

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии  
Департамент цифровых, робототехнических систем и электроники

**ОТЧЕТ**  
**ПО ЛАБОРАТОРНОЙ РАБОТЕ №6**  
**дисциплины**  
**«Основы кроссплатформенного программирования»**  
**Вариант 4**

Выполнил:  
Борцов Богдан Михайлович  
2 курс, группа ИТС-б-о-23-1,  
11.03.02 «Инфокоммуникационные  
технологии и системы связи», очная  
форма обучения

---

(подпись)

Проверил:  
Доцент департамента цифровых,  
робототехнических систем и  
электроники Воронкин Р.А.

---

(подпись)

Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_

Ставрополь, 2024 г.

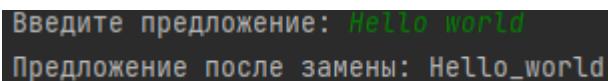
## Тема: Работа со строками в языке Python

**Цель:** приобретение навыков по работе со строками при написании программ с помощью языка программирования Python версии 3.x.

### Порядок выполнения работы:

1. Изучил теоретический материал.
2. Приступил к выполнению заданий.
3. Приступил к выполнению примеров.

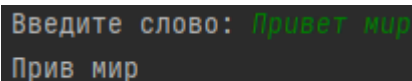
```
if __name__ == '__main__':  
    s = input("Введите предложение: ")  
    r = s.replace(' ', '_')  
    print("Предложение после замены:", r)
```



```
Введите предложение: Hello world  
Предложение после замены: Hello_world
```

Рисунок 1. Пример 1.

```
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)
```



```
Введите слово: Привет мир  
Прив мир
```

Рисунок 2. Пример 2.

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

```

if __name__ == "__main__":
    s = input("Введите предложение: ")
    n = int(input("Введите длину: "))

    # Проверить требуемую 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.
    if len(words) < 2:
        print("Предложение должно содержать несколько слов", file=sys.stderr)
        exit(1)

    # Количество пробелов для добавления.
    delta = n - len(s)

    for word in words:
        delta -= len(word)

    # Количество пробелов на каждое слово.
    w, r = divmod(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.

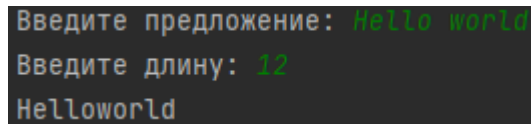
```

```

if width > 0:
    lst.append(' ' * width)

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

```



```

Введите предложение: Hello world
Введите длину: 12
Helloworld

```

Рисунок 3. Пример 3.

4. Зафиксировал изменения в репозитории.
5. Привел скриншоты результатов каждой из программы.
6. Приступил к выполнению индивидуального задания.

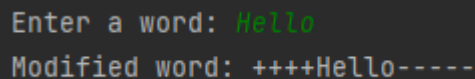
```

# Ввод слова
word = input("Enter a word: ")

# Добавление символов
modified_word = "++++" + word + "-----"

# Вывод результата
print(f"Modified word: {modified_word}")

```



```

Enter a word: Hello
Modified word: ++++Hello-----

```

Рисунок 4. Задание 1

```

# Ввод последовательности символов
sequence = input("Enter a sequence of characters: ")

# Определение количества одинаковых символов в начале
first_char = sequence[0] # Первый символ
count = 0

for char in sequence:
    if char == first_char:

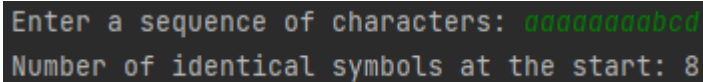
```

```

        count += 1
    else:
        break # Прекращаем подсчёт при первом отличном символе

# Вывод результата
print(f'Number of identical symbols at the start: {count}')

```



```

Enter a sequence of characters: aaaaaaaaaabcd
Number of identical symbols at the start: 8

```

Рисунок 5. Задание 2

```

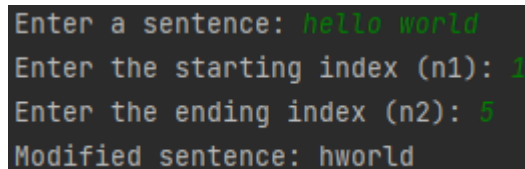
# Ввод предложения
sentence = input("Enter a sentence: ")

# Ввод индексов n1 и n2
n1 = int(input("Enter the starting index (n1): "))
n2 = int(input("Enter the ending index (n2): "))

# Удаление символов с n1-го по n2-й (включительно)
result = sentence[:n1] + sentence[n2 + 1:]

# Вывод результата
print(f'Modified sentence: {result}')

```



```

Enter a sentence: hello world
Enter the starting index (n1): 1
Enter the ending index (n2): 5
Modified sentence: hworld

```

Рисунок 6. Задание 3

7. Зафиксировал изменения в репозитории.

## **Ответы на контрольные вопросы:**

### **1. Что такое строки в языке Python?**

Строки в Python — строки в языке Python это последовательности символов, используемые для хранения текстовой информации. Они могут содержать буквы, цифры, пробелы и специальные символы.

### **2. Какие существуют способы задания строковых литералов в языке Python?**

Строки можно задавать с помощью одинарных, двойных, тройных одинарных и тройных двойных кавычек.

### **3. Какие операции и функции существуют для строк?**

Операции включают конкатенацию, повторение, а также функции, такие как `upper`, `lower`, `replace`, `split`.

### **4. Как осуществляется индексирование строк?**

Индексирование строк осуществляется с помощью квадратных скобок. Индексы начинаются с 0 для первого символа, -1 для последнего и тд.

### **5. Как осуществляется работа со срезами для строк?**

Срезы позволяют извлекать подстроки. Синтаксис: `s` - start - начальный индекс, а `end` - конечный индекс.

### **6. Почему строки Python относятся к неизменяемому типу данных?**

Строки являются неизменяемыми, потому что после создания их содержимое нельзя изменить. Любые операции, которые изменяют строку, создают новую строку.

### **7. Как проверить то, что каждое слово в строке начинается с заглавной буквы?**

`istitle`, `s.istitle`.

### **8. Как проверить строку на вхождение в неё другой строки?**

Оператор `in`: `substring in s`.

### **9. Как найти индекс первого вхождения подстроки в строку?**

Через `find`: `s.find`.

**10. Как подсчитать количество символов в строке?**

Через функцию len: len(s).

**11. Как подсчитать то, сколько раз определённый символ встречается в строке?**

Через метод count.

**12. Что такое f-строки и как ими пользоваться?**

Форматированные строки позволяют вставлять выражения в строку.

**13. Как найти подстроку в заданной части строки?**

Через метод find(start, end)

**14. Как вставить содержимое переменной в строку, воспользовавшись методом format?**

Через format: "Пример, {}".format(name).

**15. Как узнать о том, что в строке содержатся только цифры?**

Через isdigit

**16. Как разделить строку по заданному символу?**

Через метод split(,).

**17. Как проверить строку на то, что она составлена только из строчных букв?**

Через метод islower: s.islower.

**18. Как проверить то, что строка начинается со строчной буквы?**

Через islower для первого символа.

**19. Можно ли в Python прибавить целое число к строке?**

Нет. Строки и числа нельзя складывать напрямую.

**20. Как «перевернуть» строку?**

Через срез

**21. Как объединить список строк в одну строку, элементы которой разделены дефисами?**

через join

**22. Как привести всю строку к верхнему или нижнему регистру?**

через методы upper и lower

**23. Как преобразовать первый и последний символы строки к верхнему регистру?**

Через upper

**24. Как проверить строку на то, что она составлена только из прописных букв?**

Через метод isupper

**25. В какой ситуации вы воспользовались бы методом splitlines()?**

Метод splitlines() полезен для разделения строки на строки по символам новой строки.

**26. Как в заданной строке заменить на что-либо все вхождения некоей подстроки?**

Через метод replace.

**27. Как проверить то, что строка начинается с заданной последовательности символов, или заканчивается заданной последовательностью символов?**

Через методы startswith() и endswith()

**28. Как узнать о том, что строка включает в себя только пробелы?**

Используйте метод isspace: s.isspace.

**29. Что случится, если умножить некую строку на 3?**

Строка будет повторена трижды.

**30. Как привести к верхнему регистру первый символ каждого слова в строке?**

Через метод title: s.title.

**31. Как пользоваться методом partition()?**

Метод partition() разделяет строку на три части: до, разделитель и после.

**32. В каких ситуациях пользуются методом rfind()?**

Метод rfind() используется для поиска последнего вхождения подстроки в строке



**Вывод:** в ходе работы исследованы задачи обработки строк, текста и чисел в Python. Реализованы программы с использованием методов строк, конструкций ветвления и циклов. Рассмотрены алгоритмы для выполнения операций со строками, включая удаление, замену и перемещение символов, а также обработки текста: подсчёт символов, поиск подстрок, определение числовых характеристик. Решены задачи, связанные с арифметическими операциями, анализом текста и числами, включая нахождение суммы, произведения и других характеристик. Работа позволила закрепить навыки работы со строками, числами и текстовыми данными, а также разработку эффективных алгоритмов для их обработки.