

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ
ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего
образования
«СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ
УНИВЕРСИТЕТ»**

**Кафедра
инфокоммуникаций
Институт цифрового
развития**

ОТЧЁТ
по лабораторной работе №2.3
Дисциплина: «Основы программной инженерии»
Тема: «Работа со строками в языке Python»

Выполнила:
студентка 2 курса
группы Пиж-б-о-21-1
Логвинов Иван
Васильевич

Ставрополь 2022

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

1. Был создан общедоступного репозиторий в GitHub в котором были добавлены gitignore, правила для работы с IDE PyCharm с ЯП Python и лицензия MIT, репозиторий был клонировал на локальный сервер и организован в соответствии с моделью ветвления git-flow.

Владелец * / Имя репозитория *

IvanLogvinov11 / lab2_3 ✓

Отличные названия репозитория короткие и запоминающиеся. Вам нужно вдохновение? Как насчет **effective-robot**?

Описание (необязательно)

☒ **Общедоступный**
Любой пользователь Интернета может увидеть этот репозиторий. Вы выбираете, кто может совершать.

☐ **Личное**
Вы выбираете, кто может видеть и фиксировать в этом репозитории.

Инициализируйте этот репозиторий с помощью:
Пропустите этот шаг, если вы импортируете существующий репозиторий.

☒ **Добавьте файл README**
Здесь вы можете написать длинное описание для вашего проекта. [Узнать больше.](#)

Добавить .gitignore
Выберите, какие файлы не отслеживать, из списка шаблонов. [Узнать больше.](#)

.шаблон gitignore: Python ▼

Выберите лицензию
Лицензия сообщает другим пользователям, что они могут и чего не могут делать с вашим кодом. [Узнать больше.](#)

Лицензия: MIT License ▼

Это установит **main** в качестве ветки по умолчанию. Измените имя по умолчанию в настройках .

Вы создаете общедоступный репозиторий в своем личном кабинете.

Создать репозиторий

Рисунок 1.1 – Создание общедоступного репозитория

```

C:\Users\Иван\Desktop\lab2_3>git flow init

Which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [C:/Users/Иван/Desktop/lab2_3/.git/hooks]

C:\Users\Иван\Desktop\lab2_3>
C:\Users\Иван\Desktop\lab2_3>git branch
* develop
  main

C:\Users\Иван\Desktop\lab2_3>

```

Рисунок 1.3 – Репозиторий был организован в соответствии с моделью ветвления git-flow

2. Была создана папка pycharm в которую были помещены примеры.

```

C:\Users\Иван\Desktop\lab2_3\pycharm\venv\Scripts\python.exe
Введите предложение: hello, my name is Ivan
Предложение после замены: {'hello,_my_name_is_Ivan'}

Process finished with exit code 0
|

```

Рисунок 2.1 – Результат работы первого примера

```

C:\Users\Иван\Desktop\lab2_3\pycharm\venv\Scripts\python.exe
Введите слово: ivan
in

Process finished with exit code 0
|

```

Рисунок 2.2 – Результат работы второго примера

```
Введите предложение: My name is Vanya
Введите длину: 25
My    name    is    Vanya

Process finished with exit code 0
|
```

Рисунок 2.3 – Результат работы третьего примера

3. Было выполнено три индивидуальных задания в соответствии с вариантом 10

Задание 1

Дано предложение. Вывести все буквы м и н в нем.

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

import math

if __name__ == '__main__':
    s = input()
    a = s.count("м")
    b = s.count("н")
    print(a + b)
```

```
C:\Users\Иван\Desktop\lab2_3\pycharm\venv\Script
Привет меня зовут Иван
3

Process finished with exit code 0
|
```

Рисунок 3.1 – Результат выполнения программы

Задание 2

Дана последовательность слов. Проверить, правильно ли в ней записаны буквосочетания жи и ши.

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

import math

if __name__ == '__main__':
    s = input().lower()
    s1 = s.replace('жы', 'жи')
    s1 = s1.replace('шы', 'ши')
    if s1 == s:
        print('Ok')
    else:
        print('Есть ошибки')
```

```
C:\Users\Иван\Desktop\lab2_3\pycharm\venv\Sc
жизнь хороша
Ok
Process finished with exit code 0
```

Рисунок 3.2 – Результат работы программы в первом случае

```
C:\Users\Иван\Desktop\lab2_3\pycharm\venv\Script
Жизнь хороша
Есть ошибки
Process finished with exit code 0
```

Рисунок 3.3 – Результат работы программы во втором случае

Задание 3

Дано ошибочно написанное слово рпроцессо. Путем перемещения его букв получить слово процессор.

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

import math

if __name__ == '__main__':
    word = 'процессор'
    print(word[1:] + word[0])
```

```
ind3 x
C:\Users\Иван\Desktop\lab2_3\pycharm\venv\Sc
процессор

Process finished with exit code 0
```

Рисунок 3.4 – Результат работы программы с предложением на русском

Задание повышенной сложности

Дано предложение. Найти длину его самого короткого слова.

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

if __name__ == '__main__':
    min_word_len = float('inf')

    # List of punctuation marks
    punct_marks = [' ', ',', '-', ';', ':', '?', '!', '.']
    counter = 0

    user_str = str(input("Enter your sentence: "))
    str_list = list(user_str)

    # Loop through all characters of the input string
    for i in range(len(str_list)):

        # Checking if a symbol is a punctuation mark
        if str_list[i] in punct_marks:
            word_len = counter
            counter = 0

        # Determine the minimum length of a word
        if 0 < word_len < min_word_len:
            min_word_len = word_len
    else:
        counter += 1
```

```
# Checking the last word, in case of a forgotten dot at the end
word_len = counter
if 0 < word_len < min_word_len:
    min_word_len = word_len

print(min_word_len)
```

```
C:\Users\Иван\Desktop\lab2_3\pycharm\venv\Scripts
Enter your sentence: Денис шел по улице
2

Process finished with exit code 0
```

Результат работы задачи повышенной сложности

```
Командная строка
create mode 100644 pycharm/3.py
create mode 100644 pycharm/ind1.py
create mode 100644 pycharm/ind2.py
create mode 100644 pycharm/ind3.py

C:\Users\Иван\Desktop\lab2_3>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

C:\Users\Иван\Desktop\lab2_3>git merge develop
Updating 1461614..be58461
Fast-forward
 pycharm/.idea/.gitignore      | 0
 .../.idea/inspectionProfiles/profiles_settings.xml | 6 +
 pycharm/.idea/misc.xml        | 4 +
 pycharm/.idea/modules.xml     | 8 +
 pycharm/.idea/pycharm.iml     | 10 ++
 pycharm/.idea/vcs.xml         | 6 +
 pycharm/.idea/workspace.xml   | 187 ++++++
 pycharm/1.py                  | 6 +
 pycharm/2.py                  | 12 ++
 pycharm/3.py                  | 48 +++++
 pycharm/ind1.py               | 4 +
 pycharm/ind2.py               | 7 +
 pycharm/ind3.py               | 2 +
13 files changed, 300 insertions(+)
create mode 100644 pycharm/.idea/.gitignore
create mode 100644 pycharm/.idea/inspectionProfiles/profiles_settings.xml
create mode 100644 pycharm/.idea/misc.xml
create mode 100644 pycharm/.idea/modules.xml
```

Рисунок 3.5 – Коммит изменений и слияние ветки develop с main

```
C:\Users\Иван\Desktop\lab2_3>git push
Enumerating objects: 19, done.
Counting objects: 100% (19/19), done.
Delta compression using up to 8 threads
Compressing objects: 100% (16/16), done.
Writing objects: 100% (18/18), 4.26 KiB | 1.06 MiB/s, done.
Total 18 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
To https://github.com/IvanLogvinov11/lab2_3.git
   1461614..be58461  main -> main

C:\Users\Иван\Desktop\lab2_3>
```

Рисунок 3.6 – Пуш на удаленный сервер

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

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, "сырые" строки, строки в тройных апострофах или кавычках.

3. Какие операции и функции существуют для строк?

Сложение, дублирование, длина строки, длина строки, извлечение среза

и т. д.

4. Как осуществляется индексирование строк?

Доступ к символам в строках основан на операции индексирования — после строки или имени переменной, ссылающейся на строку, в квадратных скобках указываются номера позиций необходимых символов.

5. Как осуществляется работа со срезами для строк?

Есть три формы срезов. Самая простая форма среза: взятие одного символа строки, а именно, $S[i]$ — это срез, состоящий из одного символа, который имеет номер i , при этом считая, что нумерация начинается с числа 0. То есть если $S = \text{'Hello'}$, то $S[0] == \text{'H'}$, $S[1] == \text{'e'}$, $S[2] == \text{'l'}$, $S[3] == \text{'l'}$, $S[4] == \text{'o'}$.

Если указать отрицательное значение индекса, то номер будет отсчитываться с конца, начиная с номера -1.

Срез с двумя параметрами: $S[a:b]$ возвращает подстроку из $b-a$ символов, начиная с символа с индексом a , то есть до символа с индексом b , не включая его.

6. Почему строки Python относятся к неизменяемому типу данных?

Строки — один из типов данных, которые Python считает неизменяемыми, что означает невозможность их изменять. Python дает возможность изменять (заменять и перезаписывать) строки.

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

`string.istitle()`

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

`string.find()`

9. Как найти индекс первого вхождения подстроки в строку?

`s.partition(<sep>)`

10. Как подсчитать количество символов в строке?

`len(s)`

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

`s.count(<sub>)`

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

Эти строки улучшают читаемость кода, а также работают быстрее чем другие способы форматирования. F-строки задаются с помощью литерала «f» перед кавычками. Пример: `print(f"Меня зовут {name} Мне {age} лет.")`

13. Как найти подстроку в заданной части строки?

`s.find(значение, 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()`

16. Как разделить строку по заданному символу букв?

`str.split()`

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

`s.isalpha()`

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

`s.istitle()`

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

Нет

20. Как «перевернуть» строку?

`s.reverse()`

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

`str.split('-')`

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

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

23. Как преобразовать первый символ строки к верхнему регистру?

`s.capitalize()`

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

`s.isupper()`

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

`s.splitlines()` делит `s` на строки и возвращает их в списке. Любой из следующих символов или последовательностей символов считается границей строки.

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?

`Asd*3 = AsdAsdAsd`

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

`s.title()`

31. Как пользоваться методом `partition()`?

Метод `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()`?

`s.rfind(<sub>)` возвращает индекс последнего вхождения подстроки `<sub>` в `s`, который соответствует началу `<sub>`.