

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»

Выполнил:
Бакулин Вадим Романович
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.

Github: <https://github.com/zepteloid/LR6>

Порядок выполнения работы:

1. Изучил теоретический материал.
2. Приступил к выполнению заданий.
3. Был создан и клонирован репозиторий
4. В ходе выполнения лабораторной работы были проработаны

примеры:

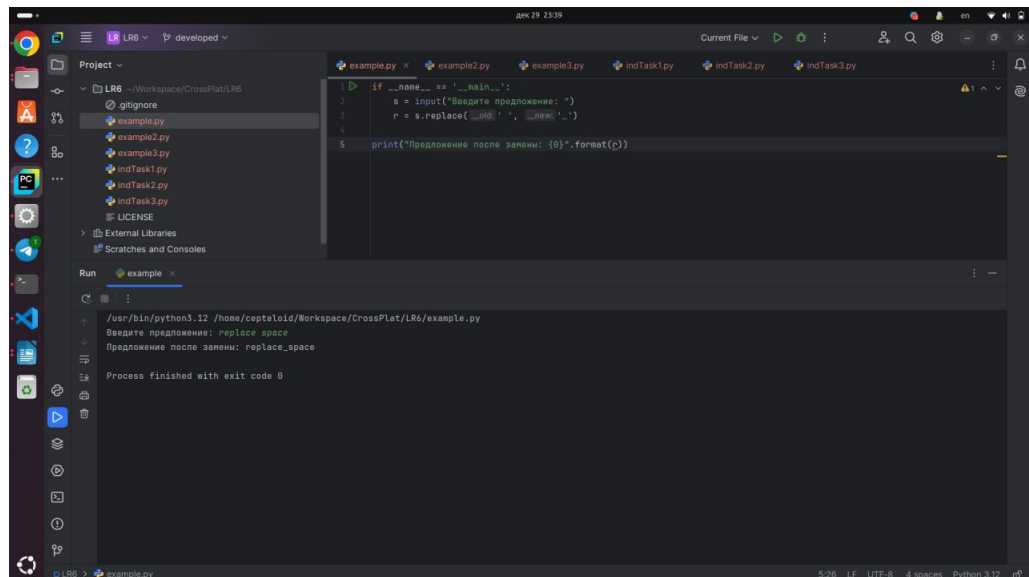


Рисунок 1. Код и выполнение программы первого примера

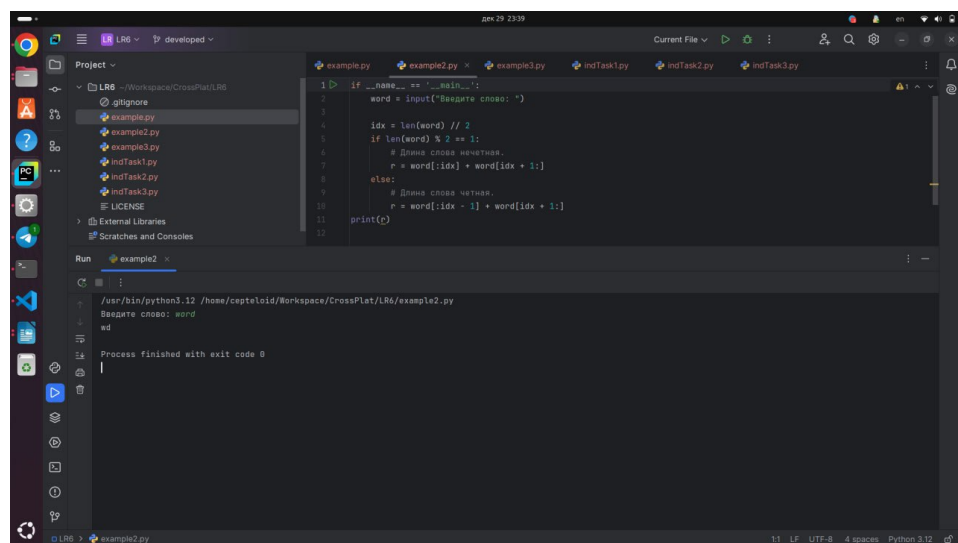


Рисунок 2. Код и выполнение программы второго примера

```
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
    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 = []
    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

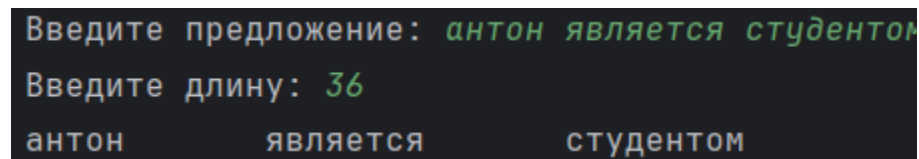
        r -= 1

    # 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: антон является студентом
Введите длину: 36
антон        является        студентом

```

Рисунок 3. Выполнение программы третьего примера

Были выполнены индивидуальные задания:

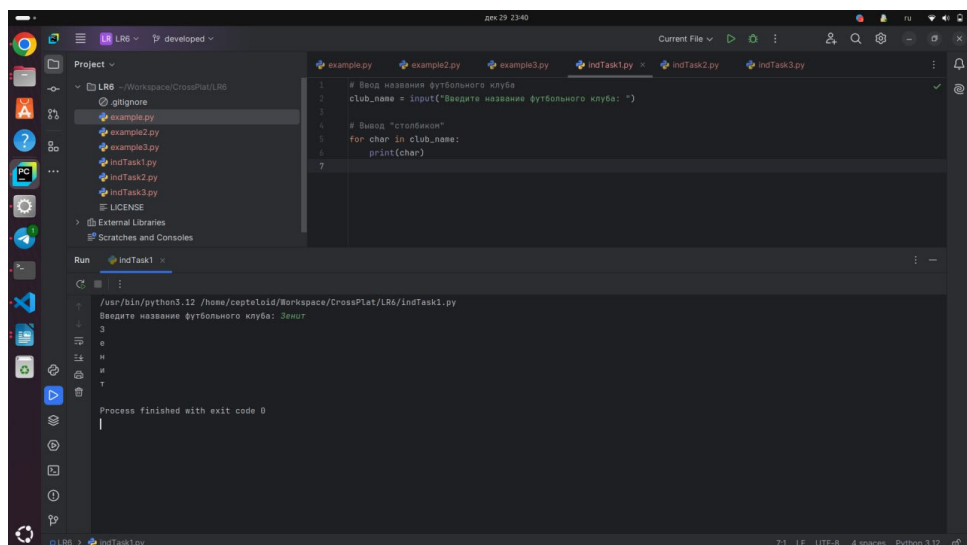


Рисунок 4. Код и результат выполнения программы первого индивидуального задания

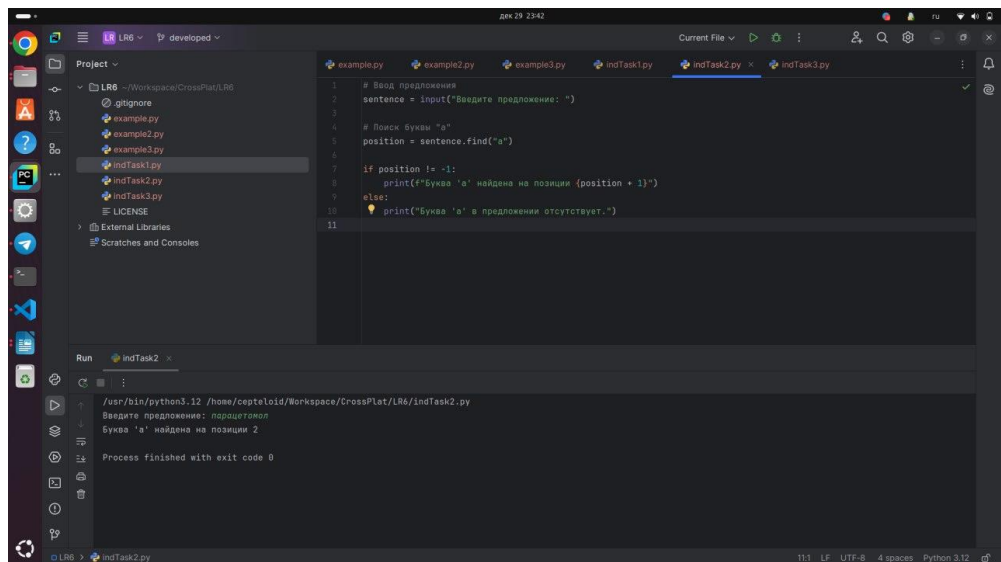


Рисунок 5. Код и результат выполнения программы второго индивидуального задания

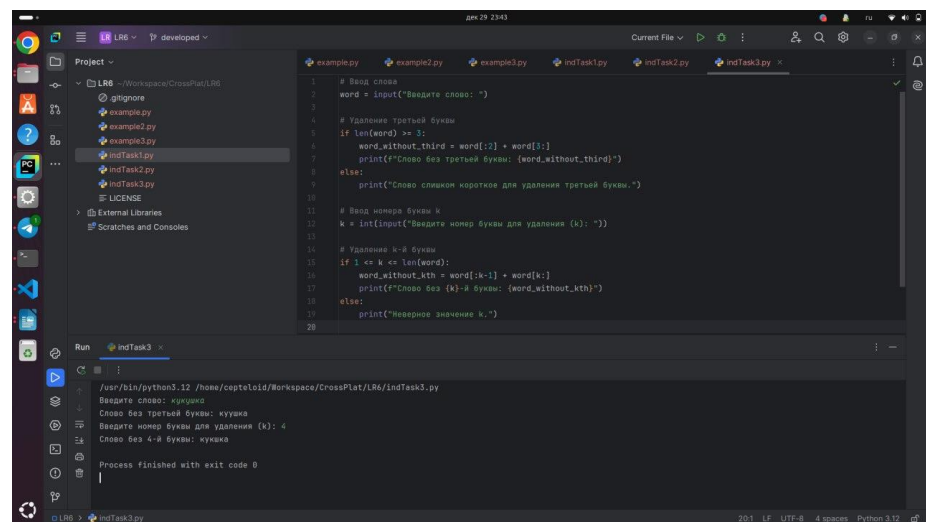


Рисунок 6. Код и результат выполнения программы третьего индивидуального задания

В ходе выполнения лабораторной работы были проиндексированы файлы, закодированы и отправлены на удаленный сервер.

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?

Это можно осуществить с помощью `find`: `s.find`.

10. Как подсчитать количество символов в строке?

Это можно осуществить с помощью функции `len`: `len(s)`.

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

Это можно осуществить с помощью метода `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?

Это можно осуществить с помощью метод `find(start, end)`

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

Это можно осуществить с помощью `format`: "Пример, {}".format(name).

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

Это можно осуществить с помощью `isdigit`

16. Как разделить строку по заданному символу?

Это можно осуществить с помощью метод `split(,)`.

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

Это можно осуществить с помощью метод `islower`: `s.islower`.

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

Это можно осуществить с помощью `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?

Это можно осуществить с помощью срез

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

Это можно осуществить с помощью `join`

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

Это можно осуществить с помощью методы `upper` и `lower`

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

Это можно осуществить с помощью `upper`

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

Это можно осуществить с помощью 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?

Это можно осуществить с помощью метода `replace`.

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

Это можно осуществить с помощью методов `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?

Это можно осуществить с помощью метода `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 выполнения лабораторной работы было изучено то, что такое строки и как с ними работать.