по правилу прямоугольника.

Представим расчет каждого элемента в виде таблицы:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| B | x1 | x2 | x3 | x4 | x5 | x6 | x7 | x8 | x9 |
| 0-(1\*-62.5):62.5 | 1-(0\*-62.5):62.5 | 0.25-(0.75\*-62.5):62.5 | 0.25-(0.75\*-62.5):62.5 | 0.75-(0.25\*-62.5):62.5 | -62.5-(62.5\*-62.5):62.5 | 0.25-(-0.25\*-62.5):62.5 | 0-(0\*-62.5):62.5 | 0-(0\*-62.5):62.5 | 0-(1\*-62.5):62.5 |
| 0-(1\*-17.5):62.5 | 0-(0\*-17.5):62.5 | -0.25-(0.75\*-17.5):62.5 | 2.75-(0.75\*-17.5):62.5 | 0.25-(0.25\*-17.5):62.5 | -17.5-(62.5\*-17.5):62.5 | -0.25-(-0.25\*-17.5):62.5 | 1-(0\*-17.5):62.5 | 0-(0\*-17.5):62.5 | 0-(1\*-17.5):62.5 |
| 0-(1\*-277.5):62.5 | 0-(0\*-277.5):62.5 | 2.75-(0.75\*-277.5):62.5 | 0.75-(0.75\*-277.5):62.5 | -0.75-(0.25\*-277.5):62.5 | -277.5-(62.5\*-277.5):62.5 | -0.25-(-0.25\*-277.5):62.5 | 0-(0\*-277.5):62.5 | 1-(0\*-277.5):62.5 | 0-(1\*-277.5):62.5 |
| 1 : 62.5 | 0 : 62.5 | 0.75 : 62.5 | 0.75 : 62.5 | 0.25 : 62.5 | 62.5 : 62.5 | -0.25 : 62.5 | 0 : 62.5 | 0 : 62.5 | 1 : 62.5 |
| (0)-(1\*(-562.5-62.5M)):62.5 | (0)-(0\*(-562.5-62.5M)):62.5 | (-0.75-0.75M)-(0.75\*(-562.5-62.5M)):62.5 | (0.25-0.75M)-(0.75\*(-562.5-62.5M)):62.5 | (5.75-M)-(0.25\*(-562.5-62.5M)):62.5 | (-562.5-62.5M)-(62.5\*(-562.5-62.5M)):62.5 | (2.25+M)-(-0.25\*(-562.5-62.5M)):62.5 | (0)-(0\*(-562.5-62.5M)):62.5 | (0)-(0\*(-562.5-62.5M)):62.5 | (0)-(1\*(-562.5-62.5M)):62.5 |

Получаем новую симплекс-таблицу:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Базис | B | x1 | x2 | x3 | x4 | x5 | x6 | x7 | x8 | x9 |
| x1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| x7 | 0.28 | 0 | -0.04 | 2.96 | 0.32 | 0 | -0.32 | 1 | 0 | 0.28 |
| x8 | 4.44 | 0 | 6.08 | 4.08 | 0.36 | 0 | -1.36 | 0 | 1 | 4.44 |
| x5 | 0.016 | 0 | 0.012 | 0.012 | 0.004 | 1 | -0.004 | 0 | 0 | 0.016 |
| F(X2) | 9 | 0 | 6 | 7 | 8 | 0 | 0 | 0 | 0 | 9+M |

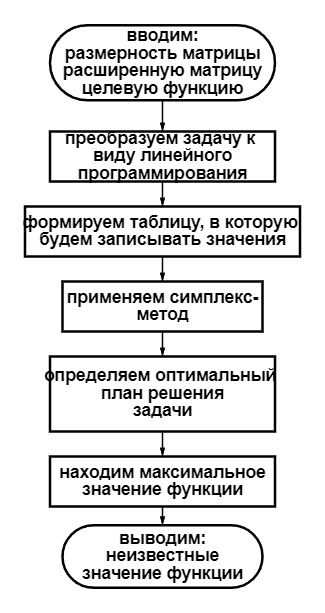
**1. Проверка критерия оптимальности**.  
Среди значений индексной строки нет отрицательных. Поэтому эта таблица определяет оптимальный план задачи.  
Окончательный вариант симплекс-таблицы:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Базис | B | x1 | x2 | x3 | x4 | x5 | x6 | x7 | x8 | x9 |
| x1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| x7 | 0.28 | 0 | -0.04 | 2.96 | 0.32 | 0 | -0.32 | 1 | 0 | 0.28 |
| x8 | 4.44 | 0 | 6.08 | 4.08 | 0.36 | 0 | -1.36 | 0 | 1 | 4.44 |
| x5 | 0.016 | 0 | 0.012 | 0.012 | 0.004 | 1 | -0.004 | 0 | 0 | 0.016 |
| F(X3) | 9 | 0 | 6 | 7 | 8 | 0 | 0 | 0 | 0 | 9+M |

Так как в оптимальном решении отсутствуют искусственные переменные (они равны нулю), то данное решение является допустимым.  
Оптимальный план можно записать так:  
**x1 = 1, x2 = 0, x3 = 0, x4 = 0, x5 = 0.016 F(X) = 9\*1 + 3\*0 + 2\*0 + 1\*0 = 9**

**Программная реализация**

**Блок-схемы алгоритмов**



**Листинг программы**

#include <locale>

#include <iostream>

#include <vector>

#include <cmath>

#define N 5

using namespace std;

struct NumInd {

float a;

int i;

};

void ReadMatr(vector<vector<float>>& matr) {

for (int i = 0; i < matr.size(); i++)

for (int j = 0; j < matr[i].size(); j++)

cin >> matr[i][j];

}

void WriteMatr(vector<vector<float>> matr) {

for (int i = 0; i < matr.size(); i++) {

for (int j = 0; j < matr[i].size(); j++)

cout << matr[i][j] << ' ';

cout << '\n';

}

cout << endl;

}

void BasicVariables(vector<vector<float>> matr, vector<int>& number) {

number.resize(matr.size());

for (int i = 0; i < matr.size(); i++) {

for (int j = 0; j < matr[i].size() - 1; j++) {

if (matr[i][j] == 1) {

int k;

for (k = 0; k < matr.size(); k++)

if (k != i && matr[k][j] != 0)

break;

if (k == matr.size())

number[i] = j + 1;

}

}

}

}

int NegativeNumbers(vector<float> vec) {

int count = 0;

for (int i = 1; i < vec.size(); i++)

if (vec[i] < 0)

count++;

return count;

}

void Sort(vector<NumInd>& numberNegative) {

for (int i = 0; i < numberNegative.size(); i++)

for (int j = numberNegative.size() - 1; j >= i + 1; j--)

if (numberNegative[j].a < numberNegative[j - 1].a) {

NumInd tmp = numberNegative[j];

numberNegative[j] = numberNegative[j - 1];

numberNegative[j - 1] = tmp;

}

}

void IndexMinNumber(vector<float> vec, vector<NumInd>& numberNegative) {

for (int i = 1; i < vec.size(); i++)

if (vec[i] < 0)

numberNegative.push\_back({ vec[i], i });

Sort(numberNegative);

}

int SimplexMethod(vector<vector<float>>& tabl, vector<int>& basicVar) {

vector<NumInd> numberNegative;

vector<NumInd> numPositive;

while (NegativeNumbers(tabl[tabl.size() - 1])) {

IndexMinNumber(tabl[tabl.size() - 1], numberNegative);

int i;

for (i = 0; i < numberNegative.size(); i++) {

for (int j = 0; j < tabl.size() - 1; j++)

if (tabl[j][numberNegative[i].i] > 0)

numPositive.push\_back({ tabl[j][numberNegative[i].i], j });

if (numPositive.size() != 0)

break;

}

if (numPositive.size()) {

float min = tabl[numPositive[0].i][0] / numPositive[0].a;

int iMin = 0;

for (int j = 1; j < numPositive.size(); j++)

if (min > tabl[numPositive[j].i][0] / numPositive[j].a) {

min = tabl[numPositive[j].i][0] / numPositive[j].a;

iMin = j;

}

basicVar[numPositive[iMin].i] = numberNegative[i].i;

for (int j = 0; j < tabl[0].size(); j++)

tabl[numPositive[iMin].i][j] /= numPositive[iMin].a;

for (int j = 0; j < tabl.size(); j++)

if (numPositive[iMin].i != j) {

float p = tabl[j][numberNegative[i].i];

for (int k = 0; k < tabl[0].size(); k++) {

tabl[j][k] += (-tabl[numPositive[iMin].i][k] \* p);

if (abs(tabl[j][k]) <= 0.0001)

tabl[j][k] = 0;

}

}

WriteMatr(tabl);

numberNegative.clear();

numPositive.clear();

}

else {

cout << "Система имеет неограниченное решение";

return 1;

}

}

return 0;

}

int CheckArtificialBasis(vector<int> basicVar, int m) {

for (int i = 0; i < basicVar.size(); i++)

if (basicVar[i] - m >= 0)

return 1;

return 0;

}

bool CheckIsBasic(vector<int> basic, int a) {

for (int i = 0; i < basic.size(); i++)

if (basic[i] == a)

return true;

return false;

}

void ArtBaseMethod(vector<vector<float>> tabl, vector<int> basicVar, vector<int> function, int m) {

cout << "\nИсходная таблица:\n";

WriteMatr(tabl);

if (!SimplexMethod(tabl, basicVar)) {

if (!CheckArtificialBasis(basicVar, m)) {

cout << "Полученная таблица:\n";

WriteMatr(tabl);

vector<vector<float>> tablbas;

tablbas.resize(tabl.size());

for (int i = 0; i < tablbas.size(); i++)

tablbas[i].resize(m);

for (int i = 0; i < tablbas.size() - 1; i++)

for (int j = 0; j < tablbas[i].size(); j++)

tablbas[i][j] = tabl[i][j];

tablbas[tablbas.size() - 1][0] = function[function.size() - 2];

for (int j = 0; j < tabl.size() - 1; j++)

tablbas[tablbas.size() - 1][0] += tablbas[j][0] \* function[basicVar[j] - 1];

for (int i = 1; i < tablbas[0].size(); i++) {

if (!CheckIsBasic(basicVar, i)) {

tablbas[tablbas.size() - 1][i] = 0;

for (int j = 0; j < tabl.size() - 1; j++)

tablbas[tablbas.size() - 1][i] += tablbas[j][i] \* function[basicVar[j] - 1];

tablbas[tablbas.size() - 1][i] -= function[i - 1];

}

}

cout << "\n";

WriteMatr(tablbas);

vector<float> function;

function.resize(N + 2);

if (!SimplexMethod(tablbas, basicVar)) {

cout << "\n";

WriteMatr(tablbas);

cout << "\nЗначение функции: " << tablbas[tablbas.size() - 1][0] << endl;

for (int i = 0; i <= N; i++)

function[i] = 0;

for (int i = 0; i < basicVar.size(); i++)

function[basicVar[i]] = tablbas[i][0];

for (int i = 0; i <= N; i++)

function[i] /= function[N + 1];

for (int i = 1; i <= N; i++)

cout << "x" << i << " = " << function[i] << endl;

}

else

cout << "Система уравнений не ограничена";

}

}

}

int SearchMulti(vector<vector<float>> matr) {

int max = matr[0][matr[0].size() - 1];

for (int i = 1; i < matr.size(); i++)

if (max < matr[i][matr[0].size() - 1])

max = matr[i][matr[0].size() - 1];

int t = 10;

while (max > t)

t \*= 10;

t \*= 10;

return t;

}

void MMethod(vector<vector<float>> tabl, vector<int> basicVar) {

vector<float> function;

function.resize(N + 2);

cout << "\nИсходная таблица:\n";

WriteMatr(tabl);

if (!SimplexMethod(tabl, basicVar)) {

WriteMatr(tabl);

cout << "\nИскомое значение функции: " << tabl[tabl.size() - 1][0] << endl;

for (int i = 0; i <= N; i++)

function[i] = 0;

for (int i = 0; i < basicVar.size(); i++)

function[basicVar[i]] = tabl[i][0];

for (int i = 0; i <= N; i++)

function[i] /= function[N + 1];

for (int i = 1; i <= N; i++)

cout << "x" << i << " = " << function[i] << endl;

}

}

int main() {

setlocale(LC\_ALL, "RUS");

int n, m;

cout << "Введите размеры матрицы: ";

cin >> n >> m;

vector<vector<float>> matr;

matr.resize(n);

for (int i = 0; i < n; i++)

matr[i].resize(m);

cout << "Введите расширенную матрицу ограничений:\n";

ReadMatr(matr);

for (int i = 0; i < matr.size(); i++)

if (matr[i][matr[0].size() - 1] < 0)

for (int j = 0; j < matr[i].size(); j++)

matr[i][j] \*= -1;

vector<vector<float>> matr2;

matr2.resize(n + 1);

for (int i = 0; i <= n; i++)

matr2[i].resize(m + 1);

for (int i = 0; i < n; i++)

for (int j = 0; j < m; j++)

matr2[i][j] = matr[i][j];

for (int i = 0; i < n; i++)

matr2[i][m - 1] \*= -1;

for (int i = 0; i < n; i++)

matr2[i][m] = 0;

vector<int> function;

function.resize(m + 2);

for (int i = 0; i <= m; i++)

cin >> function[i];

for (int i = 0; i < m; i++)

cin >> matr2[n][i];

matr2[n][m] = 1;

cout << "Введите 1 если функция стремится к максимуму, 0 если к минимуму: ";

cin >> function[m + 1];

if (function[m + 1] == 0)

for (int i = 0; i < function.size() - 1; i++)

function[i] \*= -1;

WriteMatr(matr2);

m++;

n++;

vector<int> basicVar;

BasicVariables(matr2, basicVar);

int k = 0;

for (int i = 0; i < basicVar.size(); i++)

if (basicVar[i] == 0) {

basicVar[i] = m + k;

k++;

}

vector<vector<float>> tabl;

tabl.resize(n + 1);

for (int i = 0; i < tabl.size(); i++)

tabl[i].resize(m + k);

for (int i = 0; i < tabl.size() - 1; i++)

tabl[i][0] = matr2[i][m - 1];

for (int i = 0; i < matr2.size(); i++)

for (int j = 0; j < matr2[i].size() - 1; j++)

tabl[i][j + 1] = matr2[i][j];

for (int i = 0; i < basicVar.size(); i++)

if (basicVar[i] - m >= 0) {

tabl[i][basicVar[i]] = 1;

for (int j = 0; j < tabl.size(); j++)

if (j != i)

tabl[j][basicVar[i]] = 0;

}

tabl[tabl.size() - 1][0] = 0;

for (int i = 0; i < basicVar.size(); i++)

if (basicVar[i] - m >= 0)

tabl[tabl.size() - 1][0] += matr2[i][matr2[i].size() - 1];

for (int i = 1; i < matr2[0].size(); i++) {

tabl[tabl.size() - 1][i] = 0;

for (int j = 0; j < basicVar.size(); j++)

if (basicVar[j] - m >= 0)

tabl[tabl.size() - 1][i] += matr2[j][i - 1];

}

for (int i = 0; i < tabl[0].size(); i++)

tabl[tabl.size() - 1][i] \*= -1;

cout << "Метод искусственного базиса\n";

ArtBaseMethod(tabl, basicVar, function, m);

int M = SearchMulti(matr2);

tabl[tabl.size() - 1][0] = function[function.size() - 2];

float sum = 0;

for (int i = 0; i < basicVar.size(); i++)

if (basicVar[i] - m >= 0)

sum += tabl[i][0];

tabl[tabl.size() - 1][0] -= (M \* sum);

for (int j = 1; j < matr2[0].size(); j++) {

tabl[tabl.size() - 1][j] = function[j - 1];

sum = 0;

for (int i = 0; i < basicVar.size(); i++)

if (basicVar[i] - m >= 0)

sum += tabl[i][j];

tabl[tabl.size() - 1][j] += (M \* sum);

tabl[tabl.size() - 1][j] \*= -1;

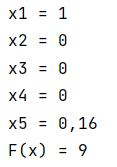
}

cout << " Метод больших штрафов\n";

MMethod(tabl, basicVar);

}

**Результат работы программы**



**Вывод**

В данной лабораторной работе нам было предложено изучить задачи ДЛП, а также освоить метод сведения задачи ДЛП к виду задачи ЛП, при помощи введения новых переменных, а также программно реализовать этот метод.

Для того, чтобы привести задачу ДЛП к виду ЛП, мы использовали дополнительную переменную, в которую записали знаменатель целевой функции, затем, используя симплекс-метод решили задачу.

Предложенный метод программно реализовали. Результат работы программы соответствует аналитическому решению.

Что касаемо программной реализации предложенного алгоритма, она получилась довольно объемной. Проанализировав представленные методы решения, мы пришли к выводу о том, что для данного метода удобнее использовать «ручное» решение при выполнении аналогичного задания.