**Практическая работа №7. Функции. Область видимости переменных**

Задание. Боев.В.М. ИСП.23.1А.

Выбрать задачи из файлов с вариантами в соответствии с таблицей.

Ссылка на варианты [ElenaKuzmina/PR7\_Function](https://github.com/ElenaKuzmina/PR7_Function)

Написать программный модуль, содержащий функции, реализующие решение задачи. Обязательно должна быть функция main(), содержащая вызовы остальных функций. В отчет включить для каждой задачи:

1. ФИО студента, группа
2. Постановка задачи
3. Описание переменных с указанием имени, типа, назначения, уровня видимости (в виде таблицы)
4. Словесный алгоритм решения задачи
5. Программный код модуля
6. Скриншоты ошибок с описанием ошибки и способом ее исправления
7. Скриншоты тестирования программного модуля задачи
8. Ссылка на свой репозиторий github с программным модулем

Таблица вариантов

|  |  |  |
| --- | --- | --- |
| Студент | Задача 1 | Задача 2 |
| [Боев Владислав](https://school.mosreg.ru/marks?school=2000002775450&group=2257826504610747540&student=2000002419400&tab=stats) | Вариант 1, задача 2 | Вариант 1, задача 3 |

Задача 1

**Формат ввода**

Вводится три строки чисел, разделённых пробелами.

**Формат вывода**

В одну строку через точку с запятой и пробел вывести числа, которые есть во второй и третьей строке, но которых нет в первой. Выводить без повторений в любом порядке.

Затем вывести сумму выбранных чисел.

**Пример 1**

| Ввод | Вывод |
| --- | --- |
| 6 8 11 17 19 9 13  15 17 7 8 5 10 9  5 1 13 18 12 7 17 | 5; 7  12 |

**Пример 2**

| Ввод | Вывод |
| --- | --- |
| 12 3 18 5 15 17 4  15 2 15 16 12 19 1 3  2 15 4 5 6 18 17 | 2  2 |

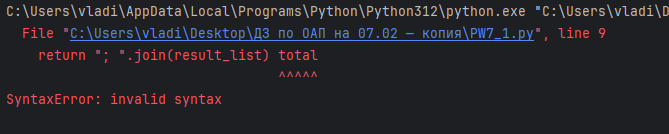
|  |  |  |  |
| --- | --- | --- | --- |
| Имя переменной | Тип | Назначение | Уровень видимости |
| Line1, line2, line3 | множество | Преобразуем строки в множество чисел | Локальный |
| Result\_set | множество | Находим числа, которые есть во второй и третьей строке, но которых нет в первой | Локальный |
| Result\_list | Список | Числа, которые есть во второй и третьей строке, но которых нет в первой, добавляются в список | Локальный |
| S1, s2, s3 | Строчный | Ввод трёх строк с числами | Локальный |
| Result | Строчный | Выводит числа в строковом формате, существующие во второй и третьей строке | Локальный |
| total | Числовой | Выводим сумму чисел, которые есть во второй и третьей строке | Локальный |

**Словесный алгоритм**

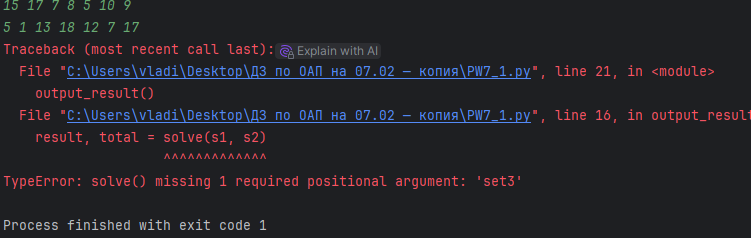
Мы вводим в трёх строках числа через пробел. Мы получим в рамках вывода только те числа, которые есть во второй и третьей строке, но нет в первой, а также сумму полученных таких чисел.

def solve(set1, set2, set3):  
 line1 = set((set1.split()))

line2 = set((set2.split()))  
 line3 = set((set3.split()))  
   
 result\_set = line2.intersection(line3) - line1  
 result\_list = list(result\_set)  
 total = sum(int(n) for n in result\_list)  
 return "; ".join(result\_list), total  
def output\_result():  
 s1 = input()  
 s2 = input()  
 s3 = input()  
 result, total = solve(s1, s2, s3)  
 print(result)  
 print(total)  
  
output\_result()

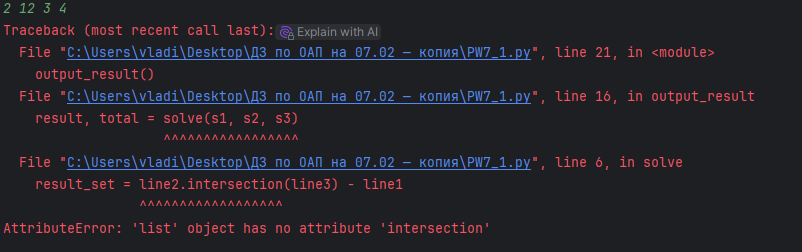


Появляется такая ошибка, потому что мы не поставили запятую после join(result\_list). Поэтому надо так: return “; “.join(result\_list), total



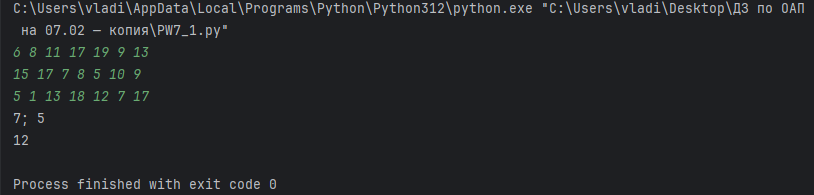
Потому что мы пропустили s3, поэтому после s2 ставим s3 через запятую и пробел

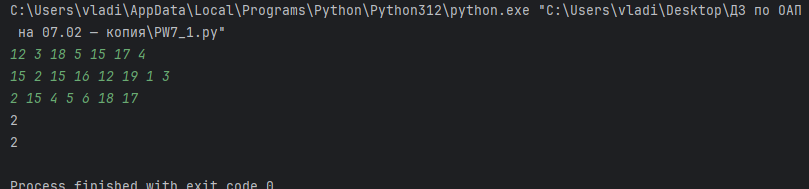
3.

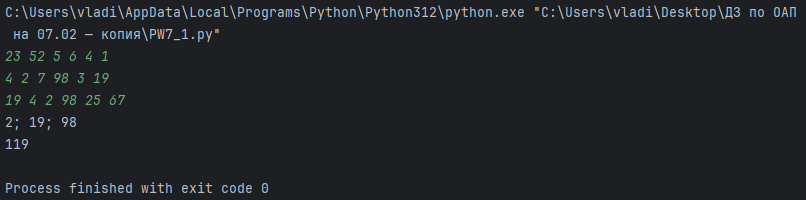


Такие ошибки возникают, потому что мы для line2 не присвоили set. Поэтому нужно для данной переменной присвоить set

**Тестирование**







Задание 2:

**Формат ввода**

Вводятся строки слов через дефис, окруженный пробелами, пока во введённой строке не появится слово HELLO.

**Формат вывода**

Из каждой строки в обратном алфавитном порядке без повторений через пробел вывести только те слова, у которых есть не больше 3 общих букв в любом регистре со словом **computer**. Строки, чьи индексы кратны 3 (нумерация индексов с 0), выводить прописными буквами.

**Пример 1**

| Ввод | Вывод |
| --- | --- |
| The - Sirius - Cybernetic - Corporations - onboard - computer  omnipresent - controlling - every - molecule - of - the - ship  immediately - awoke  HELLO - he - barked - cheerfully | THE SIRIUS ONBOARD  the ship of every  immediately awoke |

**Пример 2**

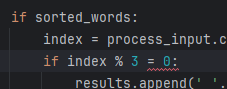
| Ввод | Вывод |
| --- | --- |
| The - computer - spat - out  A - narrow - strip - of - telegraph - tape  just - to - Register - The - fact  On - the - tape - was - printed  HELLO | THE SPAT OUT  tape strip of narrow A  to just fact The Register  WAS THE TAPE ON |

|  |  |  |  |
| --- | --- | --- | --- |
| Имя переменной | Тип | Назначение | Уровень видимости |
| target\_set | множество | Приводим каждую букву слова **computer** в нижний регистр | Локальный |
| Results | список | Здесь будут храниться слова, у которых есть не больше 3 общих букв в любом регистре со словом **computer** | Локальный |
| line | Список | Предназначена для ввода строки слов до тех пор, пока в процессе цикла не введем HELLO | Локальный |
| words | Список | Разделяем строку на слова | Локальный |
| Unique\_words | Множество | Сохраняем уникальные слова, которые подходят по условиям | Локальный |
| word\_set | Множество | В множество добавляют буквы каждого слова в нижнем регистре в рамках цикла words | Локальный |
| common\_letters | Число | Считаем количество общих букв | Локальный |
| sorted\_words | Список | Сортируем слова в обратном алфавитном порядке | Локальный |
| index | Число | Данная переменная равна числу счетчика | Локальный |
| process\_input.counter | Число | Инициализация счетчика | Локальный |

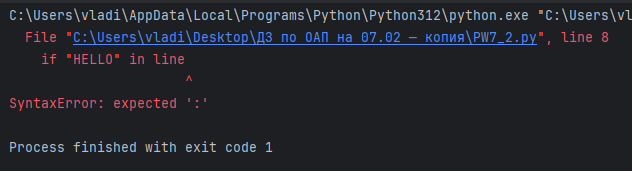
**Словесный алгоритм**

Вводим слова в любой строке через дефис, окруженный проблемами, до тех пор, пока мы не напишем HELLO. А затем в рамках вывода мы прописными буквами получим как раз строки, чьи индексы кратны 3.

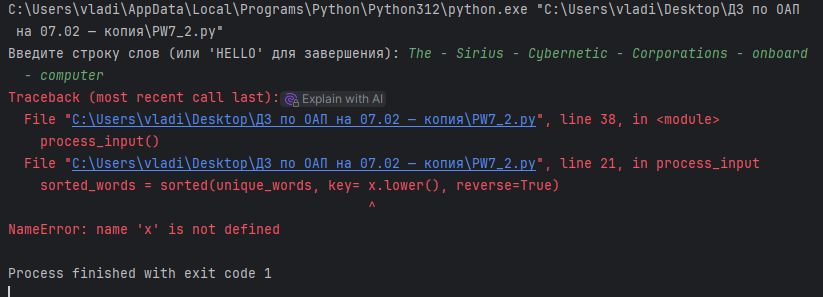
def process\_input():  
 target\_set = set("computer".lower())  
 results = []  
  
 while True:  
 line = input("Введите строку слов (или 'HELLO' для завершения): ")  
  
 if "HELLO" in line:  
 break  
  
 words = [word.strip() for word in line.split('-')]  
  
 unique\_words = set()  
  
 for word in words:  
 word\_set = set(word.lower())  
 common\_letters = len(word\_set.intersection(target\_set))  
 if common\_letters <= 3:  
 unique\_words.add(word.strip())  
  
 sorted\_words = sorted(unique\_words, key=lambda x: x.lower(), reverse=True)  
  
 if sorted\_words:  
 index = process\_input.counter  
 if index % 3 == 0:  
 results.append(' '.join(word.upper() for word in sorted\_words))  
 else:  
 results.append(' '.join(sorted\_words))  
  
 process\_input.counter += 1  
  
 for result in results:  
 print(result)  
  
  
process\_input.counter = 0  
  
process\_input()

\

Такая ошибка появляется, потому что мы должны написать два равно. Поэтому пишем ещё одно равно

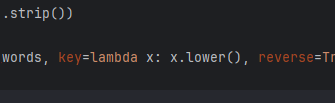


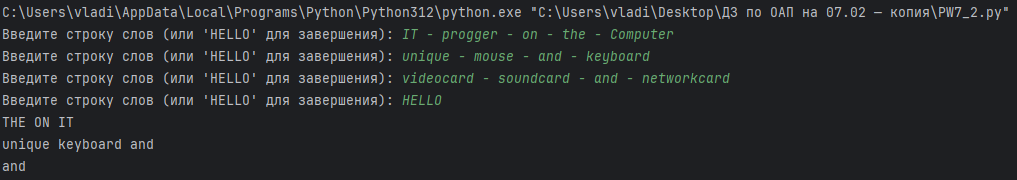
Такая ошибка появляется, потому что мы не поставили двоеточие после line. Поэтому ставим двоеточие.

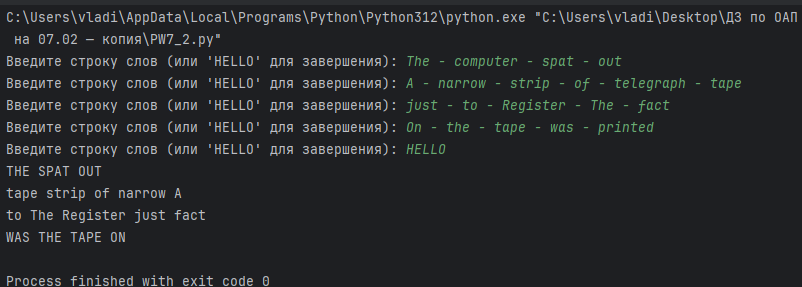


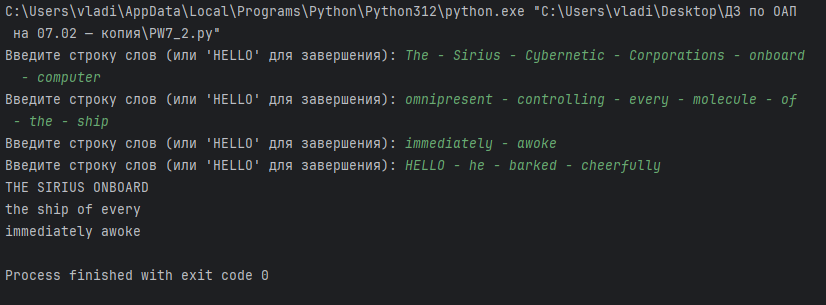
Такая ошибка произошла, потому что мы не поставили lambda x: после key=

Поэтому исправляем так:









***Ссылка на репозиторий в GitHub:*** https://github.com/vladislavboev2007/Homework-OAP-07.02.2025