Функция-метод для ввода данных для решения задачи максимизации.

def input_d()
'''Выходные данные: список введенных значений, list ''' Функция-метод для обработки введенных данных.
def processing(s)
'''Входные данные: s - список введенных значений, list Выходные данные: кортеж обработанных данных, tuple'''

Функция-метод для создания матрицы симплекс-метода. # def to_tableau(c, A, b) '''Входные данные:

c - вектор коэффициентов функции максимизации, list A - матрица коэффициентов ограничений, list
b - вектор ограничений, list
Выходные данные: матрица симплекс-метода, list '''

Функция-метод для проверки матрицы симплекс-метода. # def can_be_improved(tableau) '''
Входные данные:

tableau - матрица симплекс метода, list

Выходные данные: содержит ли последняя строка с ограничениями положительные значения, bool '''

Функция-метод для нахождения базового значения в симлекс матрице. # def get_pivot_position(tableau)
''' Входные данные:
tableau - матрица симплекс метода, list

Выходные данные: строки и столбец координаты базового значения, tuple ''' Функция-метод для шага симплекс метода.
def pivot_step(tableau, pivot_position)

'''Входные данные:
tableau - матрица симплекс метода, list
pivot_position - координаты базового значения в матрице, tuple Выходные данные: новая матрица симплекс метода, list'''

Функция-метод для проверки столбца.

def is_basic(column)

'''Входные данные:

column - столбец матрицы симплекс метода, list
Выходные данные: является ли столбец базовым или нет, bool '''

Функция-метод для нахождения решения по матрице симплекс метода. # def get_solution(tableau)
'''Входные данные:
tableau - матрица симплекс метода, list

Выходные данные: решение симплекс метода, list '''

Функция-метод симплекс метода. # def simplex(c, A, b)
'''Входные данные:

c - вектор коэффициентов функции максимизации, list
A - матрица коэффициентов ограничений, list
b - вектор ограничений, list
Выходные данные: решение симплекс метода и матрица, tuple '''

Функция-метод для нахождения целочисленных решений методом ветвей и границ. # def branches_and_bound(c, A, b, last_sol)
'''Входные данные:

c - вектор коэффициентов функции максимизации, list
A - матрица коэффициентов ограничений, list
b - вектор ограничений, list
last_sol - решение на предыдущем шаге, list
Выходные данные: список списков решений по всем ветвям, list '''

Функция-метод для распаковки списка и списков решений. # def unwrap_list(mylist, result)

''' Входные данные:
mylist - список полученных решений, list
result - список распакованных решений, list
Выходные данные: список решений по всем ветвям, list '''

Функция-метод симплекс метода. # def gomori(c, A, b, f, list_f) '''Входные данные:

c - вектор коэффициентов функции максимизации, list
A - матрица коэффициентов ограничений, list
b - вектор ограничений, list
f - функция максимизации в аналитическом виде, sympy.expression list_f - список переменных, list

Выходные данные: целочисленное решение задачи методом Гомори. ''' Функция-метод для нахождения целочисленных решений методом ветвей и границ. # def find_good(c,A,b,f,list_f,last_sol=[0])

'''Входные данные:
c - вектор коэффициентов функции максимизации, list
A - матрица коэффициентов ограничений, list

b - вектор ограничений, list
f - функция максимизации в аналитическом виде, sympy.expression list_f - список переменных, list

last_sol - решение на предыдущем шаге, list
Выходные данные: решение методом ветвей и границ. '''

Функция-метод для объединения вызова функций для задачи метода Гомори
def all_f_gomori()
'''Выходные данные: целочисленное решение задачи методом Гомори. ''' Функция-метод для объединения вызова функций для задачи метода ветвей и границ # def all_f_branches_and_bound()

''' Выходные данные: целочисленное решение задачи методом ветвей и границ. '''