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 №3

по дисциплине: Исследование операций тема:
«***Модификации симплекс метода. Методы искусственного базиса и больших штрафов***»

Выполнил: ст. группы ПВ-233
Мороз Роман Алексеевич

Проверил:
Вирченко Юрий Петрович

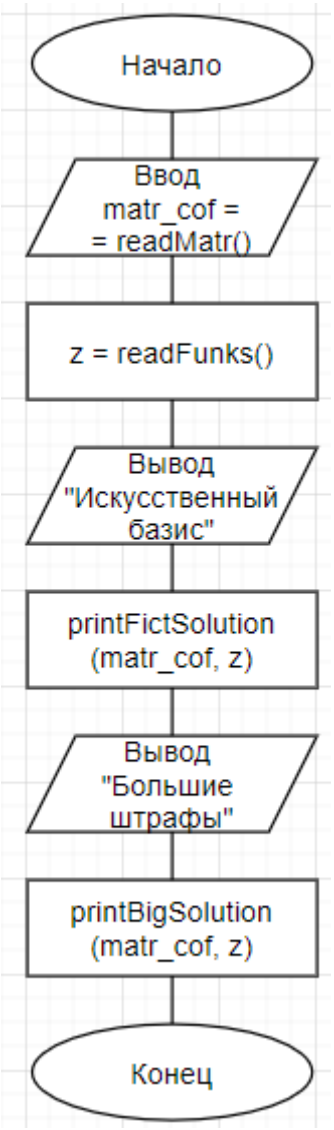
Белгород 2025 г.

Цель работы: изучение методов искусственного базиса и больших штрафов решения задач ЛП в канонической форме, не подготовленных к работе симплекс-методом в чистом виде.

Задания для подготовки к работе

1. Изучить метод и алгоритм искусственного базиса и составить программу решения задачи ЛП этим методом.
2. Изучить метод и алгоритм больших штрафов и составить программу решения задачи ЛП этим методом.
3. Запрограммировать изученные алгоритмы и отладить соответствующие программы. В рамках подготовки тестовых данных решить вручную одну из следующих ниже задач.

Вариант 9



Листинг программы:

```
from math import factorial, fabs
from itertools import *

# Функция для ввода матрицы
def readMatr():
    print("Введите количество строк: ")
    count_string = int(input())
    print("Введите матрицу: ")
    matr = [[float(x) for x in input().split()] for y in range(count_string)]
    return matr

# Функция для перевода ведущего столбца в единичный
def basic_collumn(matr_cof, number_string, number_x):
    # Сохраняем значение элемента в ведущей строке и ведущем столбце
    cof = matr_cof[number_string][number_x]
    # Делаем ведущий элемент равным 1 путем деления всей строки на этот элемент
    for i in range(len(matr_cof[number_string])):
        matr_cof[number_string][i] /= cof

    # Вычитаем ведущую строку из всех остальных строк так, чтобы элементы под
    # ведущим столбцом стали равными нулю
```

```
    for i in range(len(matr_cof)):
        if i != number_string:
            # Находим коэффициент, на который нужно умножить ведущую строку, чтобы
            # вычесть ее из текущей строки
            cof = matr_cof[i][number_x]
            # Вычитаем ведущую строку, умноженную на найденный коэффициент
            for j in range(len(matr_cof[i])):
                matr_cof[i][j] -= matr_cof[number_string][j] * cof
    # Возвращаем измененную матрицу коэффициентов
    return matr_cof
# Функция удаления строк с нулевыми коэффициентами и решения несовместных систем
def delZeroOrNesov(matr_cof):
    for i in range(len(matr_cof)):
        flag = True
        # Проверяем, все ли коэффициенты в текущей строке равны нулю
        for j in range(len(matr_cof[i])):
            if (matr_cof[i][j] != 0):
                flag = False
        # Если все коэффициенты в строке равны нулю, и свободный член также равен
        # нулю, удаляем строку
        if flag & (matr_cof[i][-1] == 0):
            matr_cof.pop(i)
        # Если все коэффициенты в строке равны нулю, но свободный член не равен
        # нулю, система несовместима
        elif flag:
            print("Ваша система несовместима")
            exit(6)
# Функция для получения списка базисных переменных
def getBasicList(matr_cof):
    basicList = []  # Список базисных переменных
    usage_string = []  #Список номеров строк,в которых базисные переменные равны 1
    # Проходим по столбцам матрицы коэффициентов
    for i in range(len(matr_cof[0])):
        count_one = 0  # Счетчик единиц в столбце
        # Счетчик других значений (не нулей и не единиц) в столбце
        count_another = 0
        one_index = 0  # Индекс строки, в которой нашлась единица
        # Проходим по строкам столбца
        for j in range(len(matr_cof)):
            # Если текущий элемент равен 1, увеличиваем счетчик единиц и
            # запоминаем индекс строки
            if matr_cof[j][i] == 1:
                count_one += 1
                one_index = j
            # Если текущий элемент не равен 0 и не равен 1, увеличиваем счетчик
            # других значений
            elif matr_cof[j][i] != 0:
                count_another += 1
        # Если в столбце ровно одна единица и других значений нет, добавляем
        # столбец в список базисных переменных
        if ((count_one == 1) & (count_another == 0)):
            basicList.append(i)
            usage_string.append(one_index)
    # Возвращаем список базисных переменных и список номеров строк
    return [basicList, usage_string]



# Функция для проверки равенства базисных переменных
def checkEqual(arr, basic, comb):
    # Проходим по всем парам элементов массива
    for i in range(len(arr) - 1):
```

```
        for j in range(i + 1, len(arr)):
            # Если элементы равны и соответствующие базисные переменные находятся
            # в заданной комбинации
            if (arr[i] == arr[j]) & (basic[i] in comb) & (basic[j] in comb):
                return True  # Возвращаем True, если найдены равные базисные
                             # переменные
    return False  # Возвращаем False, если равных базисных переменных не найдено




# Функция для получения базисной матрицы
def getBasicMatr(matr_cof, comb):
    # Создаем копию матрицы коэффициентов, чтобы не изменять исходную матрицу
    new_matr = [[el for el in row] for row in matr_cof]
    # Получаем списки базисных переменных и соответствующих им строк матрицы
    basic_list, basic_string = getBasicList(new_matr)
    usage_strings = []  # Список использованных строк

    # Проходим по всем базисным переменным из комбинации
    for el in comb:
        if el not in basic_list:  # Если переменная не является базисной
            # Находим строку, где можно сделать эту переменную базисной
            for i in range(len(new_matr)):
                if (new_matr[i][el] != 0) & (i not in usage_strings):
                    # Переводим соответствующий столбец в единичный, чтобы сделать
                    # переменную базисной
                    basic_collumn(new_matr, i, el)
                    # Добавляем использованную строку в список
                    usage_strings.append(i)
                    break
            # Удаляем строки с нулевыми коэффициентами и проверяем на
            # несовместность
            delZeroOrNesov(new_matr)
            # Обновляем списки базисных переменных
            basic_list, basic_string = getBasicList(new_matr)

            # Проверяем, не равны ли какие-либо базисные переменные, и возвращаем
            # пустую матрицу, если равны
            if checkEqual(basic_string, basic_list, comb):
                return []
        else:
            # Если переменная уже является базисной, добавляем соответствующую
            # строку в список использованных строк
            usage_strings.append(basic_string[basic_list.index(el)])

    return new_matr  # Возвращаем базисную матрицу


# Функция для чтения функции
def readFunck():
    print("Input Z: ")
    return [float(x) for x in input().split()]

# Функция для получения первой таблицы
def getFirstTable(matr_cof, z):
    # Получаем списки базисных переменных и соответствующих им строк матрицы
    basic, stringBasic = getBasicList(matr_cof)
    # Выделяем базисные переменные и соответствующие им строки из списков
    basic = [basic[i] for i in range(len(basic) - len(matr_cof), len(basic))]
```

```
    stringBasic = [stringBasic[i] for i in range(
        len(stringBasic) - len(matr_cof), len(stringBasic))]

    # Создаем пустую таблицу с нужными размерами и заполняем ее заголовками
    table = [[0 for y in range(len(matr_cof[0]) + 1)]
             for x in range(len(basic) + 2)]
    table[0][0] = "Базисные переменные"
    table[0][1] = "Свободные члены"
    for i in range(2, len(table[0])):
        table[0][i] = 'X' + str(i - 1)

    # Заполняем таблицу базисными переменными и соответствующими им свободными
    # членами
    for x in range(len(basic)):
        table[stringBasic[x] + 1][0] = 'X' + str(basic[x] + 1)
    for i in range(1, len(table) - 1):
        table[i][1] = matr_cof[i - 1][len(matr_cof[0]) - 1]
    # Заполняем остальные ячейки таблицы коэффициентами из матрицы коэффициентов
    for i in range(len(matr_cof)):
        for j in range(len(matr_cof[0]) - 1):
            table[i + 1][j + 2] = matr_cof[i][j]
    # Заполняем строку функции Z коэффициентами и свободным членом
    table[len(table) - 1][0] = 'Z'
    table[len(table) - 1][1] = z[0]
    for i in range(1, len(z)):
        table[len(table) - 1][i + 1] = -z[i]
    return table

# Функция для проверки решения
def checkSolution(table):
    for i in range(2, len(table[0])):
        if (table[len(table) - 1][i] < 0):
            return False
    return True

# Функция для получения опорного столбца
def getOpSt(table):
    max_index = 2  # Индекс начального опорного столбца
    # Проходим по всем столбцам таблицы, начиная с третьего
    for i in range(2, len(table[0])):
        # Если элемент последней строки в текущем столбце отрицательный
        if table[len(table) - 1][i] < 0:
            # Если текущий элемент меньше элемента с индексом max_index
            if table[len(table) - 1][i] < table[len(table) - 1][max_index]:
                max_index = i  # Обновляем индекс опорного столбца
    return max_index  # Возвращаем индекс опорного столбца




# Функция для выполнения следующего шага в методе искусственного базиса
def nextStep(table):
    indexMin = getOpSt(table)  # Получаем индекс опорного столбца
    indexChange = 0  # Инициализируем индекс строки для изменения
    min_val = float('inf')  # Инициализируем минимальное значение

    # Находим строку для изменения с минимальным отношением свободного члена к
    # элементу в опорном столбце
    for i in range(1, len(table) - 1):
        if table[i][indexMin] > 0:
            ratio = table[i][1] / table[i][indexMin]  # Вычисляем отношение
            if ratio < min_val:
```

```
                min_val = ratio
                indexChange = i
    # Проверяем, если indexChange все еще равно 0, значит функция не ограничена
    if indexChange == 0:
        print("Функция не ограничена")
        exit(5)
    delit = table[indexChange][indexMin]  # Получаем делитель
    # Делим все элементы строки, включая свободный член, на значение делителя
    for i in range(1, len(table[indexChange])):
        table[indexChange][i] /= delit

    newTable = [[y for y in x] for x in table]  # Создаем новую таблицу

    # Устанавливаем индекс опорного столбца в строке изменения
    newTable[indexChange][0] = 'X' + str(indexMin - 1)

    # Проходим по всем строкам кроме строки изменения
    for i in range(1, len(table)):
        if i != indexChange:
            # Получаем коэффициент для умножения строки изменения
            cof = -table[i][indexMin]
            # Добавляем к строке i строку изменения, умноженную на коэффициент cof
            for j in range(1, len(table[i])):
                newTable[i][j] = table[i][j] + table[indexChange][j] * cof

    return newTable  # Возвращаем обновленную таблицу
# Функция для вывода таблицы
def printTable(table):
    for str in table:
        for el in str:
            print(el, end=" ")
        print()


# Функция для вывода решения
def print_solution(table):
    print("Решение")
    solution = [0 for x in range(len(table[0]) - 2)]

    for i in range(1, len(table) - 1):
        print(int(table[i][0][1]))
        solution[int(table[i][0][1]) - 1] = table[i][1]
    print(solution)
    print("Значение в  точке максимума")
    print("Zmax =", table[len(table) - 1][1])

# Функция для вывода вспомогательной таблицы
def printFict(table, n):
    for i in range(len(table)-1):
        for j in range(len(table[0])):
            print(table[i][j], end=" ")
        print()

# Функция для получения опорного столбца во вспомогательной таблице
def getOpStlog(table):
    max_index = 2  # Индекс начального опорного столбца
    # Проходим по всем столбцам таблицы, начиная с третьего
    for i in range(2, len(table[0])):
        # Если элемент предпоследней строки в текущем столбце отрицательный
        if table[len(table) - 2][i] < 0:
            # Если текущий элемент меньше элемента с индексом max_index
            if table[len(table) - 2][i] < table[len(table) - 2][max_index]:
                max_index = i  # Обновляем индекс опорного столбца
```

```
    return max_index  # Возвращаем индекс опорного столбца

# Функция для выполнения следующего шага в вспомогательной таблице
def nextStepFict(table):
    indexMin = getOpStlog(table)  # Получаем индекс опорного столбца
    indexChange = 0  # Инициализируем индекс строки для изменения
    min_val = float('inf')  # Инициализируем минимальное значение

    # Находим строку для изменения с минимальным отношением свободного члена к
    # элементу в опорном столбце
    for i in range(1, len(table) - 2):
        if table[i][indexMin] > 0:
            ratio = table[i][1] / table[i][indexMin]  # Вычисляем отношение
            if ratio < min_val:
                min_val = ratio
                indexChange = i

    # Проверяем, если indexChange не является строкой Y, выбираем первую строку Y
    if table[indexChange][0][0] != 'Y':
        for i in range(1, len(table) - 2):
            if table[i][0][0] == 'Y':
                indexChange = i
                break

        # Если indexMin равен нулю или больше или равен длине таблицы - 2,
        # выбираем первый ненулевой столбец
        if (table[indexChange][indexMin] == 0) or (indexMin >= len(table[0]) -
                                                   len(table) + 2):
            for i in range(2, len(table[0]) - len(table) + 2):
                if abs(table[indexChange][i]) > 1e-5:
                    indexMin = i

    delit = table[indexChange][indexMin]  # Получаем делитель
    for i in range(1, len(table[indexChange])):
        # Делим все элементы строки на значение делителя
        table[indexChange][i] /= delit

    newTable = [[y for y in x] for x in table]  # Создаем новую таблицу
    # Устанавливаем индекс опорного столбца в строке изменения
    newTable[indexChange][0] = 'X' + str(indexMin - 1)

    # Проходим по всем строкам, кроме строки изменения
    for i in range(1, len(table)):
        if i != indexChange:
            # Получаем коэффициент для умножения строки изменения
            cof = -table[i][indexMin]
            # Добавляем к строке i строку изменения, умноженную на коэффициент cof
            for j in range(1, len(table[i])):
                newTable[i][j] = table[i][j] + table[indexChange][j] * cof

    return newTable  # Возвращаем обновленную таблицу

# Функция для добавления фиктивных переменных
def putFict(matr):
    # Создаем новую матрицу, увеличивая количество столбцов на количество строк
    newMatr = [[0.0 for j in range(len(matr[0]) + len(matr))]
               for i in range(len(matr))]

    # Заполняем новую матрицу
    for i in range(len(matr)):
        for j in range(len(matr[0]) - 1):
            # Если свободный член отрицательный, меняем знак элементов матрицы
            if matr[i][len(matr[0]) - 1] < 0:
```

```
                newMatr[i][j] = -matr[i][j]
            else:
                newMatr[i][j] = matr[i][j]

    # Добавляем фиктивные переменные в последние столбцы
    for i in range(len(matr)):
        newMatr[i][len(matr[0]) + i - 1] = 1.0

    # Заполняем последний столбец матрицы значениями свободных членов
    for i in range(len(matr)):
        if matr[i][len(matr[0]) - 1] < 0:
            newMatr[i][len(newMatr[0]) - 1] = -matr[i][len(matr[0]) - 1]
        else:
            newMatr[i][len(newMatr[0]) - 1] = matr[i][len(matr[0]) - 1]

    return newMatr  # Возвращаем новую матрицу с добавленными фиктивными
                    # переменными

# Функция для вычисления значений Z в вспомогательной таблице
def fictZ(matr):
    zf = [0.0 for i in range(len(matr[0]))]
    for j in range(len(matr)):
        zf[0] -= matr[j][len(matr[0]) - 1]
    for i in range(1, len(zf) - len(matr)):
        for j in range(len(matr)):
            zf[i] += matr[j][i - 1]
    return zf

# Функция для получения таблицы с фиктивным базисом
def getFictBasisTable(matr, z):
    # Добавляем фиктивные переменные к матрице
    newMatr = putFict(matr)
    # Вычисляем значения функции Z с учетом фиктивных переменных
    zf = fictZ(newMatr)
    # Получаем первоначальную таблицу с фиктивным базисом
    table = getFirstTable(newMatr, zf)
    # Добавляем строку для функции Z
    table.append([0 for i in range(len(table[0]))])
    table[len(table) - 1][0] = 'Z'
    table[len(table) - 1][1] = z[0]
    for i in range(1, len(z)):
        table[len(table) - 1][i + 1] = -z[i]
    # Преобразуем переменные Y в соответствии с таблицей
    for i in range(1, len(table) - 2):
        table[i][0] = 'Y' + table[i][0][1]
    # Выводим таблицу на экран перед выполнением искусственного базиса
    printFict(table, len(matr[0]) - 1)
    # Проводим шаги искусственного базиса
    for i in range(len(matr)):
        table = nextStepFict(table)
        printFict(table, len(matr[0]) - 1)
    # Удаляем лишнюю строку с искусственной функцией Z
    newTable = [[0 for i in range(len(table[0]) - len(matr))]
                for i in range(len(table))]
    for i in range(len(newTable)):
        for j in range(len(newTable[0])):
            newTable[i][j] = table[i][j]
    newTable.pop(len(newTable) - 2)
    newTable[len(newTable) - 1][0] = 'Z'
    return newTable

# Функция для вывода решения с фиктивными переменными
def printFictSolution(matr, z):
```

```
    table = getFictBasisTable(matr, z)
    printTable(table)
    while not checkSolution(table):
        table = nextStep(table)
        printTable(table)
    print_solution(table)

# Функция для вычисления значений Z с большими штрафами
def getMz(matr, z, M):
    # Инициализируем массив значений функции Z
    zf = [0.0 for i in range(len(matr[0]))]
    # Добавляем к Z исходные коэффициенты функции
    for i in range(len(z)):
        zf[i] += z[i]
    # Вычисляем значения с учетом больших штрафов (M)
    for j in range(len(matr)):
        # Вычитаем из Z значение M * свободный член
        zf[0] -= M * matr[j][len(matr[0]) - 1]
    # Вычисляем остальные значения функции Z с учетом больших штрафов (M)
    for i in range(1, len(zf) - len(matr)):
        for j in range(len(matr)):
            # Добавляем к Z значение M * коэффициент
            zf[i] += M * matr[j][i - 1]
    return zf  # Возвращаем массив значений функции Z

# Функция для получения таблицы с большими штрафами
def getBigTable(matr, z):
    M = 1000
    newMatr = putFict(matr)  # Добавляем фиктивные переменные
    zf = getMz(newMatr, z, M)  # Вычисляем значения Z с учетом больших штрафов
    table = getFirstTable(newMatr, zf)  # Получаем первоначальную таблицу
    return table  # Возвращаем таблицу

# Функция для вывода решения с большими штрафами
def printBigSolution(matr, z):
    table = getBigTable(matr, z)
    printTable(table)
    while not checkSolution(table):
        table = nextStep(table)
        printTable(table)
    print_solution(table)

# Основная функция
def main():
    matr_cof = readMatr()
    z = readFunck()
    print("Искусственный базис")
    printFictSolution(matr_cof, z)
    print("\nБольшие штрафы")
    printBigSolution(matr_cof, z)
if __name__ == "__main__":
    main()
```

Старт

Базисные переменные	x1	x2	x3	x4	x5	x6	x7
3	-1	3	2	1	1	0	22
1	4	2	5	-2	0	0	45
-2	5	6	3	-7	0	-1	62

Таблица с искусственными переменными

Базисные переменные	x1	x2	x3	x4	x5	x6	x7	u1	u2	Свободные члены
x6	3	-1	3	2	1	1	0	0	0	22
u1	1	4	2	5	-2	0	0	1	0	45
u2	-2	5	6	3	-7	0	-1	0	1	62

Промежуточная таблица

Базисные переменные	x1	x2	x3	x4	x5	x6	x7	u1	u2	Свободные члены
x6	3.25	0	3.5	3.25	0.5	1	0	0.25	0	33.25
x2	0.25	1	0.5	1.25	-0.5	0	0	0.25	0	11.25
x3	-3.25	0	3.5	-3.25	-4.5	0	-1	-1.25	1	5.75
Δ	-8.25+3.25.3f M	0	-3.5-3.5.3f M	-16.25+3.25.3f M	-7.5+4.5.3f M	0	M	-1.25+2.25.3f M	0	-63.75-5.75.3f M

Базисные переменные	x1	x2	x3	x4	x5	x6	x7	u1	u2	Свободные члены
x6	3	-1	3	2	1	1	0	0	0	22
u1	1	4	2	5	-2	0	0	1	0	45
u2	-2	5	6	3	-7	0	-1	0	1	62

Промежуточная таблица

Базисные переменные	x1	x2	x3	x4	x5	x6	x7	u1	u2	Свободные члены
x6	3.25	0	3.5	3.25	0.5	1	0	0.25	0	33.25
x2	0.25	1	0.5	1.25	-0.5	0	0	0.25	0	11.25
x3	-3.25	0	3.5	-3.25	-4.5	0	-1	-1.25	1	5.75
Δ	-8.25+3.25.3f M	0	-3.5-3.5.3f M	-16.25+3.25.3f M	-7.5+4.5.3f M	0	M	-1.25+2.25.3f M	0	-63.75-5.75.3f M

Финальная таблица

Базисные переменные	x1	x2	x3	x4	x5	x6	x7	u1	u2	Свободные члены
x4	1	0	0	1	0.769231	0.153846	0.153846	0.23076923076923078	-0.15384615384615385	4.23077
x2	-1	1	0	0	-1.17582	-0.263736	-0.120879	0.03296703296703297	0.12087912087912088	3.17582
x3	0	0	1	0	-0.571429	0.142857	-0.142857	-0.14285714285714285	0.14285714285714285	5.57143
Δ	8	0	0	0	18	3	2	-2+M	-2+M	32

9.

$$z = 7x_1 - 5x_2 + x_3 + 10x_4 - 5x_5 \to \max;$$

$$\begin{cases} 3x_1 - x_2 + 3x_3 + 2x_4 + x_5 \le 22, \\ x_1 + 4x_2 + 2x_3 + 5x_4 - 2x_5 = 45, \\ -2x_1 + 5x_2 + 6x_3 + 3x_4 - 7x_5 \ge 62, \end{cases}$$

$$x_i \ge 0 \left(i = \overline{1,5}\right).$$

Аналитическое решение:

$$7 \cdot x_1 - 5 \cdot x_2 + x_3 + 10 \cdot x_4 - 5 \cdot x_5 \rightarrow \max$$
$$3 \cdot x_1 - x_2 + 3 \cdot x_3 + 2 \cdot x_4 + x_5 \leq 22$$
$$x_1 + 4 \cdot x_2 + 2 \cdot x_3 + 5 \cdot x_4 - 2 \cdot x_5 = 45$$
$$-2 \cdot x_1 + 5 \cdot x_2 + 6 \cdot x_3 + 3 \cdot x_4 - 7 \cdot x_5 \geq 62$$

Решение методом искусственного базиса

Для каждого ограничения с неравенством добавляем дополнительные переменные x_6 и x_7.
Ограничение 1 содержит неравенство, базисной будет добавленная дополнительная переменная x_6
Ограничение 2 содержит равенство. Базисная переменная для этого ограничения будет определена позднее.
Ограничение 3 содержит неравенство с ≥. Базисная переменная для этого ограничения будет определена позднее.

Начальная симплекс-таблица

C	7	-5	1	10	-5	0	0	0
базис	x_1	x_2	x_3	x_4	x_5	x_6	x_7	b
x_6	3	-1	3	2	1	1	0	22
$?_1$	1	4	2	5	-2	0	0	45
$?_2$	-2	5	6	3	-7	0	-1	62

Для ограничения 2 добавляем искусственную переменную u_1 и делаем её базисной.
Для ограничения 3 добавляем искусственную переменную u_2 и делаем её базисной.
В целевую функцию добавляем искусственные пременные с коэффициентом -M, где M — очень большое число.

Таблица с искусственными переменными

C	7	-5	1	10	-5	0	0	-M	-M	0
базис	x_1	x_2	x_3	x_4	x_5	x_6	x_7	u_1	u_2	b
x_6	3	-1	3	2	1	1	0	0	0	22
u_1	1	4	2	5	-2	0	0	1	0	45
u_2	-2	5	6	3	-7	0	-1	0	1	62

Перепишем условие задачи с учётом добавленных искусственных переменных:

$$F = 7x_1 - 5x_2 + 1x_3 + 10x_4 - 5x_5 - Mu_1 - Mu_2 \rightarrow max$$
$$3 \cdot x_1 - x_2 + 3 \cdot x_3 + 2 \cdot x_4 + x_5 + x_6 = 22$$
$$x_1 + 4 \cdot x_2 + 2 \cdot x_3 + 5 \cdot x_4 - 2 \cdot x_5 + u_1 = 45$$
$$- 2 \cdot x_1 + 5 \cdot x_2 + 6 \cdot x_3 + 3 \cdot x_4 - 7 \cdot x_5 - x_7 + u_2 = 62$$

Выразим искусственные переменные через базовые и дополнительные:

$$u_1 = 45 - x_1 - 4 \cdot x_2 - 2 \cdot x_3 - 5 \cdot x_4 + 2 \cdot x_5$$
$$u_2 = 62 + 2 \cdot x_1 - 5 \cdot x_2 - 6 \cdot x_3 - 3 \cdot x_4 + 7 \cdot x_5 + x_7$$

Вычисляем дельты: : $\Delta_i = C_6 \cdot a_{1i} + C_8 \cdot a_{2i} + C_9 \cdot a_{3i} - C_i$

$\Delta_1 = C_6 \cdot a_{11} + C_8 \cdot a_{21} + C_9 \cdot a_{31} - C_1 = 0 \cdot 3 + -M \cdot 1 + -M \cdot (-2) - 7 = -7 + M$
$\Delta_2 = C_6 \cdot a_{12} + C_8 \cdot a_{22} + C_9 \cdot a_{32} - C_2 = 0 \cdot (-1) + -M \cdot 4 + -M \cdot 5 - -5 = 5 - 9M$
$\Delta_3 = C_6 \cdot a_{13} + C_8 \cdot a_{23} + C_9 \cdot a_{33} - C_3 = 0 \cdot 3 + -M \cdot 2 + -M \cdot 6 - 1 = -1 - 8M$
$\Delta_4 = C_6 \cdot a_{14} + C_8 \cdot a_{24} + C_9 \cdot a_{34} - C_4 = 0 \cdot 2 + -M \cdot 5 + -M \cdot 3 - 10 = -10 - 8M$
$\Delta_5 = C_6 \cdot a_{15} + C_8 \cdot a_{25} + C_9 \cdot a_{35} - C_5 = 0 \cdot 1 + -M \cdot (-2) + -M \cdot (-7) - -5 = 5 + 9M$
$\Delta_6 = C_6 \cdot a_{16} + C_8 \cdot a_{26} + C_9 \cdot a_{36} - C_6 = 0 \cdot 1 + -M \cdot 0 + -M \cdot 0 - 0 = 0$
$\Delta_7 = C_6 \cdot a_{17} + C_8 \cdot a_{27} + C_9 \cdot a_{37} - C_7 = 0 \cdot 0 + -M \cdot 0 + -M \cdot (-1) - 0 = M$
$\Delta_8 = C_6 \cdot a_{18} + C_8 \cdot a_{28} + C_9 \cdot a_{38} - C_8 = 0 \cdot 0 + -M \cdot 1 + -M \cdot 0 - -M = 0$
$\Delta_9 = C_6 \cdot a_{19} + C_8 \cdot a_{29} + C_9 \cdot a_{39} - C_9 = 0 \cdot 0 + -M \cdot 0 + -M \cdot 1 - -M = 0$
$\Delta_b = C_6 \cdot b_1 + C_8 \cdot b_2 + C_9 \cdot b_3 - C_{10} = 0 \cdot 22 + -M \cdot 45 + -M \cdot 62 - 0 = -107M$

Симплекс-таблица с дельтами

C	7	-5	1	10	-5	0	0	-M	-M	0
базис	x_1	x_2	x_3	x_4	x_5	x_6	x_7	u_1	u_2	**b**
x_6	3	-1	3	2	1	1	0	0	0	22
u_1	1	4	2	5	-2	0	0	1	0	45
u_2	-2	5	6	3	-7	0	-1	0	1	62
Δ	-7 + M	5 - 9M	-1 - 8M	-10 - 8M	5 + 9M	0	M	0	0	-107M

Текущий план X: [0, 0, 0, 0, 0, 22, 0, 45, 62]
Целевая функция F: 7·0 + -5·0 + 1·0 + 10·0 + -5·0 + 0·22 + 0·0 + -M·45 + -M·62 = -107M
Проверяем план на оптимальность: план **не оптимален**, так как Δ_2 = 5 - 9M отрицательна.

Итерация 1

Определяем *разрешающий столбец* - столбец, в котором находится минимальная дельта: 2, Δ_2: 5 - 9M
Находим симплекс-отношения Q, путём деления коэффициентов b на соответствующие значения столбца 2
В найденном столбце ищем строку с наименьшим значением Q: $Q_{min} = \frac{45}{4}$, строка 2.
На пересечении найденных строки и столбца находится *разрешающий элемент*: 4
В качестве базисной переменной u_1 берём x_2.

C	7	-5	1	10	-5	0	0	-M	-M	0	
базис	x_1	x_2	x_3	x_4	x_5	x_6	x_7	u_1	u_2	**b**	**Q**
x_6	3	-1	3	2	1	1	0	0	0	22	-
x_2	1	4	2	5	-2	0	0	1	0	45	$45 / 4 = \frac{45}{4}$
u_2	-2	5	6	3	-7	0	-1	0	1	62	$62 / 5 = \frac{62}{5}$
Δ	-7 + M	5 - 9M	-1 - 8M	-10 - 8M	5 + 9M	0	M	0	0	-107M	

Делим строку 2 на 4. Из строк 1, 3 вычитаем строку 2, умноженную на соответствующий элемент в столбце 2.

Вычисляем новые дельты: $\Delta_i = C_6 \cdot a_{1i} + C_2 \cdot a_{2i} + C_9 \cdot a_{3i} - C_i$

$\Delta_1 = C_6 \cdot a_{11} + C_2 \cdot a_{21} + C_9 \cdot a_{31} - C_1 = 0 \cdot \frac{13}{4} + -5 \cdot \frac{1}{4} + -M \cdot (-\frac{13}{4}) - 7 = -\frac{33}{4} + \frac{13}{4}M$

$\Delta_2 = C_6 \cdot a_{12} + C_2 \cdot a_{22} + C_9 \cdot a_{32} - C_2 = 0 \cdot 0 + -5 \cdot 1 + -M \cdot 0 - -5 = 0$

$\Delta_3 = C_6 \cdot a_{13} + C_2 \cdot a_{23} + C_9 \cdot a_{33} - C_3 = 0 \cdot \frac{7}{2} + -5 \cdot \frac{1}{2} + -M \cdot \frac{7}{2} - 1 = -\frac{7}{2} - \frac{7}{2}M$

$\Delta_4 = C_6 \cdot a_{14} + C_2 \cdot a_{24} + C_9 \cdot a_{34} - C_4 = 0 \cdot \frac{13}{4} + -5 \cdot \frac{5}{4} + -M \cdot (-\frac{13}{4}) - 10 = -\frac{65}{4} + \frac{13}{4}M$

$\Delta_5 = C_6 \cdot a_{15} + C_2 \cdot a_{25} + C_9 \cdot a_{35} - C_5 = 0 \cdot \frac{1}{2} + -5 \cdot (-\frac{1}{2}) + -M \cdot (-\frac{9}{2}) - -5 = \frac{15}{2} + \frac{9}{2}M$

$\Delta_6 = C_6 \cdot a_{16} + C_2 \cdot a_{26} + C_9 \cdot a_{36} - C_6 = 0 \cdot 1 + -5 \cdot 0 + -M \cdot 0 - 0 = 0$

$\Delta_7 = C_6 \cdot a_{17} + C_2 \cdot a_{27} + C_9 \cdot a_{37} - C_7 = 0 \cdot 0 + -5 \cdot 0 + -M \cdot (-1) - 0 = M$

$\Delta_8 = C_6 \cdot a_{18} + C_2 \cdot a_{28} + C_9 \cdot a_{38} - C_8 = 0 \cdot \frac{1}{4} + -5 \cdot \frac{1}{4} + -M \cdot (-\frac{5}{4}) - -M = -\frac{5}{4} + \frac{9}{4}M$

$\Delta_9 = C_6 \cdot a_{19} + C_2 \cdot a_{29} + C_9 \cdot a_{39} - C_9 = 0 \cdot 0 + -5 \cdot 0 + -M \cdot 1 - -M = 0$

$\Delta_b = C_6 \cdot b_1 + C_2 \cdot b_2 + C_9 \cdot b_3 - C_{10} = 0 \cdot \frac{133}{4} + -5 \cdot \frac{45}{4} + -M \cdot \frac{23}{4} - 0 = -\frac{225}{4} - \frac{23}{4}M$

Симплекс-таблица с обновлёнными дельтами

C	7	-5	1	10	-5	0	0	-M	-M	0	
базис	x_1	x_2	x_3	x_4	x_5	x_6	x_7	u_1	u_2	**b**	**Q**
x_6	$\frac{13}{4}$	0	$\frac{7}{2}$	$\frac{13}{4}$	$\frac{1}{2}$	1	0	$\frac{1}{4}$	0	$\frac{133}{4}$	-
x_2	$\frac{1}{4}$	1	$\frac{1}{2}$	$\frac{5}{4}$	$-\frac{1}{2}$	0	0	$\frac{1}{4}$	0	$\frac{45}{4}$	$\frac{45}{4}$
u_2	$-\frac{13}{4}$	0	$\frac{7}{2}$	$-\frac{13}{4}$	$-\frac{9}{2}$	0	-1	$-\frac{5}{4}$	1	$\frac{23}{4}$	$\frac{62}{5}$
Δ	$-\frac{33}{4} + \frac{13}{4}M$	0	$-\frac{7}{2} - \frac{7}{2}M$	$-\frac{65}{4} + \frac{13}{4}M$	$\frac{15}{2} + \frac{9}{2}M$	0	M	$-\frac{5}{4} + \frac{9}{4}M$	0	$-\frac{225}{4} - \frac{23}{4}M$	

Текущий план X: [0, $\frac{45}{4}$, 0, 0, 0, $\frac{133}{4}$, 0, 0, $\frac{23}{4}$]

Целевая функция F: $7 \cdot 0 + -5 \cdot \frac{45}{4} + 1 \cdot 0 + 10 \cdot 0 + -5 \cdot 0 + 0 \cdot \frac{133}{4} + 0 \cdot 0 + -M \cdot 0 + -M \cdot \frac{23}{4} = -\frac{225}{4} - \frac{23}{4}M$

Проверяем план на оптимальность: план **не оптимален**, так как $\Delta_3 = -\frac{7}{2} - \frac{7}{2}M$ отрицательна.

Итерация 2

Определяем *разрешающий столбец* - столбец, в котором находится минимальная дельта: 3, Δ_3: $-\frac{7}{2} - \frac{7}{2}M$

Находим симплекс-отношения Q, путём деления коэффициентов b на соответствующие значения столбца 3

В найденном столбце ищем строку с наименьшим значением Q: $Q_{min} = \frac{23}{14}$, строка 3.

На пересечении найденных строки и столбца находится *разрешающий элемент*: $\frac{7}{2}$

В качестве базисной переменной u_2 берём x_3.

C	7	-5	1	10	-5	0	0	-M	-M	0	
базис	x_1	x_2	x_3	x_4	x_5	x_6	x_7	u_1	u_2	**b**	**Q**
x_6	$\frac{13}{4}$	0	$\frac{7}{2}$	$\frac{13}{4}$	$\frac{1}{2}$	1	0	$\frac{1}{4}$	0	$\frac{133}{4}$	$\frac{133}{4} / \frac{7}{2} = \frac{19}{2}$
x_2	$\frac{1}{4}$	1	$\frac{1}{2}$	$\frac{5}{4}$	$-\frac{1}{2}$	0	0	$\frac{1}{4}$	0	$\frac{45}{4}$	$\frac{45}{4} / \frac{1}{2} = \frac{45}{2}$
x_3	$-\frac{13}{4}$	0	$\frac{7}{2}$	$-\frac{13}{4}$	$-\frac{9}{2}$	0	-1	$-\frac{5}{4}$	1	$\frac{23}{4}$	$\frac{23}{4} / \frac{7}{2} = \frac{23}{14}$
Δ	$-\frac{33}{4} + \frac{13}{4}M$	0	$-\frac{7}{2} - \frac{7}{2}M$	$-\frac{65}{4} + \frac{13}{4}M$	$\frac{15}{2} + \frac{9}{2}M$	0	M	$-\frac{5}{4} + \frac{9}{4}M$	0	$-\frac{225}{4} - \frac{23}{4}M$	

Делим строку 3 на $\frac{7}{2}$. Из строк 1, 2 вычитаем строку 3, умноженную на соответствующий элемент в столбце 3.

Вычисляем новые дельты: $\Delta_i = C_6 \cdot a_{1i} + C_2 \cdot a_{2i} + C_3 \cdot a_{3i} - C_i$

$$\Delta_1 = C_6 \cdot a_{11} + C_2 \cdot a_{21} + C_3 \cdot a_{31} - C_1 = 0 \cdot \frac{13}{2} + -5 \cdot \frac{5}{7} + 1 \cdot (-\frac{13}{14}) - 7 = -\frac{23}{2}$$
$$\Delta_2 = C_6 \cdot a_{12} + C_2 \cdot a_{22} + C_3 \cdot a_{32} - C_2 = 0 \cdot 0 + -5 \cdot 1 + 1 \cdot 0 - -5 = 0$$
$$\Delta_3 = C_6 \cdot a_{13} + C_2 \cdot a_{23} + C_3 \cdot a_{33} - C_3 = 0 \cdot 0 + -5 \cdot 0 + 1 \cdot 1 - 1 = 0$$
$$\Delta_4 = C_6 \cdot a_{14} + C_2 \cdot a_{24} + C_3 \cdot a_{34} - C_4 = 0 \cdot \frac{13}{2} + -5 \cdot \frac{12}{7} + 1 \cdot (-\frac{13}{14}) - 10 = -\frac{39}{2}$$
$$\Delta_5 = C_6 \cdot a_{15} + C_2 \cdot a_{25} + C_3 \cdot a_{35} - C_5 = 0 \cdot 5 + -5 \cdot \frac{1}{7} + 1 \cdot (-\frac{9}{7}) - -5 = 3$$
$$\Delta_6 = C_6 \cdot a_{16} + C_2 \cdot a_{26} + C_3 \cdot a_{36} - C_6 = 0 \cdot 1 + -5 \cdot 0 + 1 \cdot 0 - 0 = 0$$
$$\Delta_7 = C_6 \cdot a_{17} + C_2 \cdot a_{27} + C_3 \cdot a_{37} - C_7 = 0 \cdot 1 + -5 \cdot \frac{1}{7} + 1 \cdot (-\frac{2}{7}) - 0 = -1$$
$$\Delta_8 = C_6 \cdot a_{18} + C_2 \cdot a_{28} + C_3 \cdot a_{38} - C_8 = 0 \cdot \frac{3}{2} + -5 \cdot \frac{3}{7} + 1 \cdot (-\frac{5}{14}) - -M = -\frac{5}{2} + M$$
$$\Delta_9 = C_6 \cdot a_{19} + C_2 \cdot a_{29} + C_3 \cdot a_{39} - C_9 = 0 \cdot (-1) + -5 \cdot (-\frac{1}{7}) + 1 \cdot \frac{2}{7} - -M = 1 + M$$
$$\Delta_b = C_6 \cdot b_1 + C_2 \cdot b_2 + C_3 \cdot b_3 - C_{10} = 0 \cdot \frac{55}{2} + -5 \cdot \frac{73}{7} + 1 \cdot \frac{23}{14} - 0 = -\frac{101}{2}$$

Симплекс-таблица с обновлёнными дельтами

C	7	-5	1	10	-5	0	0	-M	-M	0	
базис	x_1	x_2	x_3	x_4	x_5	x_6	x_7	u_1	u_2	**b**	**Q**
x_6	$\frac{13}{2}$	0	0	$\frac{13}{2}$	5	1	1	$\frac{3}{2}$	-1	$\frac{55}{2}$	$\frac{19}{2}$
x_2	$\frac{5}{7}$	1	0	$\frac{12}{7}$	$\frac{1}{7}$	0	$\frac{1}{7}$	$\frac{3}{7}$	$-\frac{1}{7}$	$\frac{73}{7}$	$\frac{45}{2}$
x_3	$-\frac{13}{14}$	0	1	$-\frac{13}{14}$	$-\frac{9}{7}$	0	$-\frac{2}{7}$	$-\frac{5}{14}$	$\frac{2}{7}$	$\frac{23}{14}$	$\frac{23}{14}$
Δ	$-\frac{23}{2}$	0	0	$-\frac{39}{2}$	3	0	-1	$-\frac{5}{2} + M$	$1 + M$	$-\frac{101}{2}$	

Текущий план X: $[\ 0, \frac{73}{7}, \frac{23}{14}, 0, 0, \frac{55}{2}, 0, 0, 0\ ]$

Целевая функция F: $7 \cdot 0 + -5 \cdot \frac{73}{7} + 1 \cdot \frac{23}{14} + 10 \cdot 0 + -5 \cdot 0 + 0 \cdot \frac{55}{2} + 0 \cdot 0 + -M \cdot 0 + -M \cdot 0 = -\frac{101}{2}$

Проверяем план на оптимальность: план **не оптимален**, так как $\Delta_1 = -\frac{23}{2}$ отрицательна.

Итерация 3

Определяем *разрешающий столбец* - столбец, в котором находится минимальная дельта: 4, Δ_4: $-\frac{39}{2}$

Находим симплекс-отношения Q, путём деления коэффициентов b на соответствующие значения столбца 4

В найденном столбце ищем строку с наименьшим значением Q: $Q_{min} = \frac{55}{13}$, строка 1.

На пересечении найденных строки и столбца находится *разрешающий элемент*: $\frac{13}{2}$

В качестве базисной переменной x_6 берём x_4.

C	7	-5	1	10	-5	0	0	-M	-M	0	
базис	x_1	x_2	x_3	x_4	x_5	x_6	x_7	u_1	u_2	**b**	**Q**
x_4	$\frac{13}{2}$	0	0	$\frac{13}{2}$	5	1	1	$\frac{3}{2}$	-1	$\frac{55}{2}$	$\frac{55}{2} / \frac{13}{2} = \frac{55}{13}$
x_2	$\frac{5}{7}$	1	0	$\frac{12}{7}$	$\frac{1}{7}$	0	$\frac{1}{7}$	$\frac{3}{7}$	$-\frac{1}{7}$	$\frac{73}{7}$	$\frac{73}{7} / \frac{12}{7} = \frac{73}{12}$
x_3	$-\frac{13}{14}$	0	1	$-\frac{13}{14}$	$-\frac{9}{7}$	0	$-\frac{2}{7}$	$-\frac{5}{14}$	$\frac{2}{7}$	$\frac{23}{14}$	-
Δ	$-\frac{23}{2}$	0	0	$-\frac{39}{2}$	3	0	-1	$-\frac{5}{2} + M$	$1 + M$	$-\frac{101}{2}$	

Делим строку 1 на $\frac{13}{2}$. Из строк 2, 3 вычитаем строку 1, умноженную на соответствующий элемент в столбце 4.

Вычисляем новые дельты: $\Delta_i = C_4 \cdot a_{1i} + C_2 \cdot a_{2i} + C_3 \cdot a_{3i} - C_i$

$$\Delta_1 = C_4 \cdot a_{11} + C_2 \cdot a_{21} + C_3 \cdot a_{31} - C_1 = 10 \cdot 1 + -5 \cdot (-1) + 1 \cdot 0 - 7 = 8$$
$$\Delta_2 = C_4 \cdot a_{12} + C_2 \cdot a_{22} + C_3 \cdot a_{32} - C_2 = 10 \cdot 0 + -5 \cdot 1 + 1 \cdot 0 - -5 = 0$$
$$\Delta_3 = C_4 \cdot a_{13} + C_2 \cdot a_{23} + C_3 \cdot a_{33} - C_3 = 10 \cdot 0 + -5 \cdot 0 + 1 \cdot 1 - 1 = 0$$
$$\Delta_4 = C_4 \cdot a_{14} + C_2 \cdot a_{24} + C_3 \cdot a_{34} - C_4 = 10 \cdot 1 + -5 \cdot 0 + 1 \cdot 0 - 10 = 0$$
$$\Delta_5 = C_4 \cdot a_{15} + C_2 \cdot a_{25} + C_3 \cdot a_{35} - C_5 = 10 \cdot \frac{10}{13} + -5 \cdot (-\frac{107}{91}) + 1 \cdot (-\frac{4}{7}) - -5 = 18$$
$$\Delta_6 = C_4 \cdot a_{16} + C_2 \cdot a_{26} + C_3 \cdot a_{36} - C_6 = 10 \cdot \frac{2}{13} + -5 \cdot (-\frac{24}{91}) + 1 \cdot \frac{1}{7} - 0 = 3$$
$$\Delta_7 = C_4 \cdot a_{17} + C_2 \cdot a_{27} + C_3 \cdot a_{37} - C_7 = 10 \cdot \frac{2}{13} + -5 \cdot (-\frac{11}{91}) + 1 \cdot (-\frac{1}{7}) - 0 = 2$$
$$\Delta_8 = C_4 \cdot a_{18} + C_2 \cdot a_{28} + C_3 \cdot a_{38} - C_8 = 10 \cdot \frac{3}{13} + -5 \cdot \frac{3}{91} + 1 \cdot (-\frac{1}{7}) - -M = 2 + M$$
$$\Delta_9 = C_4 \cdot a_{19} + C_2 \cdot a_{29} + C_3 \cdot a_{39} - C_9 = 10 \cdot (-\frac{2}{13}) + -5 \cdot \frac{11}{91} + 1 \cdot \frac{1}{7} - -M = -2 + M$$
$$\Delta_b = C_4 \cdot b_1 + C_2 \cdot b_2 + C_3 \cdot b_3 - C_{10} = 10 \cdot \frac{55}{13} + -5 \cdot \frac{289}{91} + 1 \cdot \frac{39}{7} - 0 = 32$$

Симплекс-таблица с обновлёнными дельтами

C	7	-5	1	10	-5	0	0	-M	-M	0	
базис	x_1	x_2	x_3	x_4	x_5	x_6	x_7	u_1	u_2	**b**	**Q**
x_4	1	0	0	1	$\frac{10}{13}$	$\frac{2}{13}$	$\frac{2}{13}$	$\frac{3}{13}$	$-\frac{2}{13}$	$\frac{55}{13}$	$\frac{55}{13}$
x_2	-1	1	0	0	$-\frac{107}{91}$	$-\frac{24}{91}$	$-\frac{11}{91}$	$\frac{3}{91}$	$\frac{11}{91}$	$\frac{289}{91}$	$\frac{73}{12}$
x_3	0	0	1	0	$-\frac{4}{7}$	$\frac{1}{7}$	$-\frac{1}{7}$	$-\frac{1}{7}$	$\frac{1}{7}$	$\frac{39}{7}$	-
Δ	8	0	0	0	18	3	2	2 + M	-2 + M	32	

Текущий план X: [0, $\frac{289}{91}$, $\frac{39}{7}$, $\frac{55}{13}$, 0, 0, 0, 0, 0]

Целевая функция F: $7 \cdot 0 + -5 \cdot \frac{289}{91} + 1 \cdot \frac{39}{7} + 10 \cdot \frac{55}{13} + -5 \cdot 0 + 0 \cdot 0 + 0 \cdot 0 + -M \cdot 0 + -M \cdot 0 = 32$

Проверяем план на оптимальность: отрицательные дельты отсутствуют, следовательно **план оптимален**.

Ответ: $x_1 = 0, x_2 = \frac{289}{91}, x_3 = \frac{39}{7}, x_4 = \frac{55}{13}, x_5 = 0, F = 32$

Вывод: в ходе выполнения работы были изучены методы искусственного базиса и больших штрафов решения задач ЛП в канонической форме, не подготовленных к работе симплекс-методом в чистом виде.

Проведённые задания по подготовке к работе помогли разработать программу для решения задач линейного программирования искусственного базиса и больших штрафов. Программа преобразует систему линейных ограничений в таблицу, определяет оптимальное решение путём выбора опорного элемента и выполнения шагов методов, и наконец, записывает значения целевой функции и переменных для анализа. Результатом работы программы являются точки максимума, где линейная целевая функция достигает оптимального значения при соблюдении заданных линейных ограничений. Была создана программа,

которая правильно реализует методы искусственного базиса и больших штрафов.