ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ»

Факультет информационных технологий и программирования

Дисциплина:

«Прикладная математика»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1

Выполнил:

М33091 Ларин В.Д.

Проверила:

Москаленко М. А.

Санкт-Петербург

2021 г.

Постановка задачи:
Необходимо решить задачу линейного программирования. Пусть в задаче есть m ограничений, а целевая функция зависит от n основных переменных.

Алгоритм

Привидение к каноническому виду:
Первым делом необходимо привести все ограничения к каноническому виду — виду, в котором все условия задаются равенствами. Для этого предварительно все неравенства с ≥ умножаются на -1, для получения неравенств с ≤.

Чтобы привести ограничения с неравенствами к каноническому виду, для каждого ограничения вводят переменную, называемую дополнительной с коэффициентом 1. В ответе эти переменные учитываться не будут, однако сильно упростят начальные вычисления. При этом дополнительные переменные являются базисными, а потому могут быть использованы для формирования начального опорного решения.

Формирование начального базиса:

После того как задача приведена к каноническому виду, необходимо найти начальный базис для формирования первого опорного решения. Если в процессе приведения были добавлены дополнительные переменные, то они становятся базисными.

Иначе необходимо выделить среди коэффициентов ограничений столбец, который участвует в формировании единичной матрицы в заданной строке (например, если требуется определить вторую базисную переменную, то необходимо искать столбец, в котором второе число равно 1, а остальные равны нулю). Если такой столбец найден, то переменная, соответствующая этому столбцу, становится базисной.

В противном случае можно поискать столбец, в котором все значения кроме числа в заданной строке равны нулю, и, если он будет найден, то разделить всезначения строки на число, стоящее на пересечении этих строки и столбца, тем самым образовав столбец, участвующий в формировании единичной матрицы.

Если такой столбец отсутствует, то для формирования базиса необходимо применить исключение Гаусса для первого ненулевого столбца, который еще не является базисным. Для этого вся строка делится на элемент в найденном столбце, а из остальных строк вычитается полученная строка, разделенная на значение, стоящее в этом же столбце. После этой операции все значения вне данной строки будут обнулены, и столбец можно будет считать базисным.
После того как базис сформирован, нужно построить начальную симплекс-таблицу. Она строится следующим образом:

- Для удобства в первой строке можно записать коэффициенты C_i целевой функции (для дополнительных переменных эти коэффициенты равны нулю)
- Вторая строка формирует шапку таблицы. В ней первый столбец называется базис, а остальные перечисляют основные переменные $x_1..x_n$ идополнительные $x_{n+1}..x_{n+k}$
- Затем построчно перечисляются базисные переменные и коэффициенты ограничений

Схематично начальная таблица будет выглядеть примерно так:

C	c_1	c_2	...	c_n	0	0	...	0	0
базис	$\mathbf{x_1}$	$\mathbf{x_2}$	...	$\mathbf{x_n}$	$\mathbf{x_{n+1}}$	$\mathbf{x_{n+2}}$	...	$\mathbf{x_{n+k}}$	**b**
$\mathbf{x_{e_1}}$	a_{11}	a_{12}	...	a_{1n}	a_{1n+1}	a_{1n+2}	...	a_{1n+k}	b_1
$\mathbf{x_{e_2}}$	a_{21}	a_{22}	...	a_{2n}	a_{2n+1}	a_{2n+2}	...	a_{2n+k}	b_2
...	...	...	...	...	...	...	...	...	...
$\mathbf{x_{e_m}}$	a_{m1}	a_{m2}	...	a_{mn}	a_{mn+1}	a_{mn+2}	...	a_{mn+k}	b_m

Избавляемся от отрицательных свободных коэффициентов:

После приведения к каноническому виду или после алгебраических преобразований при формировании базиса некоторые из свободных коэффициентов (b_i) могли стать отрицательными, что не позволяет перейти к дальнейшим вычислениям. Чтобы избавиться от отрицательных значений b необходимо:

- Найти строку, в которой находится максимальное по модулю значение b. Пусть это будет строка i;
- Найти максимальный по модулю элемент в этой строке. Пусть он находится в столбце j;
- Строку i разделить на элемент, стоящий на пересечении i-ой строки и j-го столбца;
- Из каждой оставшейся строки k вычесть строку i, умноженную на элемент строки k и столбца j;
- Переменную, соответствующую найденному столбцу j, сделать базисной (добавить в базис вместо переменной, находящейся в строке i).

Этот шаг необходимо повторять до тех пор, пока все отрицательные b не станут положительными или в строке не останется отрицательных элементов. Если строка с максимальным по модулю b_i не содержит отрицательных элементов, то такая задача не имеет решений и на этом алгоритм заканчивает свою работу. В противном случае все b_i положительны и алгоритм переходит к следующему этапу — расчёту дельт.

Расчёт дельт:

Дельты — это параметры, на основании которых проверяется оптимальность текущего решения и улучшается функция. Они рассчитываются для каждой из переменных ограничений и записываются последней строкой таблицы.
Для расчёта дельт используется следующая формула: $\Delta_i = c_{e1} \cdot a_{1i} + c_{e2} \cdot a_{2i} + ... + c_{em} \cdot a_{mi} - c_i$. Проще говоря, чтобы вычислить дельту по заданной i-ой переменной, нужно перемножить коэффициенты условий в i-ом столбце на коэффициенты целевой функции при соответствующих базисных переменных, сложить эти произведения и вычесть из полученной суммы коэффициент целевой функции столбца i.

Проверка плана на оптимальность:

После того как дельты рассчитаны, необходимо проверить оптимальность текущего плана. Критерий оптимальности формулируется следующим образом:
При максимизации функции: *текущее решение считается оптимальным, если в таблице отсутствуют отрицательные дельты*.
При минимизации функции: *текущее решение считается оптимальным, еслив таблице отсутствуют положительные дельты*.

Если текущий план оптимален, то алгоритм завершает свою работу. Значениям переменных соответствуют значения столбца свободных коэффициентов b. Если свободной переменной нет в базисе, то её значение считается нулевым. Значение целевой функции, принимаемой на данном наборе, находится в строке с дельтами в том же столбце. Если какое-либо из значений столбца b отрицательно, то решения задачи не существует.

Переход к более оптимальному решению:

Если текущий план оказался не оптимальным, то алгоритм ищет столбец с наименьшей (с наибольшей, если ищется минимум) дельтой. После чего вычисляются симплекс-отношения Q. Для этого значения свободных коэффициентов делятся на ненулевые коэффициенты из найденного столбца. Если результат деления получается отрицательным, то такие отношение игнорируются.
Среди найденных симплекс-отношений ищется строка, в которой находится симплекс-отношение с наименьшим значением. Если таких отношений нет, то алгоритм останавливает свою работу, так как *целевая функция не ограничена и решения не существует*.

В противном случае строка с наименьшим отношением считается разрешающей и, аналогично избавлению от отрицательных свободных коэффициентов, делится на разрешающий элемент, расположенный в найденных столбце и строке, и из остальных строк вычитается найденная строка, разделенная на значения, стоящие в этом же столбце соответствующей строки. Переменная, стоящая в разрешающем столбце заменяет базисную переменную, находящуюся в найденной строке.
После этого вычисляются новые дельты и проверяется новый план. Так продолжается до тех пор, пока не будет выполнен критерий оптимальности плана или не будет установлено, что решения не существует.

Листинг кода:

```
import numpy as np
import math


class SimplexMethod:
    def __init__(self, c, A, b):
        self.table = np.c_[A, b]  # Симплекс-таблица, которую мы строим
        self.c = np.append(c, 0)
        self._find_basis()  # Определяет изначальный базис (набор переменных,
которые будут выражены через ограничения)
        self._check_negative_b()
        self._update_deltas()

    def _is_basic(self, col_row):
        return sum(col_row) == 1 and len([c for c in col_row if c == 0]) ==
len(col_row) - 1

    def _pivot_step(self, pivot_position):  # Меняем в симплекс-таблице свободные
переменные на базисные
        new_table = np.zeros(shape=self.table.shape)
        i, j = pivot_position
        pivot_value = self.table[i][j]
        new_table[i] = np.array(self.table[i]) / pivot_value  # Делим строку
опроного элемента на него самого
        for eq_i, eq in enumerate(self.table):
            if eq_i != i:
                multiplier = np.array(new_table[i]) * self.table[eq_i][j] # Метод
исключения Гаусса-Жордана
                new_table[eq_i] = np.array(self.table[eq_i]) - multiplier
        self.basis[i] = j
        self.table = new_table

    def _find_basis(self):  # Подготвока к дальнейшему решению
        self.basis = [-1] * (self.table.shape[0])
        for i, row in enumerate(self.table): # Индекс и значение/ строчка и
значения в строчке
            column_j = next(j for j, x in enumerate(row) if x != 0 and j not in
self.basis) # Еще не использовали для базиса (чтобы не делал базис одного вида)
            self._pivot_step((i, column_j))

    def _check_negative_b(self, maxiter=1000):  # Среди свободных коэфициентов не
должно быть отрицательных чисел
        b = self.table[:, -1]
        it = 0
        while it < maxiter + 1:
            if all(x >= 0 for x in b):
                return
            max_b = max(b.min(), b.max(), key=abs)
            i = np.where(max_b)[0][0]
            row = self.table[i][:-1]
            if not any(x < 0 for x in row):
                raise Exception("No solution. No negative number in row.")
            max_row = max(row.min(), row.max(), key=abs)
            j = np.where(max_row)[0][0]
            self._pivot_step((i, j))
            b = self.table[:, -1]
            it += 1
        if it == maxiter:
            raise Exception("No solution. Maxiter exceeded")
```

```
    def _update_deltas(self):  # Видоизменяем дельты
        ci = np.array([self.c[i] for i in self.basis])
        deltas = []
        for j in range(self.table.shape[1]):
            d = self.table[:, j] @ ci.T - self.c[j]  # @ - перемножение матриц
            deltas += [d]
        D = np.array(deltas)
        self.D = D

    def _can_max_progress(self):
        D = self.D[:-1]
        return any(d for d in D if d < 0) # Хотя бы один элемент отрицательный в
дельта

    def _get_max_position(self):  # Координаты опорного элемента (пайвота)
        D = self.D[:-1] # не берем дельту для b (значение целевой функции)
        column = np.argmin(D)
        xj = self.table[:, column]  # выделили столб
        b = self.table[:, -1]  # свободные коэффициенты
        simplex_Q = [math.inf if x <= 0 else b[i] / x for i, x in enumerate(xj)]
        # Набор отношений свободных коэффициентов к иксу / присваиваем
бесконечность, для того чтобы проверить условие
        if (all([r == math.inf for r in simplex_Q])):
            raise Exception("Linear program is unbounded.")
        row = simplex_Q.index(min(simplex_Q))
        return row, column

    def maximize(self):  # Максимазация функции
        while self._can_max_progress():  # Ищем до тех пор пока есть решение
            position = self._get_max_position()
            self._pivot_step(position)
            self._update_deltas()
        return self._get_solution()

    def minimize(self):
        self.c *= -1
        solution = self.maximize()
        self.c *= -1
        return solution

    def _get_solution(self):
        solution = np.zeros(shape=self.c.shape)
        for i in self.basis:
            one_index = np.where(self.table[:, i] == 1)[0][0] #
            solution[i] = self.table[:, -1][one_index]
        return solution

    def function_value(self, X):
        return self.c @ X.T
```

Вопросы на защиту

1. **Общая и каноническая форма задачи линейного программирования:**
   - Общая форма - это задача *максимизации или минимизации* линейной функции с линейными ограничениями в виде как равенств, так и неравенств.
   - Каноническая форма - это задача *максимизации* линейной функции с линейными ограничениями в виде равенств.
   - Указанные формы ЗЛП эквивалентны в том смысле, что каждая из них может быть приведена к любой другой путем несложных тождественных преобразований.
   - Чтобы привести задачу к канонической форме, необходимо предварительно все неравенства с ≥ умножить на -1, для получения неравенств с ≤.
   - Чтобы привести ограничения с неравенствами ≤ к каноническому виду, для каждого ограничения вводят переменную, называемую дополнительной с коэффициентом 1.
2. **Двойственная задачи ЛП:**

   Двойственная задача для заданной задачи линейного программирования - это другая задача линейного программирования, которая получается из исходной (прямой) задачи следующим образом:

   Каждая переменная в прямой задаче становится ограничением двойственной задачи;

   Каждое ограничение в прямой задаче становится переменной в двойственной задаче;

   Направление цели обращается – максимум в прямой задаче становится минимумом в двойственной, и наоборот.

3. **Метод искусственного базиса:**
    - Метод искусственного базиса используется для нахождения допустимого базисного решения задачи линейного программирования, когда в условии присутствуют ограничения типа равенств.
    - Аналогично базовому симплекс-методу для всех ограничений с неравенством вводятся дополнительные переменные, причём для ограничений с ≥ они берутся с коэффициентом -1, а для ограничений с ≤ с коэффициентом 1. Ограничения с равенством остаются без изменений.
    - Если свободный коэффициент какого-либо из ограничений меньше нуля, то такое ограничение умножается на -1 (знак неравенства при этом меняется на противоположный)
    - Для того, чтобы сформировать начальный базис в первую очередь можно поискать столбец, у которого одно значение равно единице, а все значения остальные значения равны нулю, и сделать соответствующую переменную базисной для этой строки.
    - Если найти такой столбец не удалось, то делают следующее:
      Для всех ограничений, не имеющих базисной переменной, добавляем искусственную переменную с коэффициентом 1. В целевую функцию добавляем эту же переменную с коэффициентов **-M**, если ищется максимум или с коэффициентом **M**, если ищется минимум. **M** всего лишь является очень большим числом.
    - После того, как начальный базис сформирован необходимо вычислить дельты. Дельты вычисляются полностью аналогично базовому методу: $\Delta_i = c_{e1} \cdot a_{1i} + c_{e2} \cdot a_{2i} + ... + c_{em} \cdot a_{mi} - c_i$.
    - Единственным отличием будет тот факт, что результат может содержать значения с M.
    - Когда дельты будут получены необходимо проверить текущий опорный план на оптимальность.
    - Если план оптимален, то алгоритм завершает свою работу, иначе формирует более оптимальное решение и повторяет процесс.

4. **Доказать, что ОДР (область допустимых решений) является выпуклым множеством:**

Теорема:

Множество всех допустимых решений системы ограничений задачи ЛП является выпуклым.

Доказательство:

Приведем матричную форму записи канонической задачи:

$$F = C * X \to max\ (min)$$
$$\begin{cases} A * X = B \\ X \geq 0 \end{cases}$$

Здесь С - матрица-строка, А - матрица-системы, Х - матрица-столбец переменных, В - матрица-столбец свободных членов:

$$C = (c_1, \dots, c_n);\quad A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \cdots & \cdots & \cdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix};\quad X = \begin{pmatrix} x_1 \\ \cdots \\ x_n \end{pmatrix};\quad B = \begin{pmatrix} b_1 \\ \dots \\ b_m \end{pmatrix}$$

Пусть $X_1,\ X_2$ - два допустимых решения задачи ЛП, заданной в матричной форме. Тогда $AX_1 = B,\ X_1 \geq 0$ и $AX_2 = B,\ X_2 \geq 0$. Рассмотрим выпуклую линейную комбинацию решений X_1, X_2:

$$X = \alpha_1 X_1 + \alpha_2 X_2 \text{ при } \alpha_1 \geq 0,\ \alpha_2 \geq 0,\ \alpha_1 + \alpha_2 = 1,$$

и покажем, что Х также является допустимым решением системы.

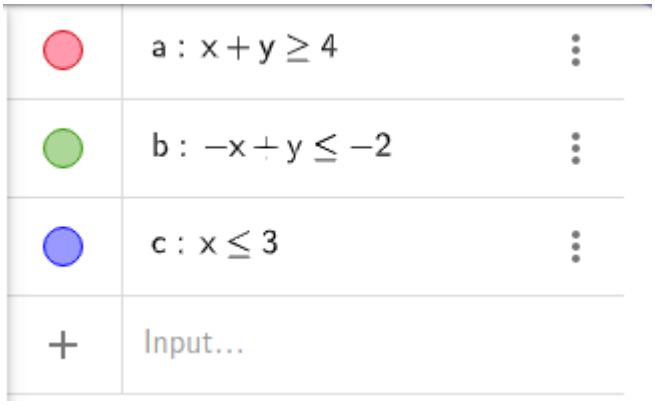
Действительно:

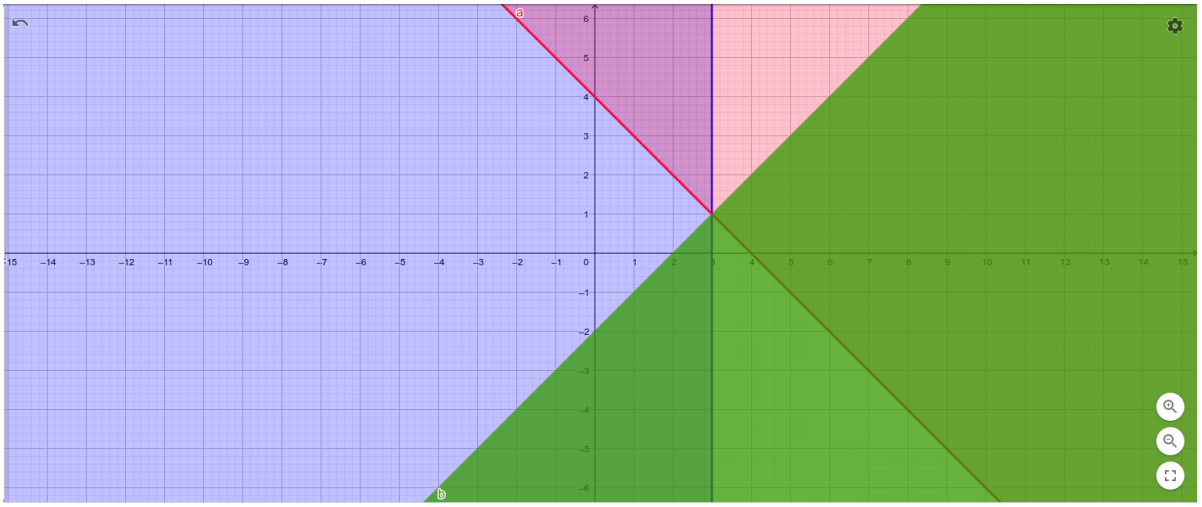
$$AX = A((1-\alpha_2)X_1 + \alpha_2 X_2) = (1-\alpha_2)AX_1 + \alpha_2 AX_2 = (1-\alpha_2)B + \alpha_2 B = B,$$

т.е. Х удовлетворяет системе. Но так как $X_1 \geq 0,\ X_2 \geq 0, \alpha_1 \geq 0, \alpha_2 \geq 0,$ то и Х > 0, т.е. решение удовлетворяет неотрицательности.

Таким образом, множество всех решений задачи ЛП является выпуклым (выпуклой многогранной областью).

5. **Может ли ОДР в задаче линейного программирования состоять из одной единственной точки? Если да, то приведите пример:**





Тут 2 переменные (для удобства визуализации) и несколько ограничений

Первые 2 ограничения выделили допустимые иксы: [3, +беск). Однако третье ограничение оставило от области 1 точку, в которой функция будет принимать и свое максимально и свое минимально значение в данной области (так как 1 точка)

6. **В чём заключается графический метод решения задачи ЛП?**
   - Графический метод применяется для задач линейного программирования с двумя переменными, когда ограничения выражены неравенствами, и задач со многими переменными при условии, что в их канонической записи содержится не более двух свободных переменных.
   - Каждое из неравенств задачи линейного программирования определяет на координатной плоскости (x1, x2) некоторую полуплоскость. Пересечение этих полуплоскостей задает область допустимых решений (ОДР), то есть любая точка из этой области является решением системы ограничений.
   - В общем случае область допустимых решений может быть представлена одной из следующих фигур: выпуклым многоугольником, неограниченной многоугольной областью, лучом, отрезком, точкой или пустой областью. В последнем случае говорят, что ограничения не совместны. Если система ограничений включает равенства, то они определяют на координатной плоскости (x1, x2) прямую линию. Область допустимых решений всегда представляет собой выпуклую фигуру.

Алгоритм решения задачи ЛП графическим методом:

- Находим область допустимых решений системы ограничений задачи. Для этого каждое из неравенств системы заменяем равенством и строим соответствующие этим равенствам граничные прямые. Каждая из построенных прямых делит плоскость на две полуплоскости. Чтобы графически определить, по какую сторону от граничной прямой располагается полуплоскость, содержащая решения, удовлетворяющие рассматриваемому неравенству, достаточно проверить одну какую-либо точку, не лежащую на прямой (например (0,0)). Если при подстановке ее координат в левую часть неравенства оно выполняется, то надо заштриховать полуплоскость, содержащую данную точку. Если же неравенство не выполняется, надо заштриховать полуплоскость, не содержащую данную точку. Отмечаем общую область для всех неравенств. Таким образом, получим область допустимых решений рассматриваемой задачи ЛП.
- Формируем графическое изображение целевой функции: Приравняем целевую функцию к постоянной величине L: $L = c1\ x1 + c2\ x2$. Это уравнение при фиксированном значении L определяет прямую, а при изменении L семейство параллельных прямых,

каждая из которых называется линией уровня. Проводим линию уровня L0.

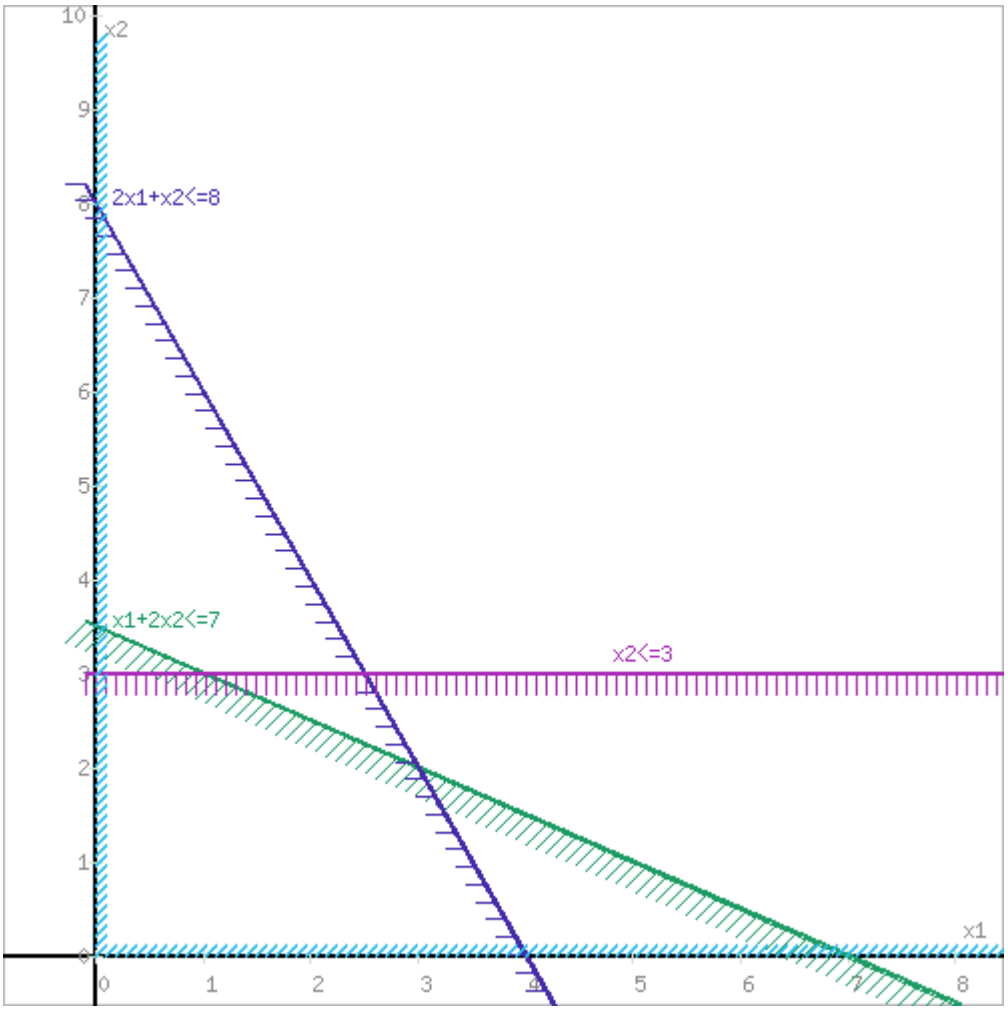
- Определяем направление возрастания целевой функции (вектор □). Для определения направления максимального возрастания значения целевой функции строим вектор-градиент целевой функции, который начинается в точке (0,0), заканчивается в точке (c1, c2). Если линия уровня и вектор-градиент построены верно, то они будут перпендикулярны.
- Находим оптимальное решение задачи ЛП. Линию уровня перемещаем по направлению вектора □ для задач на максимум и в направлении, противоположном □ , для задач на минимум. Перемещение линии уровня производится до тех пор, пока у нее окажется только одна общая точка с областью допустимых решений (ОДР). Эта точка определяет единственное решение задачи ЛП и будет точкой экстремума. Если окажется, что линия уровня параллельна одной из сторон ОДР, то задача ЛП будет иметь бесчисленное множество решений. Если ОДР представляет неограниченную область, то целевая функция может быть неограниченна. Задача ЛП может быть неразрешима, когда определяющие ее ограничения окажутся противоречивыми.
- Находим координаты точки экстремума и значение целевой функции в этой точке. Для вычисления координат оптимальной точки решим систему уравнений прямых, на пересечении которых находится эта точка. Подставляя найденный результат в целевую функцию, получим искомое оптимальное значение целевой функции.

7. **Используя графический метод, найти решение задачи линейного программирования:**

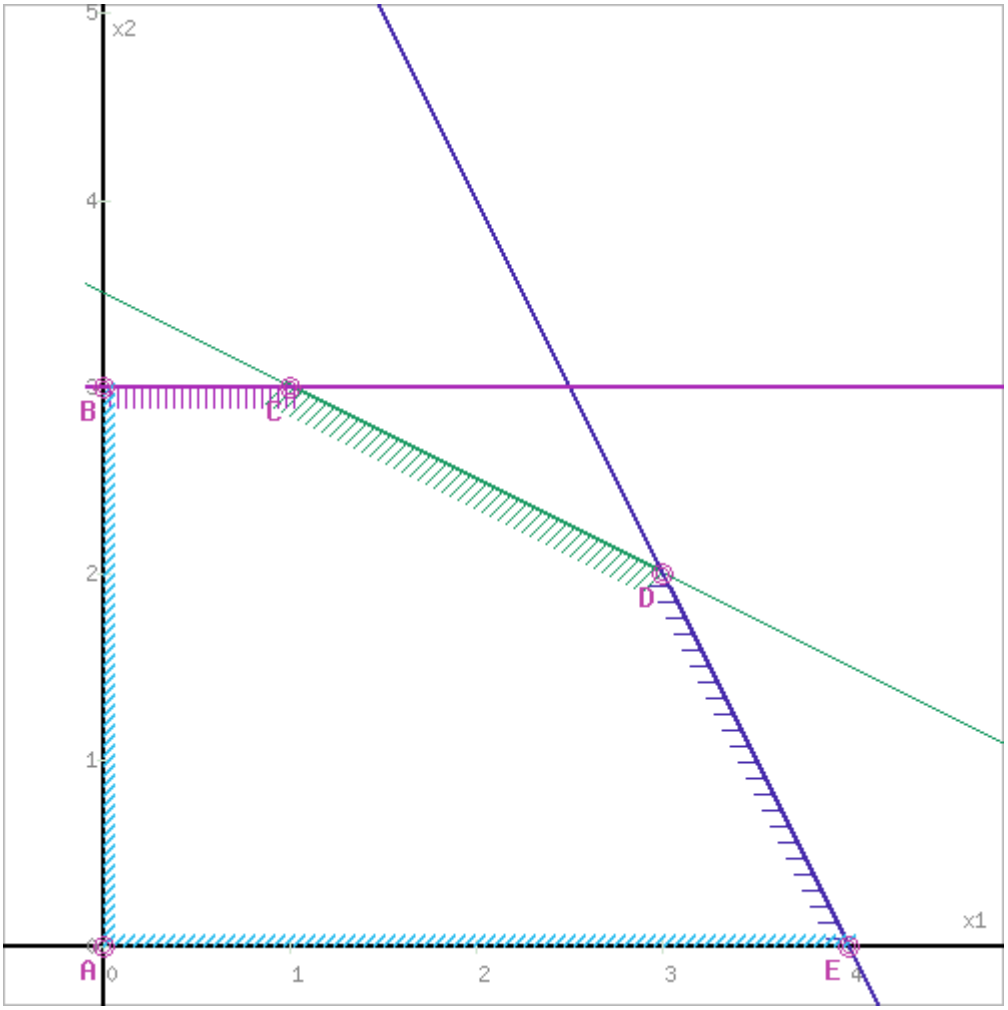
$$f(x) = -3x_1 - 2x_2 \to \min,$$

$$\begin{cases} x_1 + 2x_2 \le 7, \\ 2x_1 + x_2 \le 8, \\ x_2 \le 3, \\ x_1 \ge 0, \quad x_2 \ge 0. \end{cases}$$

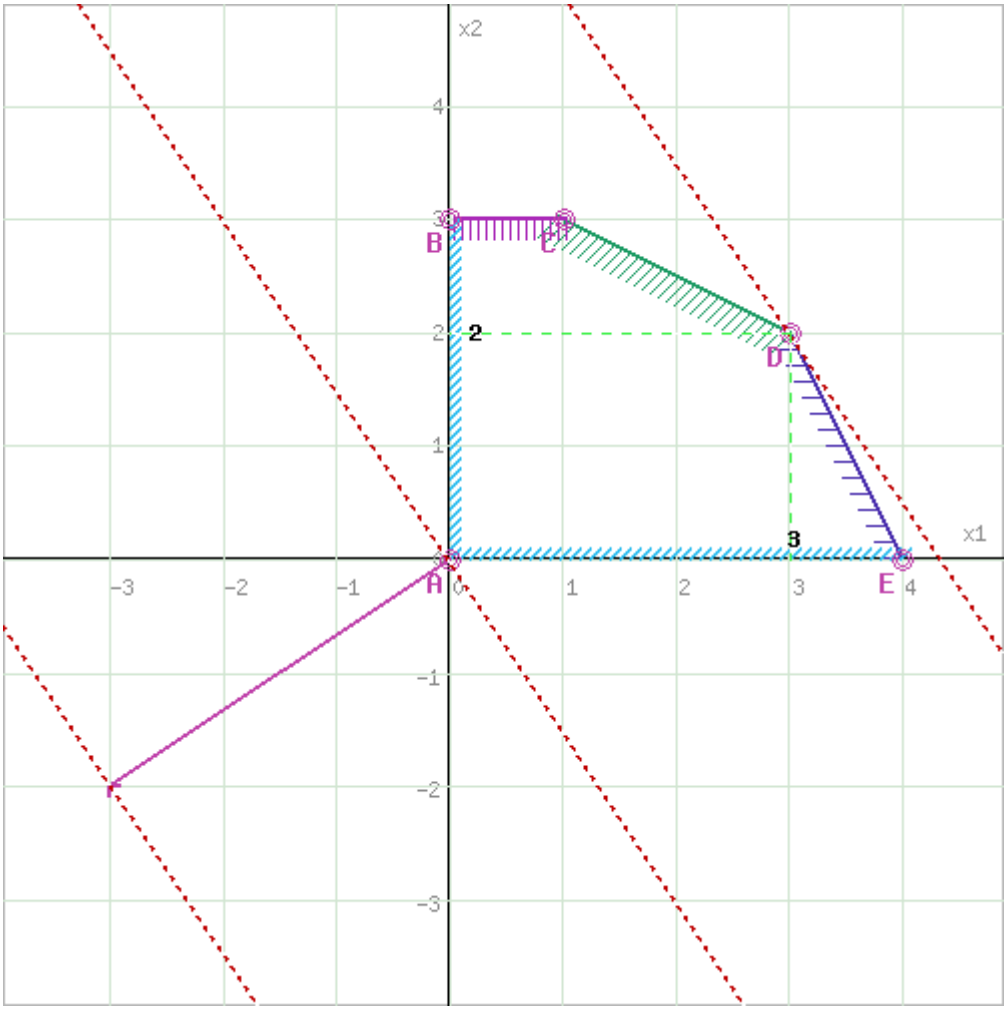
Построим область допустимых решений, т.е. решим графически систему неравенств. Для этого построим каждую прямую и определим полуплоскости, заданные неравенствами (полуплоскости обозначены штрихом).



Пересечением полуплоскостей будет являться область, координаты точек которого удовлетворяют условию неравенствам системы ограничений задачи. Обозначим границы области многоугольника решений.



Построим прямую, отвечающую значению функции $F = -3x_1-2x_2 = 0$. Вектор-градиент, составленный из коэффициентов целевой функции, указывает направление максимизации F(X). Начало вектора – точка (0; 0), конец – точка (-3;-2). Будем двигать эту прямую параллельным образом. Поскольку нас интересует минимальное решение, поэтому двигаем прямую до первого касания обозначенной области. На графике эта прямая обозначена пунктирной линией.



Прямая **F(x) = const** пересекает область в точке D. Так как точка D получена в результате пересечения прямых **(1)** и **(2)**, то ее координаты удовлетворяют уравнениям этих прямых:

$x_1+2x_2=7$

$2x_1+x_2=8$

Решив систему уравнений, получим: $x_1 = 3$, $x_2 = 2$

Откуда найдем минимальное значение целевой функции:

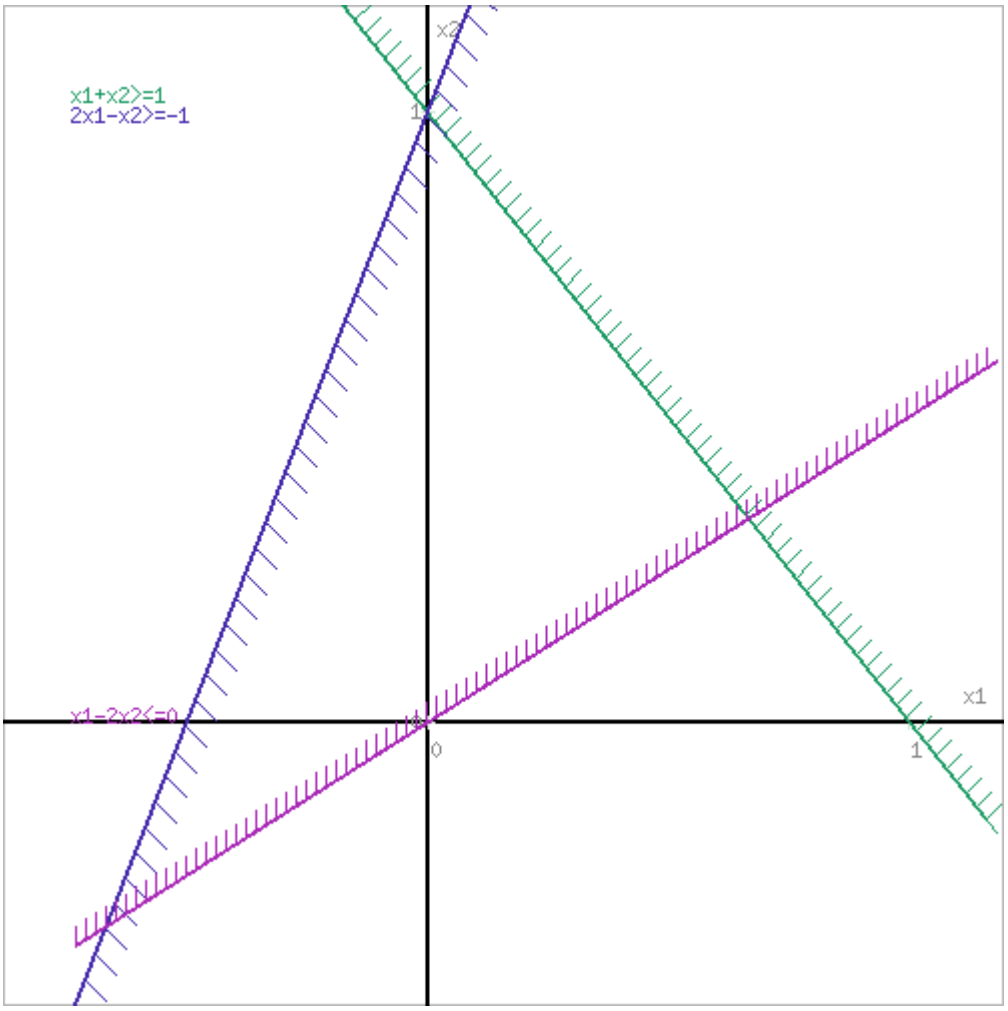
F(x) = -3*3 - 2*2 = -13

8. Используя графический метод, найти решение задачи линейного программирования:

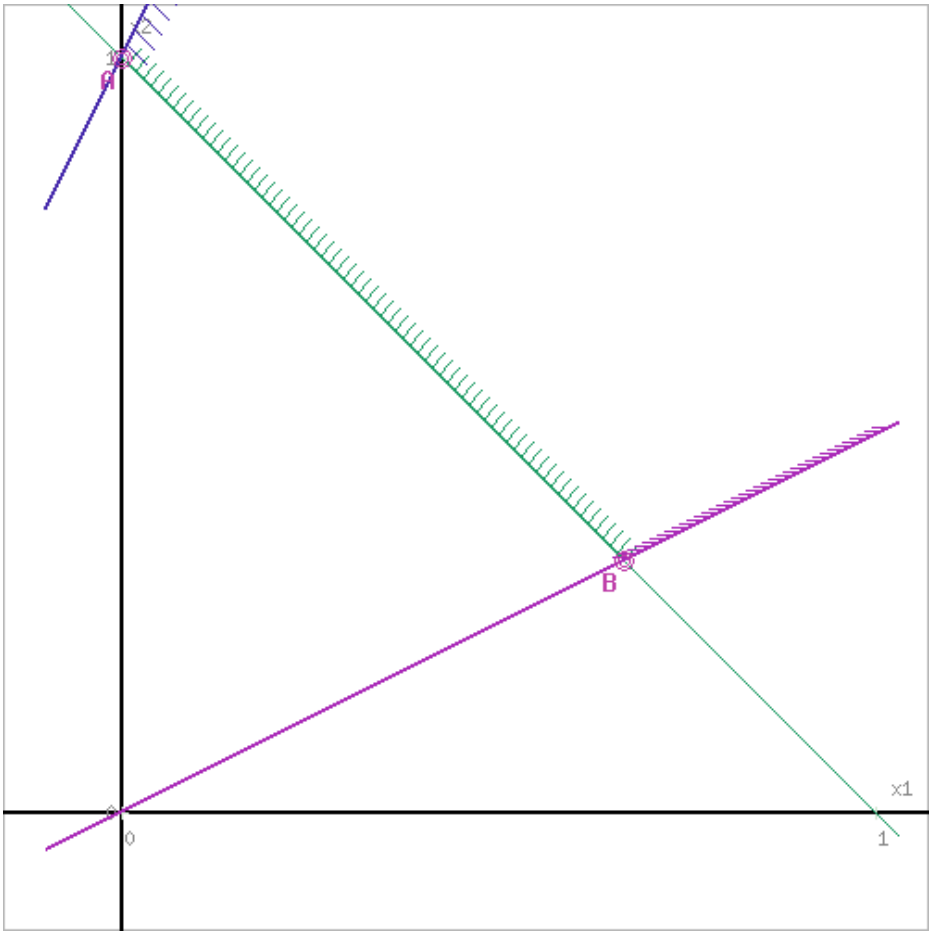
$$f(x) = -x_1 - 2x_2 \to \min,$$

$$\begin{cases} x_1 + x_2 \geq 1, \\ 2x_1 - x_2 \geq -1, \\ x_1 - 2x_2 \leq 0, \\ x_1 \geq 0, \quad x_2 \geq 0. \end{cases}$$

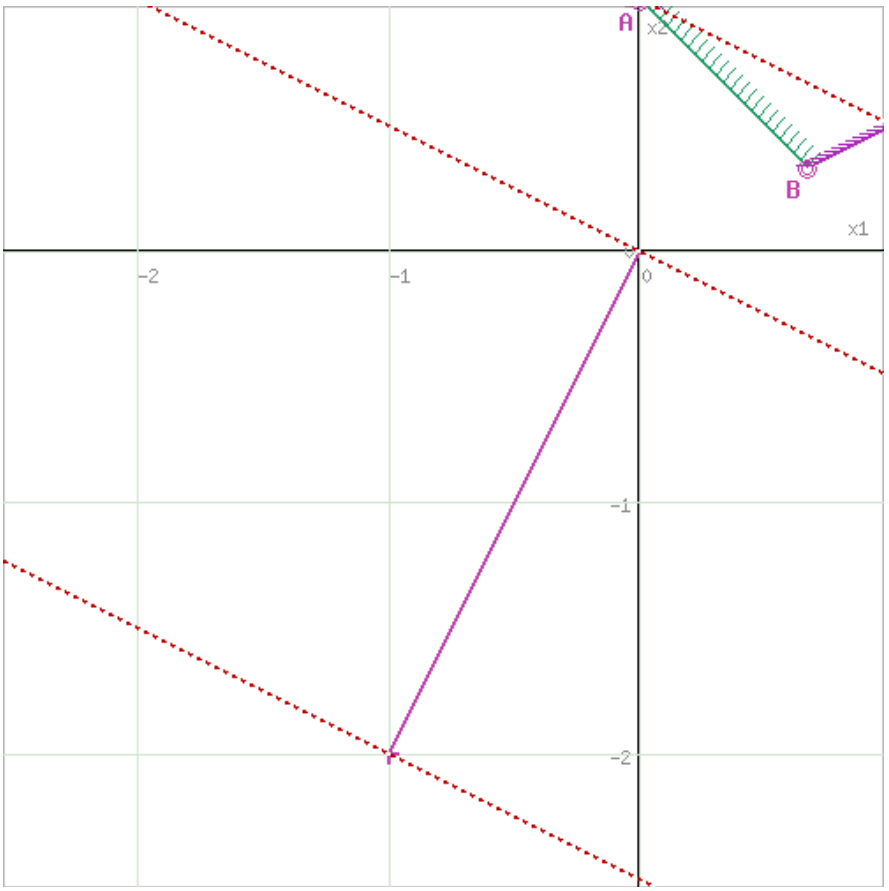
Построим область допустимых решений, т.е. решим графически систему неравенств. Для этого построим каждую прямую и определим полуплоскости, заданные неравенствами (полуплоскости обозначены штрихом).



Пересечением полуплоскостей будет являться область, координаты точек которого удовлетворяют условию неравенствам системы ограничений задачи. Обозначим границы области многоугольника решений.



Построим прямую, отвечающую значению функции $F = -x_1-2x_2 = 0$. Вектор-градиент, составленный из коэффициентов целевой функции, указывает направление максимизации F(X). Начало вектора – точка (0; 0), конец – точка (-1;-2). Будем двигать эту прямую параллельным образом. Поскольку нас интересует минимальное решение, поэтому двигаем прямую до первого касания обозначенной области. На графике эта прямая обозначена пунктирной линией.



Задача не имеет допустимых решений. ОДР представляет собой бесконечное множество (не ограничена).

9. Решить задачу симплекс-методом, используя x_0 в качестве начальной точки:

$$f(x) = -5x_1 + 4x_2 - x_3 - 3x_4 - 5x_5 \to \min,$$

$$\begin{cases} 3x_1 - x_2 + 2x_4 + x_5 = 5, \\ 2x_1 - 3x_2 + x_3 + 2x_4 + x_5 = 6, \\ 3x_1 - x_2 + x_3 + 3x_4 + 2x_5 = 9, \\ x_j \geq 0. \end{cases}$$

$$x_0 = (0, 0, 1, 2, 1).$$

Начальная симплекс-таблица:

Из-за того, что нам даны базовые точки, мы должны выразить их, то есть привести к единичному виду.

Используя исключение Гаусса:

Делим строку 1 на 2, а из второй и третьей строк вычитаем первую, умноженную на соответствующий элемент в четвёртом столбце.

Делим строку 2 на 1, а из первой и третьей строк вычитаем вторую, умноженную на соответствующий элемент в третьем столбце.

Делим строку 3 на 0.5, а из первой и второй строк вычитаем третью, умноженную на соответствующий элемент в пятом столбце.

C	**-5**	**4**	**-1**	**-3**	**-5**	**0**
Базис	x_1	x_2	x_3	x_4	x_5	**b**
x_4	2	-5	0	1	0	2
x_3	-1	-4	1	0	0	1
x_5	-1	9	0	0	1	1

Проверка плана на оптимальность и переход к более оптимальному решению:

Вычисляем дельты:

Вычисляем дельты:

$\Delta_1 = C_4 \cdot a_{11} + C_3 \cdot a_{21} + C_5 \cdot a_{31} - C_1 = -3 \cdot 2 + -1 \cdot (-1) + -5 \cdot (-1) - -5 = 5$

$\Delta_2 = C_4 \cdot a_{12} + C_3 \cdot a_{22} + C_5 \cdot a_{32} - C_2 = -3 \cdot (-5) + -1 \cdot (-4) + -5 \cdot 9 - 4 = -30$

$\Delta_3 = C_4 \cdot a_{13} + C_3 \cdot a_{23} + C_5 \cdot a_{33} - C_3 = -3 \cdot 0 + -1 \cdot 1 + -5 \cdot 0 - -1 = 0$

$\Delta_4 = C_4 \cdot a_{14} + C_3 \cdot a_{24} + C_5 \cdot a_{34} - C_4 = -3 \cdot 1 + -1 \cdot 0 + -5 \cdot 0 - -3 = 0$

$\Delta_5 = C_4 \cdot a_{15} + C_3 \cdot a_{25} + C_5 \cdot a_{35} - C_5 = -3 \cdot 0 + -1 \cdot 0 + -5 \cdot 1 - -5 = 0$

$\Delta_b = C_4 \cdot b_1 + C_3 \cdot b_2 + C_5 \cdot b_3 - C_6 = -3 \cdot 2 + -1 \cdot 1 + -5 \cdot 1 - 0 = -12$

Симплекс-таблица с дельтами:

C	**-5**	**4**	**-1**	**-3**	**-5**	**0**
Базис	x_1	x_2	x_3	x_4	x_5	**b**
x_4	2	-5	0	1	0	2
x_3	-1	-4	1	0	0	1
x_5	-1	9	0	0	1	1
Δ	5	-30	0	0	0	-12

Проверяем план на оптимальность: план не оптимален, так как Δ1 = 5 положительна.

Итерация 1:

Определяем *разрешающий столбец* - столбец, в котором находится максимальная дельта: 1, Δ_1: 5

Находим симплекс-отношения Q, путём деления коэффициентов b на соответствующие значения столбца 1

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

На пересечении найденных строки и столбца находится *разрешающий элемент*: 2

В качестве базисной переменной x_4 берём x_1.

Обновлённая таблица:

C	**-5**	**4**	**-1**	**-3**	**-5**	**0**	
Базис	x_1	x_2	x_3	x_4	x_5	**b**	**Q**
x_4	2	-5	0	1	0	2	2 / 2 = 1
x_3	-1	-4	1	0	0	1	-
x_5	-1	9	0	0	1	1	-
Δ	5	-30	0	0	0	-12	

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

Вычисляем новые дельты:

$\Delta_1 = C_1 \cdot a_{11} + C_3 \cdot a_{21} + C_5 \cdot a_{31} - C_1 = -5 \cdot 1 + -1 \cdot 0 + -5 \cdot 0 - -5 = 0$

$\Delta_2 = C_1 \cdot a_{12} + C_3 \cdot a_{22} + C_5 \cdot a_{32} - C_2 = -5 \cdot (-2.5) + -1 \cdot (-6.5) + -5 \cdot 6.5 - 4 = -17.5$

$\Delta_3 = C_1 \cdot a_{13} + C_3 \cdot a_{23} + C_5 \cdot a_{33} - C_3 = -5 \cdot 0 + -1 \cdot 1 + -5 \cdot 0 - -1 = 0$

$\Delta_4 = C_1 \cdot a_{14} + C_3 \cdot a_{24} + C_5 \cdot a_{34} - C_4 = -5 \cdot 0.5 + -1 \cdot 0.5 + -5 \cdot 0.5 - -3 = -2.5$

$\Delta_5 = C_1 \cdot a_{15} + C_3 \cdot a_{25} + C_5 \cdot a_{35} - C_5 = -5 \cdot 0 + -1 \cdot 0 + -5 \cdot 1 - -5 = 0$

$\Delta_b = C_1 \cdot b_1 + C_3 \cdot b_2 + C_5 \cdot b_3 - C_6 = -5 \cdot 1 + -1 \cdot 2 + -5 \cdot 2 - 0 = -17$

Обновлённая таблица:

C	**-5**	**4**	**-1**	**-3**	**-5**	**0**	
Базис	x_1	x_2	x_3	x_4	x_5	**b**	**Q**
x_4	2	-5	0	1	0	2	1
x_3	-1	-4	1	0	0	1	-
x_5	-1	9	0	0	1	1	-
Δ	0	-17.5	0	-2.5	0	-17	

Текущий план X: [1, 0, 2, 0, 2]

Целевая функция F: $-5 \cdot 1 + 4 \cdot 0 + -1 \cdot 2 + -3 \cdot 0 + -5 \cdot 2 = -17$

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

Ответ: $x_1 = 1$, $x_2 = 0$, $x_3 = 2$, $x_4 = 0$, $x_5 = 2$, $F = -17$

10. Найти наибольшее значение функций:

$$f(x) = -x_1 + 3x_2 \to \max,$$

$$\begin{cases} x_1 + 2x_2 \le 4, \\ x_1 - x_2 \ge 1, \\ x_1 + x_2 \le 8, \\ x_1 \ge 0, \quad x_2 \ge 0. \end{cases}$$

Формирование начального базиса:

Нужно привести все ограничения к каноническому виду — виду, в котором все условия задаются равенствами.

Меняем знаки у ограничений с ≥, путём умножения на -1:

$$\begin{cases} x_1 + 2x_2 \le 4, \\ -x_1 + x_2 \le -1, \\ x_1 + x_2 \le 8 \end{cases}$$

Для каждого ограничения с неравенством добавляем дополнительные переменные $x_3 .. x_5$.

Перепишем ограничения в каноническом виде:

$$\begin{cases} x_1 + 2x_2 + x_3 = 4, \\ -x_1 + x_2 + x_4 = -1, \\ x_1 + x_2 + x_5 = 8 \end{cases}$$

Ищем начальное базисное решение:

Ограничение 1 содержит неравенство, базисной будет добавленная дополнительная переменная x_3

Ограничение 2 содержит неравенство, базисной будет добавленная дополнительная переменная x_4

Ограничение 3 содержит неравенство, базисной будет добавленная дополнительная переменная x_5

Начальная симплекс-таблица:

C	-1	3	0	0	0	0
Базис	x_1	x_2	x_3	x_4	x_5	**b**
x_3	1	2	1	0	0	4
x_4	-1	1	0	1	0	-1
x_5	1	1	0	0	1	8

В столбце **b** присутствуют отрицательные значения.
Максимальное по модулю $|b|_{max} = |-1|$ находится в строке 2.
Максимальный по модулю элемент в строке $2 = -1$ находится в столбце 1.
В качестве базисной переменной x_4 берём x_1.

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

Обновлённая таблица:

C	-1	3	0	0	0	0
Базис	x_1	x_2	x_3	x_4	x_5	**b**
x_3	0	3	1	1	0	3
x_1	1	-1	0	-1	0	1
x_5	0	2	0	1	1	7

Проверка плана на оптимальность и переход к более оптимальному решению:
Вычисляем дельты:
$\Delta_1 = C_3 \cdot a_{11} + C_1 \cdot a_{21} + C_5 \cdot a_{31} - C_1 = 0 \cdot 0 + -1 \cdot 1 + 0 \cdot 0 - -1 = 0$
$\Delta_2 = C_3 \cdot a_{12} + C_1 \cdot a_{22} + C_5 \cdot a_{32} - C_2 = 0 \cdot 3 + -1 \cdot (-1) + 0 \cdot 2 - 3 = -2$
$\Delta_3 = C_3 \cdot a_{13} + C_1 \cdot a_{23} + C_5 \cdot a_{33} - C_3 = 0 \cdot 1 + -1 \cdot 0 + 0 \cdot 0 - 0 = 0$
$\Delta_4 = C_3 \cdot a_{14} + C_1 \cdot a_{24} + C_5 \cdot a_{34} - C_4 = 0 \cdot 1 + -1 \cdot (-1) + 0 \cdot 1 - 0 = 1$
$\Delta_5 = C_3 \cdot a_{15} + C_1 \cdot a_{25} + C_5 \cdot a_{35} - C_5 = 0 \cdot 0 + -1 \cdot 0 + 0 \cdot 1 - 0 = 0$
$\Delta_b = C_3 \cdot b_1 + C_1 \cdot b_2 + C_5 \cdot b_3 - C_6 = 0 \cdot 3 + -1 \cdot 1 + 0 \cdot 7 - 0 = -1$

Симплекс-таблица с дельтами:

C	-1	3	0	0	0	0
Базис	x_1	x_2	x_3	x_4	x_5	**b**
x_3	0	3	1	1	0	3
x_1	1	-1	0	-1	0	1
x_5	0	2	0	1	1	7
Δ	0	-2	0	1	0	-1

Проверяем план на оптимальность: план не оптимален, так как Δ2 = -2 отрицательна.

Итерация 1:

Определяем разрешающий столбец - столбец, в котором находится минимальная дельта: 2, Δ2: -2

Находим симплекс-отношения Q, путём деления коэффициентов b на соответствующие значения столбца 2

В найденном столбце ищем строку с наименьшим значением $Q: Q_{min} = 1$, строка 1.На пересечении найденных строки и столбца находится разрешающий элемент: 3

В качестве базисной переменной x_3 берём x_2.

Обновлённая таблица:

C	-1	3	0	0	0	0	
Базис	x_1	x_2	x_3	x_4	x_5	**b**	**Q**
x_2	0	3	1	1	0	3	3 / 3 = 1
x_1	1	-1	0	-1	0	1	-
x_5	0	2	0	1	1	7	7 / 2 = 3.5
Δ	0	-2	0	1	0	-1	

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

Вычисляем новые дельты:

$\Delta_1 = C_2 \cdot a_{11} + C_1 \cdot a_{21} + C_5 \cdot a_{31} - C_1 = 3 \cdot 0 + -1 \cdot 1 + 0 \cdot 0 - -1 = 0$

$\Delta_2 = C_2 \cdot a_{12} + C_1 \cdot a_{22} + C_5 \cdot a_{32} - C_2 = 3 \cdot 1 + -1 \cdot 0 + 0 \cdot 0 - 3 = 0$
$\Delta_3 = C_2 \cdot a_{13} + C_1 \cdot a_{23} + C_5 \cdot a_{33} - C_3 = 3 \cdot 0.3 + -1 \cdot 0.3 + 0 \cdot (-0.6) - 0 = 0.6$
$\Delta_4 = C_2 \cdot a_{14} + C_1 \cdot a_{24} + C_5 \cdot a_{34} - C_4 = 3 \cdot 0.3 + -1 \cdot (-0.6) + 0 \cdot 0.3 - 0 = 1.6$
$\Delta_5 = C_2 \cdot a_{15} + C_1 \cdot a_{25} + C_5 \cdot a_{35} - C_5 = 3 \cdot 0 + -1 \cdot 0 + 0 \cdot 1 - 0 = 0$
$\Delta_b = C_2 \cdot b_1 + C_1 \cdot b_2 + C_5 \cdot b_3 - C_6 = 3 \cdot 1 + -1 \cdot 2 + 0 \cdot 5 - 0 = 1$

Симплекс-таблица с обновлёнными дельтами:

C	**-1**	**3**	**0**	**0**	**0**	**0**	
Базис	x_1	x_2	x_3	x_4	x_5	**b**	**Q**
x_2	0	1	0.3	0.3	0	1	1
x_1	1	0	0.3	-0.6	0	2	-
x_5	0	0	-0.6	0.3	1	5	3.5
Δ	0	0	0.6	1.6	0	1	

Текущий план X: [2, 1, 0, 0, 5]
Целевая функция F: $-1 \cdot 2 + 3 \cdot 1 + 0 \cdot 0 + 0 \cdot 0 + 0 \cdot 5 = 1$
Проверяем план на оптимальность: отрицательные дельты отсутствуют, следовательно **план оптимален**.
Ответ: $x_1 = 2$, $x_2 = 1$, F =1