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

**Феоктистов Глеб ИСП 23.1А**

Задание

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

|  |  |  |
| --- | --- | --- |
| [Феоктистов Глеб](https://school.mosreg.ru/marks?school=2000002775450&group=2257826504610747540&student=2000000060033&tab=stats" \t "_blank) | Вариант 7, задача 2 | Вариант 7, задача 3 |

**Задача 2**

**Формат ввода**

Вводятся три строки слов, записанных через пробел.

**Формат вывода**

Через плюс без повторений в произвольном порядке вывести все слова, которые встречались хотя бы в одной строке.

Затем вывести самое длинное из выбранных слов. Если таких несколько, самое маленькое при лексикографическом сравнении.

**Пример 1**

| Ввод | Вывод |
| --- | --- |
| all the doors  of our ship  have a light cheerful disposition | all+ship+have+doors+the+disposition+a+our+light+cheerful+of  disposition |

**Пример 2**

| Ввод | Вывод |
| --- | --- |
| oh ah hm hma  hma hoooray ohh  ah hma hmmmmma | ohh+hmmmmma+hma+hoooray+hm+ah+oh  hmmmmma |

**Программный код:**

def process\_input():  
 global unique\_words  
 unique\_words = set()  
  
 for \_ in range(3):  
 line = input("Введите строку: ")  
 words = line.split()  
 unique\_words.update(words)  
  
  
def longest\_word():  
 longest\_word = ""  
 for word in unique\_words:  
 if (len(word) > len(longest\_word)) or (len(word) == len(longest\_word) and word < longest\_word):  
 longest\_word = word  
 return longest\_word  
  
  
process\_input()  
print("+".join(unique\_words))  
print(longest\_word())

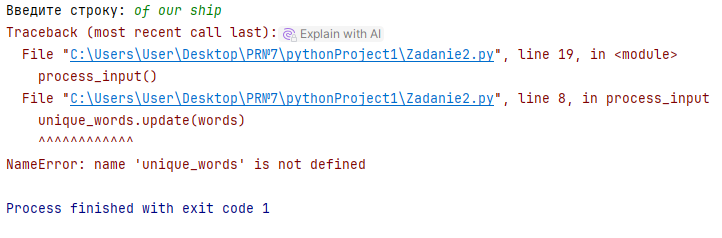
**Словесный алгоритм:**

Этот код принимает три строки от пользователя, находит все уникальные слова из этих строк, выводит их через знак "+" и находит самое длинное уникальное слово, упорядочивая слова лексикографически (по алфавиту) при одинаковой длине.

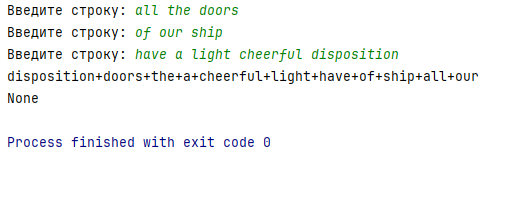
**Описание переменных:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя переменной** | **Тип** | **Назначение** | **Уровень видимости** |
| Unique\_words | set | Содержит все уникальные слова, полученные из всех строк | Глобальный |
| Line | str | Введённая пользователем строка | Локальный |
| Words | list | Список слов, полученных из строки line после разделения по пробелам | Локальный |
| \_ | int | Используется как переменная-заполнитель, так как значение не используется | Локальный |
| Longest\_word | str | Содержит самое длинное слово из unique\_words или слово с наименьшим лексикографическим порядком, если есть слова одинаковой длины. | Локальный |
| Word | str | Текущее слово при проходе по множеству unique\_words | Локальный |

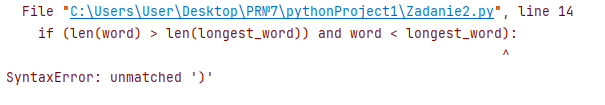
**Ошибки:**



Имя 'unique\_words' не определено. Здесь нужно добавить переменную set.

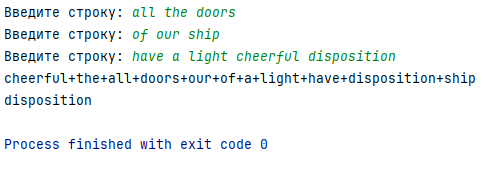


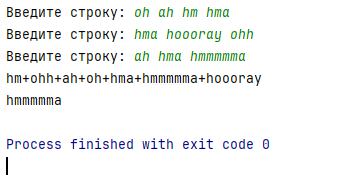
Пишут None, потому что не возвращают значение самого длинного слова(longest\_word). Нужно в вызове функции её обязательно вернуть!



Синтаксическая ошибка в сравнении переменных.

**Тестирование:**





**Задача 3**

**Формат ввода**

Вводятся строки чисел, записанных через #, пока не появится число 999 в строке.

**Формат вывода**

Из каждой строки через пробел без повторений вывести числа, в которых нет цифр из первого в строке числа. Выводить в возрастающем порядке, в каждой второй, начиная с первой, в конце вывести сумму выбранных в этой строке чисел.

Если чисел не нашлось, вывести пустую строку, сумму не выводить.

**Пример 1**

| Ввод | Вывод |
| --- | --- |
| 331#270#217#334#272#54#240  310#390#143#75#217#162#158  236#239#171#108#999#121#91 | 54 240 270 272 836  75 |

**Пример 2**

| Ввод | Вывод |
| --- | --- |
| 114#258#24#238#170#31#127  216#68#229#6#401#198#152  172#334#374#243#394#332#333  21#247#22#999#63#4#109 | 238 258 496  333 334 394 1061 |

**Программный код:**

def solve():  
 line\_num = 0  
 while True:  
 line = input()  
 numbers = line.split('#')  
  
 if '999' in numbers:  
 numbers = numbers[:numbers.index('999')]  
  
 if not numbers:  
 break  
  
 first\_num = numbers[0]  
 forbidden\_digits = set(first\_num)  
  
 valid\_numbers = []  
 for num\_str in numbers[1:]:  
 valid = True  
 for digit in num\_str:  
 if digit in forbidden\_digits:  
 valid = False  
 break  
 if valid:  
 valid\_numbers.append(int(num\_str))  
  
 valid\_numbers = sorted(list(set(valid\_numbers)))  
  
 if valid\_numbers:  
 print(\*valid\_numbers, end="")  
 if line\_num % 2 == 0:  
 print(" {}".format(sum(valid\_numbers)))  
 else:  
 print()  
 else:  
 print()  
  
 line\_num += 1  
  
solve()

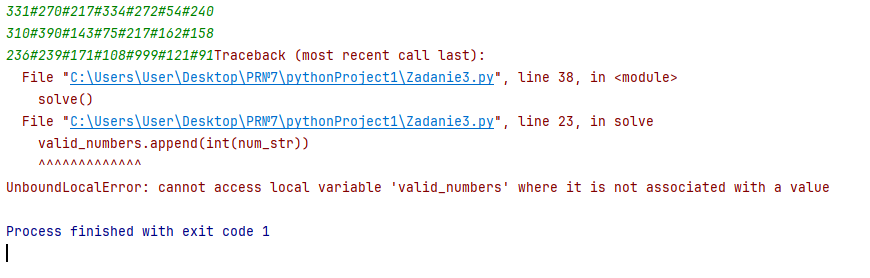
**Словесный алгоритм:**

Код обрабатывает строки чисел, разделенных символом '#'. Он фильтрует числа, основываясь на цифрах, присутствующих в первом числе строки и удаляет дубликаты. Отсортированные числа выводятся на экран. Каждая вторая строка выводит сумму отсортированных чисел после самих чисел. Строка завершается, только если ввели пустую строку или строка содержит '999'.

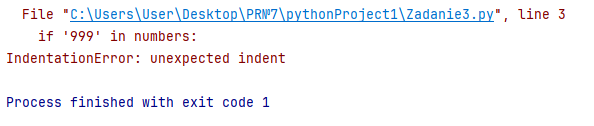
**Описание переменных:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя переменной** | **Тип** | **Назначение** | **Уровень видимости** |
| Line\_num | int | Отслеживает номер строки | Локальный |
| Line | str | Сохраняет входную строку | Локальный |
| Numbers | str | Список строк, извлеченных из входной строки, разделенных символом «#» | Локальный |
| First\_num | str | Первая числовая строка из списка | Локальный |
| Forbidden\_digits | set | Набор запрещенных цифр, извлеченных из первого числа | Локальный |
| Valid\_numbers | list | Список допустимых чисел, извлеченных из ввода | Локальный |
| Num\_str | str | Временная переменная, содержащая числовую строку из списка | Локальный |
| Valid | bool | Флаг, указывающий, является ли число допустимым | Локальный |
| Digit | str | Временная переменная, содержащая цифру из числовой строки | Локальный |

**Ошибки:**



Не создан список допустимых чисел. Нужно создать пустой список.



Не была создана функция для инициализации переменной и цикла while. Нужно добавить def и имя функции.

**Тестирование:**

