МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯРОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ

Отчет о лабораторной работе №2.3 по дисциплине основы программной инженерии

Выполнила:
Лобанова Елизавета Евгеньевна,
2 курс, группа ПИЖ-б-о-20-1

Проверил:
Доцент кафедры прикладной математики и компьютерной безопасности,
Воронкин Р.А.

Отчет защищен с оценкой_____Дата защиты_____

Ставрополь, 2021 г.

Ход работы

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

if __name__ == '__main__':
    s = input("Введите предложение: ")
    r = s.replace(' ', '_')
    print(f"Предложение после замены: {r}")
```

Пример 1

```
Введите предложение: работа примера номер 1
Предложение после замены: работа_примера_номер_1

Process finished with exit code 0
```

Результат 1 программы (1)

```
Введите предложение: папапапа арааоаоа нр г це
Предложение после замены: папапапа_арааоаоа_нр_г_це

Process finished with exit code 0
```

Результат 1 программы (2)

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

if __name__ == '__main__':
    word = input("Введите слово: ")
    idx = len(word) // 2
    if len(word) % 2 == 1:
        # Длина слова нечетная.
        r = word[:idx] + word[idx+1:]
    else:
        # Длина слова четная.
        r = word[:idx-1] + word[idx+1:]
    print(r)
```

Пример 2

```
Введите слово: линейка
линйка

Process finished with exit code 0
```

Результат 2 программы (1)

```
Введите слово: яблоко
ябко

Process finished with exit code 0
```

Результат 2 программы (2)

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import sys

if __name__ == '__main__':
    s = input("Введите предложение: ")
    n = int(input("Введите длину: "))
    # Проверить требуюемую длину.
    if len(s) >= n:
        print(
            "Заданная длина должна быть больше длины предложения",
            file=sys.stderr
        )
        exit(1)
    # Разделить предложение на слова.
    words = s.split(' ')
    # Проверить количество слов в предложении.
```

```python
    if len(words) < 2:
        print(
            "Предложение должно содержать несколько слов",
            file=sys.stderr
        )
        exit(1)
    # Количество пробелов для добавления.
    delta = n
    for word in words:
        delta -= len(word)
    # Количество пробелов на каждое слово.
    w, r = delta // (len(words) - 1), delta % (len(words) - 1)
    # Сформировать список для хранения слов и пробелов.
    lst = []
    # Пронумеровать все слова в списке и перебрать их.
    for i, word in enumerate(words):
        lst.append(word)
```

```
    # Если слово не является последним, добавить пробелы.
        if i < len(words) - 1:
            # Определить количество пробелов.
            width = w
            if r > 0:
                width += 1
                r -= 1
        # Добавить заданное количество пробелов в список.
            if width > 0:
                lst.append(' ' * width)

    # Вывести новое предложение, объединив все элементы списка lst.
    print(''.join(lst))
```

Пример 3

```
Введите предложение: вводим пример предложения
Введите длину: 44
вводим          пример          предложения

Process finished with exit code 0
```

Результат 3 программы (1)

```
Введите предложение: повар
Введите длину: 31
Предложение должно содержать несколько слов

Process finished with exit code 1
```

Результат 3 программы (2)

```
Введите предложение: мышь бежит
Введите длину: 4
Заданная длина должна быть больше длины предложения

Process finished with exit code 1
```

Результат 3 программы (3)

Вариант 12

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

if __name__ == '__main__':
    s = input("Введите предложение: ")
    n = s.count('нн')
    print(f"В предложение нн повторяется {n} раз")
```

Индивидуальная программа 1

```
Введите предложение: Мы посетили картинную галерею
В предложение нн повторяется 1 раз

Process finished with exit code 0
```

Результат работы индивидуальной программы 1 (1)

```
Введите предложение: нншгнпав ннрпав нн нн нн
В предложение нн повторяется 5 раз

Process finished with exit code 0
```

Результат работы индивидуальной программы 1 (2)

```python
# -*- coding: utf-8 -*-

if __name__ == '__main__':
    s = input("Введите предложение: ")
    n = s.find(',')
    p = s.find(',', n+1, )
    if n > 0 and p > 0:
        s = s[n+1: p]
    else:
        s = s[n+1:]
    print (s)
```

Индивидуальная программа 2

```
Введите предложение: За окном зима, а не весна
 а не весна

Process finished with exit code 0
```

Результат работы индивидуальной программы 2 (1)

```
Введите предложение: прол, павуп пе пква, пршцз
 павуп пе пква

Process finished with exit code 0
```

Результат работы индивидуальной программы 2 (2)

```
Введите предложение: cfghjkl, poiuy fgui, ;iu, hkjh, oiug
 poiuy fgui

Process finished with exit code 0
```

Результат работы индивидуальной программы 2 (3)

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

if __name__ == '__main__':
    word1 = "прроцесор"
    word2 = 'теекстовыйфайл'
    word3 = 'програма и аллгоритм'
    word4 = 'процесор и паммять'

    word1 = word1.replace('р', '', 1)
    word1 = word1[:6] + 'с' + word1[6:]
    word2 = word2.replace('е', '', 1)
    word2 = word2[:9] + ' ' + word2[9:]
    word3 = word3.replace('л', '', 1)
    word3 = word3[:6] + 'м' + word3[6:]
    word4 = word4.replace('м', '', 1)
    word4 = word4[:5] + 'с' + word4[5:]
    print (f"Исправленные фразы: \n- {word1};\n- {word2};"
           f"\n- {word3};\n- {word4}.")
```

Индивидуальная программа 3

```
Исправленные фразы:
- процессор;
- текстовый файл;
- программа и алгоритм;
- процессор и память.

Process finished with exit code 0
```

Результат работы индивидуальной программы 3

Ответы на вопросы:

1 Строки в Python - упорядоченные последовательности символов, используемые для хранения и представления текстовой информации, поэтому с помощью строк можно работать со всем, что может быть представлено в текстовой форме.

2. Строки в апострофах и в кавычках, экранированные последовательности - служебные символы, "Сырые" строки, строки в тройных апострофах или кавычках.

3. Сложение, умножение, оператор принадлежности. Строковых функций в Python много, вот некоторые из них: chr() – Преобразует целое число в символ ord() – Преобразует символ в целое число len() – Возвращает длину строки str() – Изменяет тип объекта на string

4. В Python строки являются упорядоченными последовательностями символьных данных и могут быть проиндексированы. Доступ к отдельным символам в строке можно получить, указав имя строки, за которым следует число в квадратных скобках []. Индексация строк начинается с нуля: у первого символа индекс 0, следующего 1 и так далее. Индекс последнего символа в python — ‘‘длина строки минус один’’.

5. Если s это строка, выражение формы s[m:n] возвращает часть s , начинающуюся с позиции m , и до позиции n , но не включая позицию. Если пропустить первый индекс, срез начинается с начала строки. Аналогично, если опустить второй индекс s[n:], срез длится от первого индекса до конца строки.

6. Более легкое представление в памяти.

7. s.istitle()

8. if s1 in s2

9. s.find().

10. len(s)

11. s.count().

12. f-строки упрощают форматирование строк. Пример: print(f’’ This is {name}, he is {age} years old”)

13. string.find([, [,]])

14. 'Hello, {}!'.format('Vasya')

15. string.isdigit()

16. 'foo.bar.baz.qux'.rsplit(sep='.') – пример разделения

17. string.islower()

18. s[0].isupper()

19. С точки зрения математической операции нельзя, можно лишь только вывести из без разделения друг от друга.

20. s = s[::-1]

21. ‘-‘.join[]

22. Верхний – s.upper(), нижний – s.lower()

23. s[0].upper() s[len(s)-1].upper()

24. s.isupper()

25. В случае, если надо сохранить символы конца стркои.

26. s.replace(“что менять”, “на что менять”)

27. string.endswith([, [,]]) – заканчивается, string.startswith([, [,]]) – начинается.

28. s.isspace()

29. Будет получена копия строки, состоящая из 3 исходных.

30. s.title()

31. string.partition() делит строку на основе разделителя. s.partition() отделяет от s подстроку длиной от начала до первого вхождения . Возвращаемое значение представляет собой кортеж из трех частей: Часть s до Разделитель Часть s после

32. Индекс последнего вхождения подстроки в строку.