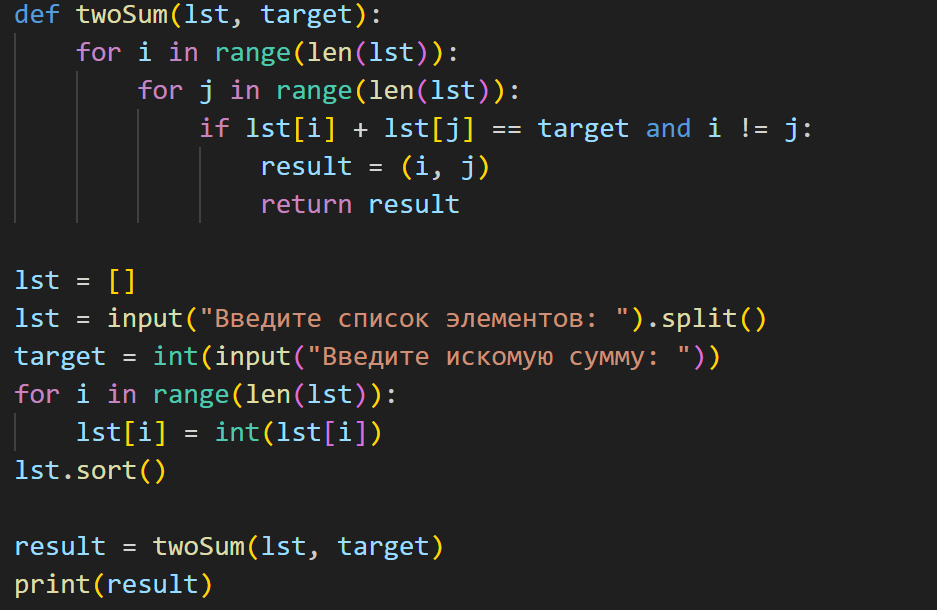
**Задача 1.**

****

**1. Назначение программы**

Программа реализует алгоритм поиска **двух чисел в списке**, сумма которых равна заданному значению (target). Результат возвращается в виде индексов этих чисел в исходном списке.

**2. Описание работы кода**

**2.1. Функция twoSum(lst, target)**

**Назначение:** Поиск пары чисел в списке lst, сумма которых равна target.

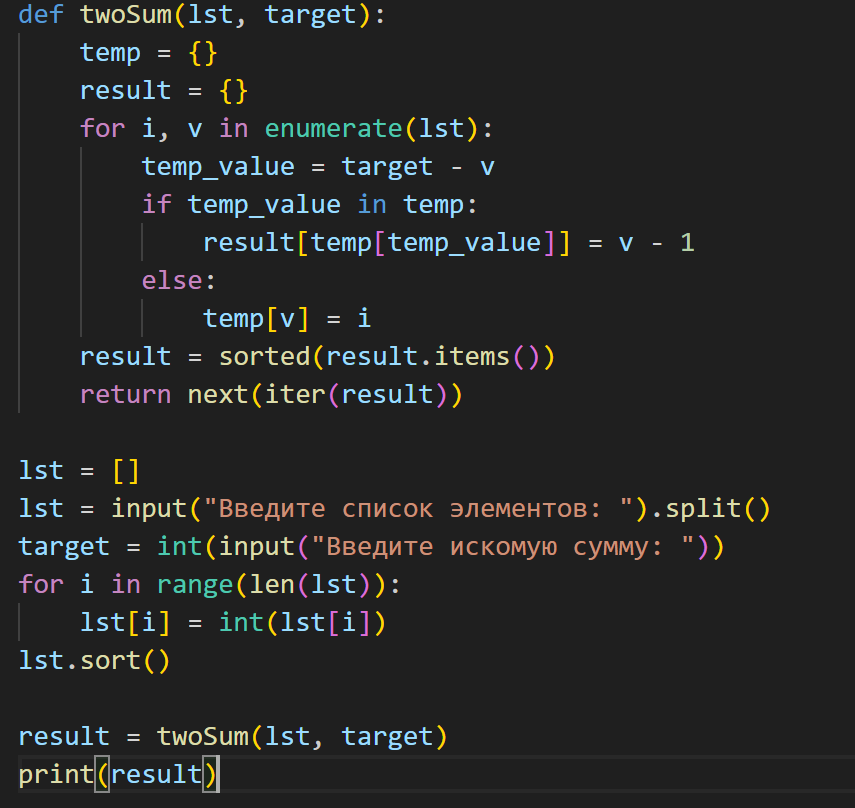
**Аргументы:**

* lst — список целых чисел.
* target — искомая сумма.

**Логика работы:**

1. Перебирает все возможные пары элементов списка с помощью двух вложенных циклов for.
2. Проверяет условия:
   * Сумма элементов пары равна target.
   * Индексы элементов различны (i != j).
3. Возвращает кортеж индексов (i, j) при нахождении подходящей пары.

**Задача 2.**



**1. Назначение программы**

Программа ищет **пару чисел в списке**, сумма которых равна заданному значению (target), и возвращает их индексы. В отличие от предыдущей версии, здесь используется оптимизированный подход с применением словаря для уменьшения вычислительной сложности.

**2. Описание работы кода**

**2.1. Функция twoSum(lst, target)**

**Назначение:** Поиск пары чисел в списке lst, сумма которых равна target, с использованием хеш-таблицы.

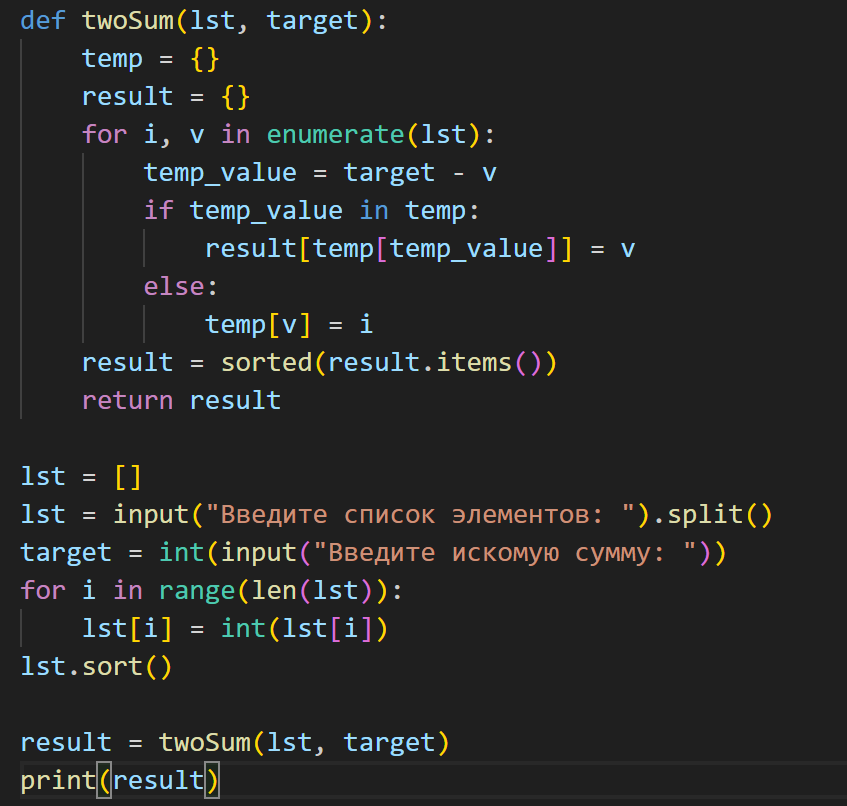
**Аргументы:**

* lst — список целых чисел.
* target — искомая сумма.

**Логика работы:**

1. Создаются два словаря: temp (для хранения значений и их индексов) и result (для сохранения найденных пар).
2. Для каждого элемента v в списке:
   * Вычисляется temp\_value = target - v.
   * Если temp\_value есть в temp, в result добавляется запись:
   * Иначе элемент v и его индекс сохраняются в temp.
3. Результат сортируется, и возвращается первая найденная пара.

**Задача 3.**

****

**1. Назначение программы**

Программа предназначена для поиска **пар чисел в списке**, сумма которых равна заданному значению (target). Результат возвращается в виде списка кортежей, содержащих индексы и значения элементов. Код демонстрирует попытку оптимизации с использованием хеш-таблицы.

**2. Описание работы кода**

**2.1. Функция twoSum(lst, target)**

**Назначение:** Поиск пар чисел в списке lst, сумма которых равна target.

**Аргументы:**

* lst — список целых чисел.
* target — искомая сумма.

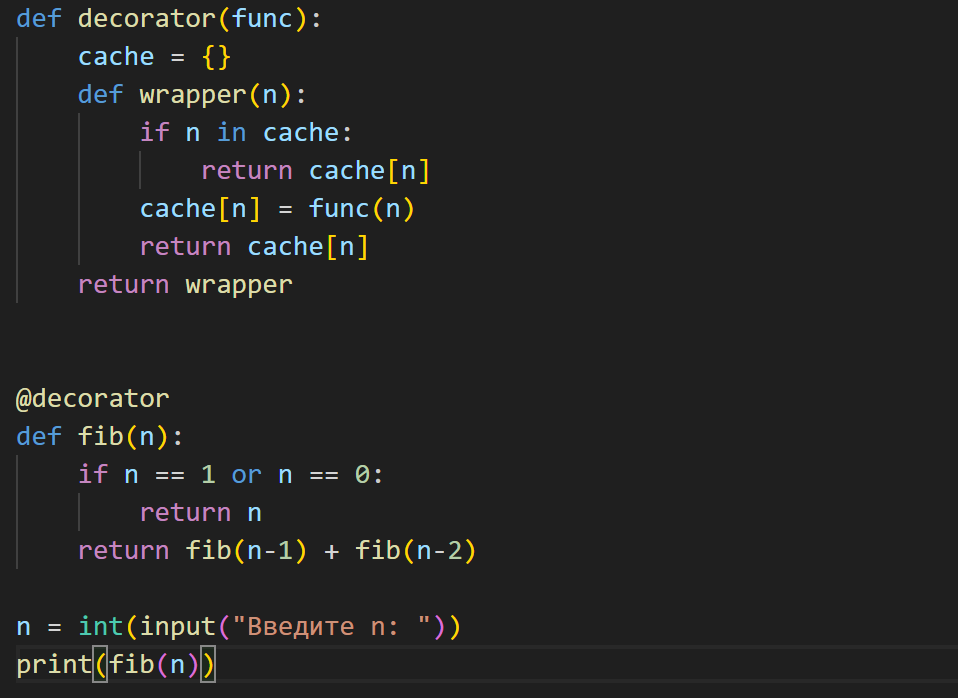
**Логика работы:**

1. Создаются два словаря:
   * temp — для хранения значений и их индексов.
   * result — для сохранения найденных пар.
2. Для каждого элемента v в списке:
   * Вычисляется temp\_value = target - v.
   * Если temp\_value есть в temp, в result добавляется запись: ключ: индекс первого числа, значение: второе число
   * Иначе элемент v и его индекс сохраняются в temp.
3. Результат сортируется и возвращается в виде списка кортежей (индекс, значение).

**2.2. Взаимодействие с пользователем**

1. Пользователь вводит список чисел и целевую сумму.
2. Список преобразуется в целые числа и **сортируется**, что изменяет исходные индексы элементов.
3. Вызывается функция twoSum, результат выводится на экран.

**Задача 4.**



**1. Назначение программы**

Программа вычисляет **n-ное число Фибоначчи** с использованием рекурсии и оптимизации через **кеширование**. Код демонстрирует:

* Применение декораторов для добавления функциональности (кеширование результатов).
* Эффективное устранение избыточных вычислений в рекурсивном алгоритме.

**2. Описание работы кода**

**2.1. Декоратор decorator**

**Назначение:** Добавляет кеширование результатов функции для избежания повторных вычислений.

**Логика работы:**

1. Создает словарь cache для хранения результатов (ключ — n, значение — результат fib(n)).
2. Функция wrapper(n) проверяет:
   * Если n есть в cache, возвращает значение из кеша.
   * Если n нет в cache, вычисляет fib(n), сохраняет результат в cache и возвращает его.

**2.2. Функция fib(n)**

**Назначение:** Рекурсивное вычисление n-ного числа Фибоначчи.

**Формула:**

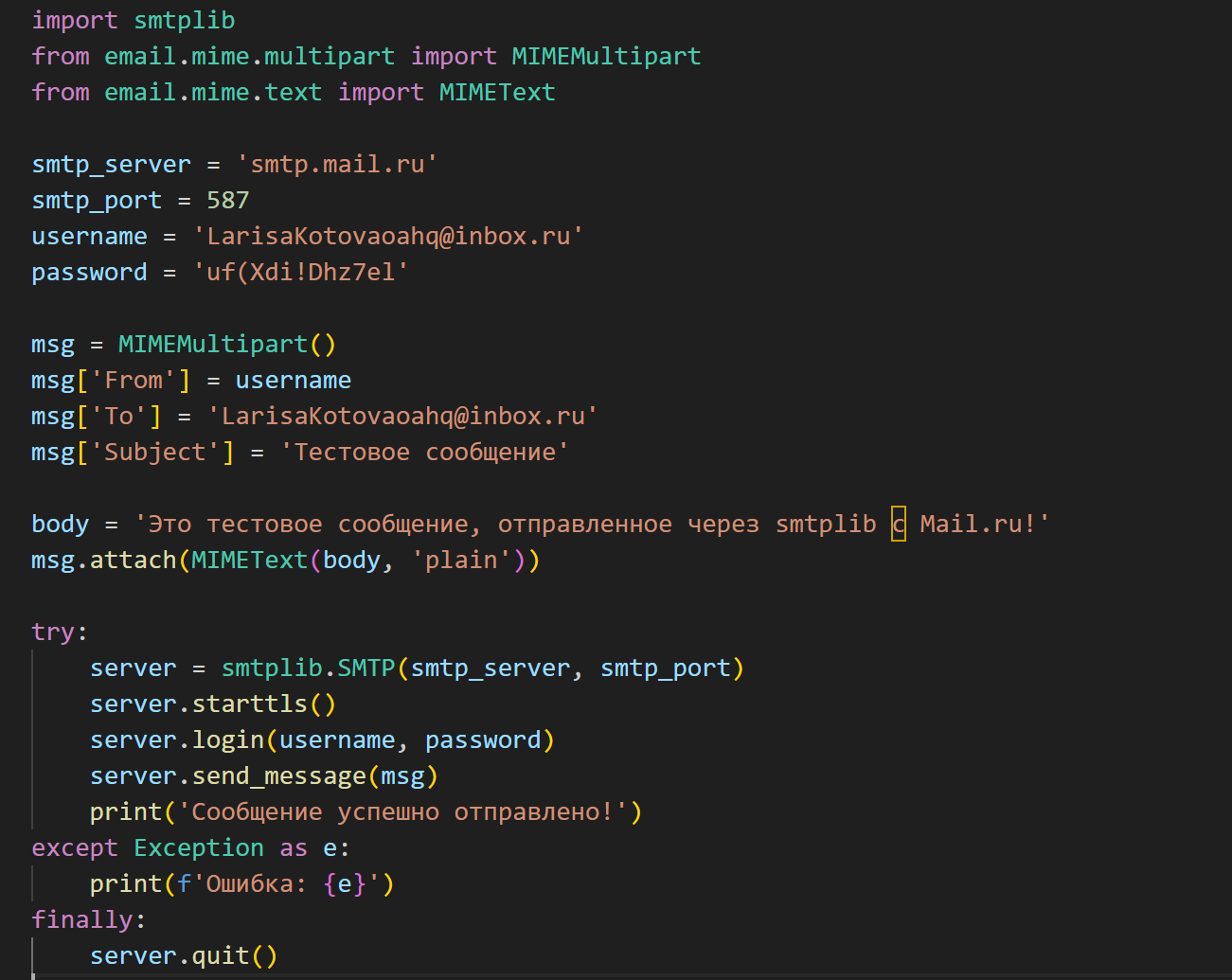
* fib(0) = 0, fib(1) = 1 (базовые случаи).
* fib(n) = fib(n-1) + fib(n-2) для n > 1.

**Особенность:**  
Без кеширования сложность алгоритма была бы экспоненциальной (O(2^n)). С кешированием сложность снижается до линейной (O(n)).

**2.3. Взаимодействие с пользователем**

1. Пользователь вводит целое число n.
2. Программа вызывает fib(n) и выводит результат.

**Задача 5.**



**1. Назначение программы**

Программа предназначена для **отправки электронных писем** через SMTP-сервер **Mail.ru**. Код демонстрирует:

* Настройку подключения к почтовому серверу.
* Создание и отправку текстового сообщения.
* Обработку ошибок при выполнении операций.

**2. Описание работы кода**

**2.1. Настройка SMTP-сервера**

* **Сервер:** smtp.mail.ru (порт 587).
* **Аутентификация:** Логин (username) и пароль (password) от аккаунта Mail.ru.
* **Протокол:** Используется STARTTLS для шифрования соединения.

**2.2. Создание сообщения**

1. **Объект MIMEMultipart:**
   * Поля:
     + From: Адрес отправителя.
     + To: Адрес получателя (в примере совпадает с отправителем).
     + Subject: Тема письма ("Тестовое сообщение").
   * **Тело письма:** Текстовое сообщение, добавленное через MIMEText.

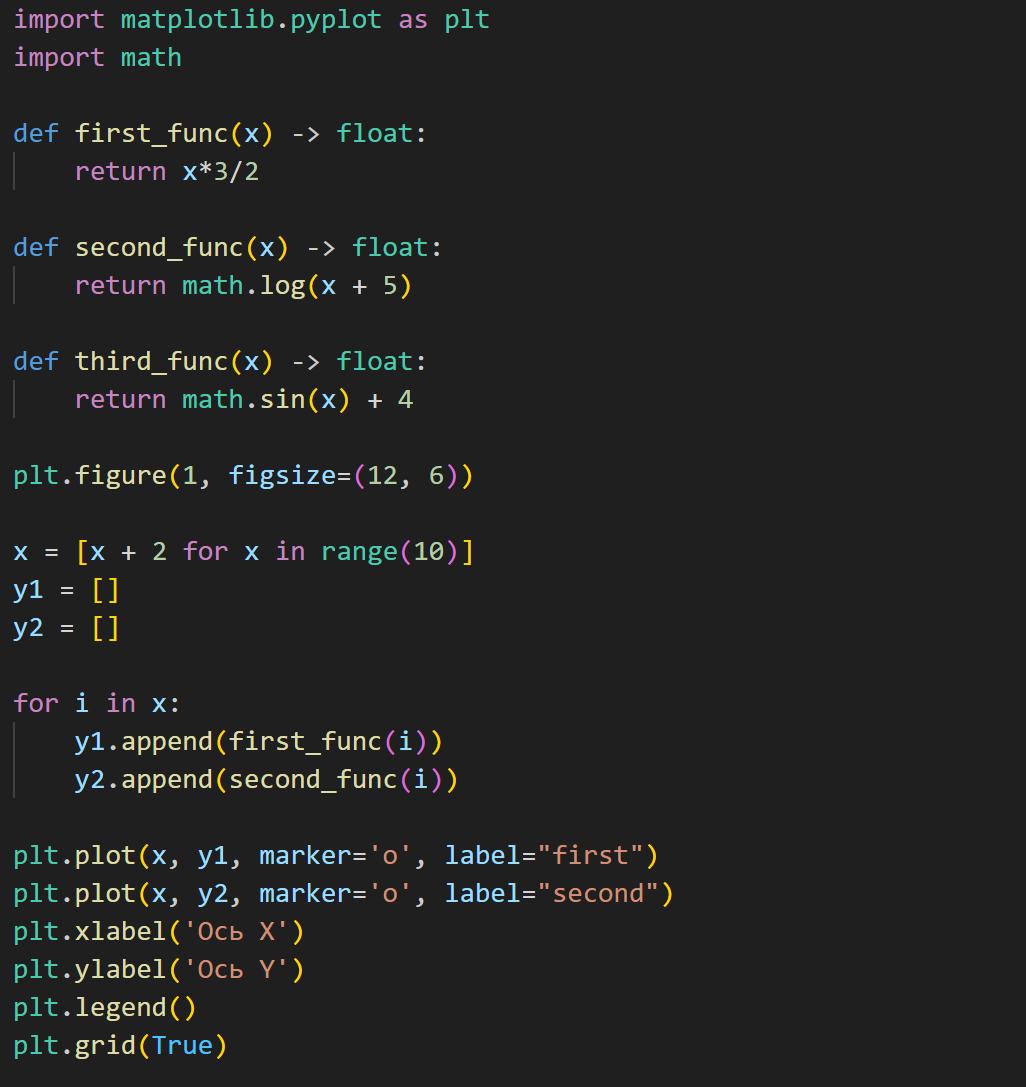
**2.3. Отправка сообщения**

1. Установка соединения с сервером (smtplib.SMTP).
2. Активация шифрования (starttls()).
3. Аутентификация (login()).
4. Отправка сообщения (send\_message()).
5. Закрытие соединения (quit()).

**2.4. Обработка ошибок**

* Блок try-except-finally перехватывает исключения (например, некорректные данные, проблемы с сетью) и выводит сообщение об ошибке.

**Задача 6.**





**1. Назначение программы**

Программа визуализирует **три математические функции** на двух отдельных графиках с использованием библиотеки matplotlib. Цель: демонстрация работы с графиками, настройкой их внешнего вида и отображением нескольких функций в разных областях.

**2. Описание работы кода**

**2.1. Определение функций**

1. **first\_func(x)**
   * Линейная функция: y=32x*y*=23​*x*.
2. **second\_func(x)**
   * Логарифмическая функция: y=ln⁡(x+5)*y*=ln(*x*+5).
3. **third\_func(x)**
   * Синусоида с смещением: y=sin⁡(x)+4*y*=sin(*x*)+4.

**2.2. Построение графиков**

**График 1 (Figure 1):**

* **Размер:** 12x6 дюймов.
* **Данные:**
  + x∈[2,11]*x*∈[2,11] (значения от 2 до 11).
  + y1*y*1: Результаты first\_func.
  + y2*y*2: Результаты second\_func.
* **Настройки:**
  + Линии с маркерами (marker='o').
  + Подписи осей (xlabel, ylabel).
  + Легенда (label), сетка (grid).

**График 2 (Figure 2):**

* **Данные:**
  + x∈[2,11]*x*∈[2,11].
  + y*y*: Результаты third\_func.
* **Настройки:**
  + Красная линия с маркерами (color='red').
  + Подписи осей, сетка.

**2.3. Визуализация**

* Команда plt.show() отображает оба графика в отдельных окнах.