

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

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЁТ

по лабораторной работе №2.3

Дисциплина: «Программирование на Python»
Тема: «Работа со строками в языке Python»

Выполнил:
Епифанов Алексей Александрович
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и
вычислительная техника»,
направленность (профиль)
«Программное обеспечение средств
вычислительной
техники и автоматизированных систем
», очная форма обучения

(подпись)

Руководитель практики:
Воронкин Роман Александрович

(подпись)

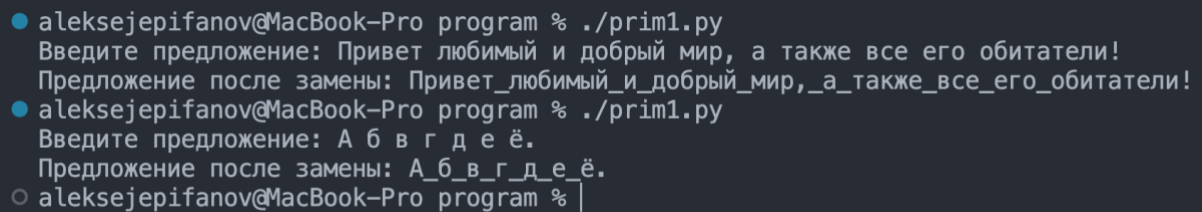
Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

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

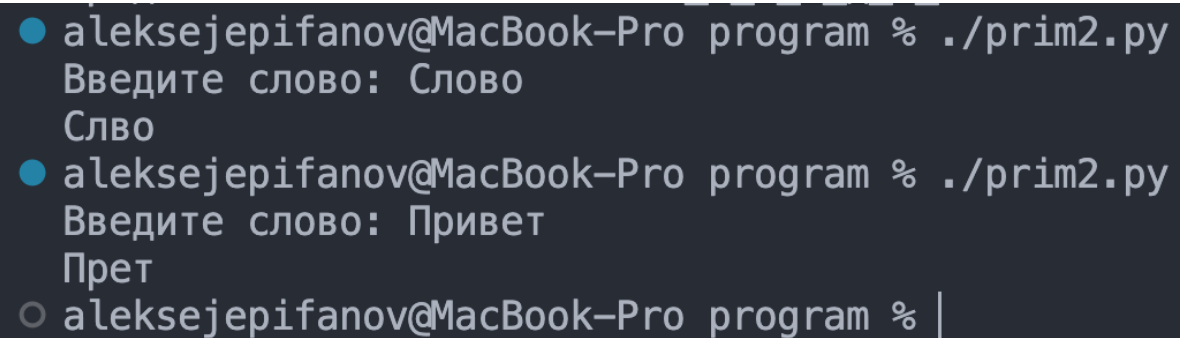
Порядок выполнения работы:

1. Создал новый репозиторий, клонировал его, в нем создал ветку developer и перешел на нее.
2. Проработал примеры лабораторной работы:



```
● aleksejerifanov@MacBook-Pro program % ./prim1.py
Введите предложение: Привет любимый и добрый мир, а также все его обитатели!
Предложение после замены: Привет_любимый_и_добрый_мир,_а_также_все_его_обитатели!
● aleksejerifanov@MacBook-Pro program % ./prim1.py
Введите предложение: А б в г д е ё.
Предложение после замены: А_б_в_г_д_е_ё.
○ aleksejerifanov@MacBook-Pro program % |
```

Рисунок 1. Несколько запусков программы примера 1



```
● aleksejerifanov@MacBook-Pro program % ./prim2.py
Введите слово: Слово
Слво
● aleksejerifanov@MacBook-Pro program % ./prim2.py
Введите слово: Привет
Прет
○ aleksejerifanov@MacBook-Pro program % |
```

Рисунок 2. Несколько запусков программы примера 2

```

● aleksejerifanov@MacBook-Pro program % ./prim3.py
Введите предложение: Тестовое предложение для примера 3.
Введите длину: 40
Тестовое предложение для примера 3.
⊗ aleksejerifanov@MacBook-Pro program % ./prim3.py
Введите предложение: Тестовое предложение для примера 3.
Введите длину: 35
Заданная длина должна быть больше длины предложения
● aleksejerifanov@MacBook-Pro program % ./prim3.py
Введите предложение: Тестовое предложение для примера 3.
Введите длину: 36
Тестовое предложение для примера 3.
⊗ aleksejerifanov@MacBook-Pro program % ./prim3.py
Введите предложение: Привет.
Введите длину: 4
Заданная длина должна быть больше длины предложения
⊗ aleksejerifanov@MacBook-Pro program % ./prim3.py
Введите предложение: Привет.
Введите длину: 10
Предложение должно содержать несколько слов
○ aleksejerifanov@MacBook-Pro program % |

```

Рисунок 3. Несколько запусков программы примера 3

3. Выполнил индивидуальное задание 1 вариант 10: Дано предложение. Вывести все буквы м и н в нем.

```

#!/usr/bin/env python3
# -*- coding: utf-8 -*-
if __name__ == '__main__':
    sentence = input("Введите предложение на русском языке: ")
    letters_m_n = []
    for letter in sentence:
        if (letter == 'м' or letter == 'М'
            or letter == 'н' or letter == 'Н'):
            letters_m_n.append(letter)
    if not letters_m_n:
        print("Букв 'м' и 'н' нет в данном предложении", file=sys.stderr)
        exit(1)
    print("Буквы 'м' и 'н' в предложении:", ', '.join(letters_m_n))

```

```

● aleksejefifanov@MacBook-Pro program % ./ind1.py
Введите предложение на русском языке: Меня зовут Алексей, я сегодня купил мандарин.
Буквы 'м' и 'н' в предложении: М, н, н, м, н, н
⊗ aleksejefifanov@MacBook-Pro program % ./ind1.py
Введите предложение на русском языке: Привет.
Букв 'м' и 'н' нет в данном предложении
○ aleksejefifanov@MacBook-Pro program % |

```

Рисунок 4. Несколько запусков программы индивидуального задания 1

4. Выполнил индивидуальное задание 2 вариант 10: Дана последовательность слов. Проверить, правильно ли в ней записаны буквосочетания жи и ши.

```

#!/usr/bin/env python3
# -*- coding: utf-8 -*-
if __name__ == '__main__':
    words = input("Введите последовательность слов: ").split()
    errors = []
    toogle = False
    for word in words:
        if "жи" in word or "ши" in word:
            toogle = True
        elif "жы" in word or "шы" in word:
            toogle = True
        errors.append(word)
    if not toogle:
        print("В последовательности слов нет буквосочетаний\пкоторые
необходимо проверить")
    elif len(errors) == 0:
        print("В последовательности нет ошибок с буквосочетаниями 'жи' и 'ши'.")
    else:
        print("Обнаружены ошибки в следующих словах:")
        for error in errors:
            if "жы" in error:
                print(f"Ошибка в слове '{error}': буквосочетание 'жы'")
            else:
                print(f"Ошибка в слове '{error}': буквосочетание 'шы'")

```

```

● aleksejefifanov@MacBook-Pro program % ./ind2.py
Введите последовательность слов: Жираф жил жидкость
В последовательности нет ошибок с буквосочетаниями 'жи' и 'ши'.
● aleksejefifanov@MacBook-Pro program % ./ind2.py
Введите последовательность слов: жираф жыл жидкость широко зашивать
Обнаружены ошибки в следующих словах:
Ошибка в слове 'жираф': буквосочетание 'жы'
Ошибка в слове 'жыл': буквосочетание 'жы'
Ошибка в слове 'зашивать': буквосочетание 'шы'
● aleksejefifanov@MacBook-Pro program % ./ind2.py
Введите последовательность слов: три два один
В последовательности слов нет буквосочетаний
которые необходимо проверить
○ aleksejefifanov@MacBook-Pro program % |

```

Рисунок 5. Несколько запусков программы индивидуального задания 2

5. Выполнил индивидуальное задание 3 вариант 10: Дано слово, оканчивающееся символом «.». Вставить заданную букву после первой буквы и.

```

#!/usr/bin/env python3
# -*- coding: utf-8 -*-
import sys
if __name__ == '__main__':
    word = input("Введите слово, оканчивающееся символом '!\n")
    if not word.endswith('.'):
        print("Ошибка: Входное слово не оканчивается символом '!',",
              file=sys.stderr)
        exit(1)
    # Ищем индекс первой буквы "и"
    index_of_i = word.find('и')
    if index_of_i != -1:
        letter = input("Введите букву: ")
        # Вставляем заданную букву после первой буквы "и"
        inserted_word = word[:index_of_i + 1] + \
            letter + word[index_of_i + 1:]
        print("Результат:", inserted_word)
    else:
        print("Ошибка: В слове нет буквы 'и'.", file=sys.stderr)
        exit(1)

```

```

● aleksejerifanov@MacBook-Pro program % ./ind3.py
Введите слово, оканчивающееся символом '.'':
Привет.
Введите букву: ж
Результат: Призвет.
⊗ aleksejerifanov@MacBook-Pro program % ./ind3.py
Введите слово, оканчивающееся символом '.'':
Пока
Ошибка: Входное слово не оканчивается символом '.'!'
⊗ aleksejerifanov@MacBook-Pro program % ./ind3.py
Введите слово, оканчивающееся символом '.'':
Пока.
Ошибка: В слове нет буквы 'и'.
○ aleksejerifanov@MacBook-Pro program % |

```

Рисунок 6. Несколько запусков индивидуального задания 3

6. Выполнил задание повышенной сложности вариант 10: Даны три слова. Напечатать только те буквы слов, которые есть лишь в одном из слов.

Рассмотреть два варианта:

повторяющиеся буквы каждого слова рассматриваются;

повторяющиеся буквы каждого слова не рассматриваются.

```
#!/usr/bin/env python3
```

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

```
def find_unique_letters_with_repeats(words):
```

```
    unique_letters = set(''.join(words)) # Получаем уникальные буквы всех слов
```

```
    result = []
```

```
    for letter in unique_letters:
```

```
        count = 0
```

```
        for word in words:
```

```
            if letter in word:
```

```
                count += 1
```

```
        if count == 1:
```

```
            result.append(letter)
```

```
    return result
```

```
def find_unique_letters_without_repeats(words):
```

```
    unique_letters = set(''.join(words)) # Получаем уникальные буквы всех слов
```

```
    result = []
```

```
    for letter in unique_letters:
```

```
        count = 0
```

```

for word in words:
    if word.count(letter) > 1:
        count = 0
        break
    elif letter in word:
        count += 1
    if count == 1:
        result.append(letter)
return result
if __name__ == '__main__':
    # Запрос ввода трёх слов у пользователя
    word1 = input("Введите первое слово: ")
    word2 = input("Введите второе слово: ")
    word3 = input("Введите третье слово: ")
    words = [word1.lower(), word2.lower(), word3.lower()]
    # Решаем первый вариант и выводим результат
    result1 = find_unique_letters_with_repeats(words)
    print("Повторяющиеся буквы каждого слова рассматриваются:", result1)
    # Решаем второй вариант и выводим результат
    result2 = find_unique_letters_without_repeats(words)
    print("Повторяющиеся буквы каждого слова не рассматриваются:", result2)

```



```

aleksejerifanov@MacBook-Pro program % ./hard_10.py
Введите первое слово: Привет
Введите второе слово: Пока
Введите третье слово: Здравствуй
Повторяющиеся буквы каждого слова рассматриваются: ['д', 'о', 'у', 'з', 'к', 'е', 'и', 'й', 'с']
Повторяющиеся буквы каждого слова не рассматриваются: ['д', 'о', 'у', 'з', 'к', 'е', 'и', 'й', 'с']
aleksejerifanov@MacBook-Pro program % ./hard_10.py
Введите первое слово: папирус
Введите второе слово: бумага
Введите третье слово: кора
Повторяющиеся буквы каждого слова рассматриваются: ['б', 'п', 'г', 'к', 'и', 'о', 'с', 'м']
Повторяющиеся буквы каждого слова не рассматриваются: ['б', 'г', 'к', 'и', 'о', 'с', 'м']
aleksejerifanov@MacBook-Pro program %

```

Рисунок 7. Несколько запусков программы задания повышенной сложности

Ответы на контрольные вопросы:

1. Что такое строки в языке Python?

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. Какие существуют способы задания строковых литералов в языке Python?

Строки в апострофах и в кавычках, экранированные последовательности - служебные символы, «сырые» строки - подавляют экранирование, строки в тройных апострофах или кавычках.

3. Какие операции и функции существуют для строк?

Оператор сложения (+), умножения (*), принадлежности подстроки (in).

Функции:

chr() - преобразует целое число в символ;

ord() - преобразует символ в целое число;

len() - возвращает длину строки;

str() - изменяет тип объекта на string.

4. Как осуществляется индексирование строк?

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. Как осуществляется работа со срезами для строк?

Python также допускает возможность извлечения подстроки из строки, известную как «string slice». Если s это строка, выражение формы s[m:n] возвращает часть s, начинающуюся с позиции m, и до позиции n, но не включая позицию.

Существует еще один вариант синтаксиса среза, о котором стоит упомянуть. Добавление дополнительного «:» и третьего индекса означает шаг, который указывает, сколько символов следует пропустить после извлечения каждого символа в срезе.

6. Почему строки Python относятся к неизменяемому типу данных?

Строки — один из типов данных, которые Python считает неизменяемыми, что означает невозможность их изменять. На самом деле нет

особой необходимости изменять строки. Обычно можно легко сгенерировать копию исходной строки с необходимыми изменениями.

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

`string.istitle()` определяет, начинаются ли слова строки с заглавной буквы.

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

`s.find(<sub>)` возвращает первый индекс в `s` который соответствует началу строки `<sub>`, если же в `s` нет `<sub>`, то функция выдаст -1

9. Как найти индекс первого вхождения подстроки в строку?

`s.find(<sub>)` возвращает первый индекс в `s` который соответствует началу строки `<sub>`, если же в `s` нет `<sub>`, то функция выдаст -1

10. Как подсчитать количество символов в строке?

`len(s)` возвращает количество символов в строке `s`

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

`s.count(<sub>)` возвращает количество точных вхождений подстроки `<sub>` в `s`

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

В Python версии 3.6 был представлен новый способ форматирования строк. Эта функция официально названа литералом отформатированной строки, но обычно упоминается как f- строки (f-string).

Возможности форматирования строк огромны и не будут подробно описана здесь.

Одной простой особенностью f-строк, которые вы можете начать использовать сразу, является интерполяция переменной. Вы можете указать имя переменной непосредственно в f-строковом литерале (f'string'), и python заменит имя соответствующим значением.

Пример: `print(f"Произведение {n} на {m} равно {prod}")`, где `m`, `n`, `prod` это переменные.

13. Как найти подстроку в заданной части строки?

`s.find(подстрока, начало, конец).`

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

`print('{}'.format(s)).`

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

`s.isdigit()` возвращает `True` когда строка `s` не пустая и все ее символы являются цифрами, а `False` если нет

16. Как разделить строку по заданному символу?

`str.split('заданный символ').`

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

`s.islower()` возвращает `True`, если строка `s` не пустая, и все содержащиеся в ней буквенные символы строчные, а `False` если нет. Не алфавитные символы игнорируются

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

`S[0].islower()` выдаст `True` если строка начинается со строчной буквы и `False` если нет.

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

Нет.

20. Как «перевернуть» строку?

`s[::-1]`

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

`str.join('-', s)`, где `s` – это список строк

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

`s.upper()`, `s.lower()`.

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

`string[0].upper() + string[1:-1] + string[-1].upper()`

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

`s.isupper()`.

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

`splitlines()` делит `s` на строки и возвращает их в список. Любой из следующих символов или последовательностей символов считается границей строки: `\n`, `\r`, `\r\n`, `\v` или же `\x0b`, `\f` или же `\x0c`, `\x1c`, `\x1d`, `\x1e`, `\x85`, `\u2028`, `\u2029`.

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

`s.replace(old, new)`.

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

`str.startswith()` и `str.endswith()`.

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

`s.isspace()`.

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

Она напечатается 3 раза.

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

`s.title()`.

31. Как пользоваться методом `partition()`?

Разбивает строку при первом появлении строки аргумента и возвращает кортеж, содержащий часть перед разделителем, строку аргумента и часть после разделителя.

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

`rfind()` и `find()` оба используются для поиска вхождения подстроки в строку, но есть различие в том, что `rfind()` ищет справа налево (с конца строки),

в то время как `find()` ищет слева направо (с начала строки). То есть `rfind()` находит последнее вхождение, а `find()` первое вхождение подстроки в строку.

Вывод: в результате выполнения работы были приобретены навыки по работе со строками при написании программ с помощью языка программирования Python версии 3.x.