

**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  
Джолдошова Мээрим  
Бекболотовна

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.

**Create a new repository**

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

**Owner \*** meiokava / **Repository name \*** lbrt\_2.3 ✓

Great repository names are short and memorable. Need inspiration? How about [jubilant-system?](#)

**Description** (optional)

☒ **Public**  
Anyone on the internet can see this repository. You choose who can commit.

☐ **Private**  
You choose who can see and commit to this repository.

**Initialize this repository with:**  
Skip this step if you're importing an existing repository.

☒ **Add a README file**  
This is where you can write a long description for your project. [Learn more.](#)

**Add .gitignore**  
Choose which files not to track from a list of templates. [Learn more.](#)

**.gitignore template:** Python

**Choose a license**  
A license tells others what they can and can't do with your code. [Learn more.](#)

**License:** MIT License

ⓘ You are creating a public repository in your personal account.

**Create repository**

Рисунок 1.1 – Создание общедоступного репозитория

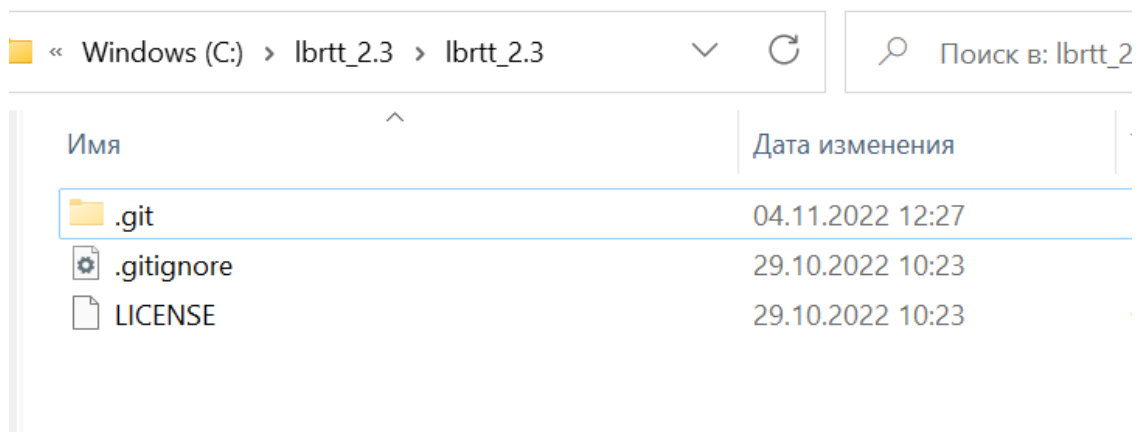


Рисунок 1.2 – Репозиторий был успешно клонирован

```
C:\lbrrt_2.3\lbrrt_2.3>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:/lbrrt_2.3/lbrrt_2.3/.git/hooks]

C:\lbrrt_2.3\lbrrt_2.3>git branch -a
* develop
  main
remotes/origin/HEAD -> origin/main
remotes/origin/main

C:\lbrrt_2.3\lbrrt_2.3>
```

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\мвидео\PycharmProjects\pythonProject26\venv\Scripts\python.exe C:\Users\мвидео\PycharmPr
Введите предложение: hey there not there
Предложение после замены: hey_there_not_there

Process finished with exit code 0
```

Рисунок 2.1 – Результат работы первого примера

```
C:\Users\мвидео\PycharmProjects\pythonProject27\venv\Scripts\python.exe C:\Users\мвидео\PycharmProjects\pythonProject27\venv\Scripts\python.exe
Введите слово: python
python
Process finished with exit code 0
```

Рисунок 2.2 – Результат работы второго примера

```
C:\Users\мвидео\PycharmProjects\pythonProject28\venv\Scripts\python.exe C:\Users\мвидео\PycharmProjects\pythonProject28\venv\Scripts\python.exe
Введите предложение: hello world
Введите длину: 13
hello world
Process finished with exit code 0
```

Рисунок 2.3 – Результат работы третьего примера

3. Было выполнено три индивидуальных задания в соответствии с вариантом 5

#### Задание 1

Дано слово. Добавить к нему в начале и конце столько звездочек, сколько букв в этом слове.

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

if __name__ == '__main__':
    word = input("enter word: ")
    print('*' * len(word) + word + ('*' * len(word)))
```

```
C:\Users\мвидео\PycharmProjects\pythonProject33\venv\Scripts\python.exe C:\Users\мвидео\PycharmProjects\pythonProject33\venv\Scripts\python.exe
enter word: genesis
*****genesis*****
Process finished with exit code 0
```

Рисунок 3.1 – Результат выполнения программы

#### Задание 2

Даны два слова. Определить, сколько начальных букв первого слова

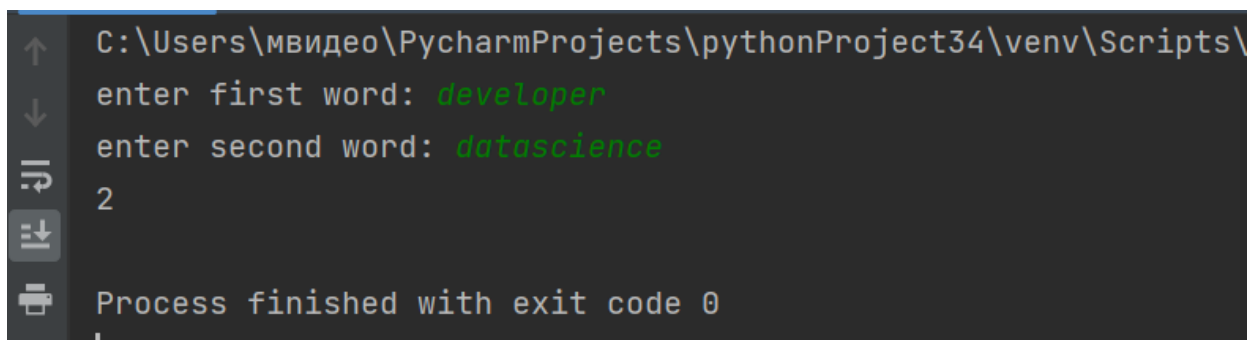
совпадает с начальными буквами второго слова. Рассмотреть два случая:

известно, что слова разные;

слова могут быть одинаковыми.

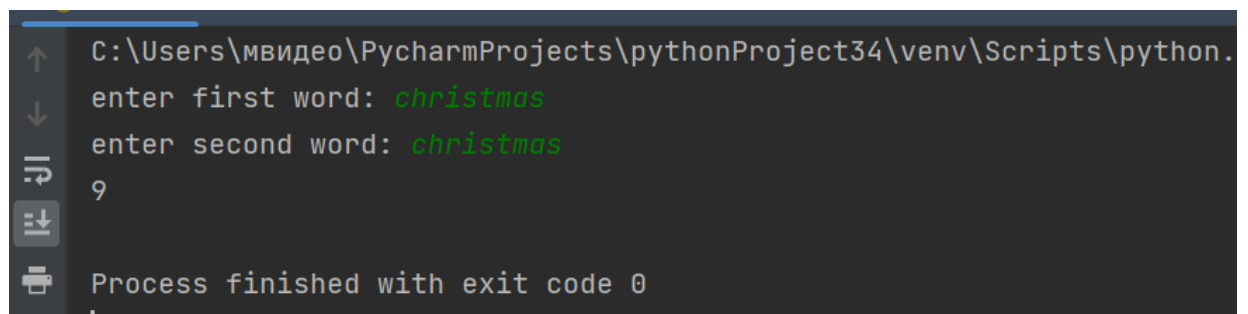
```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

if __name__ == '__main__':
    word1 = input('enter first word: ')
    word2 = input('enter second word: ')
    k = 0
    for i in range(len(word1)):
        if word1[i] == word2[i]:
            k += 1
    print(k)
```



```
C:\Users\мвидео\PycharmProjects\pythonProject34\venv\Scripts\python.
enter first word: developer
enter second word: datascience
2
Process finished with exit code 0
```

Рисунок 3.2 – Результат работы программы в первом случае



```
C:\Users\мвидео\PycharmProjects\pythonProject34\venv\Scripts\python.
enter first word: christmas
enter second word: christmas
9
Process finished with exit code 0
```

Рисунок 3.3 – Результат работы программы во втором случае

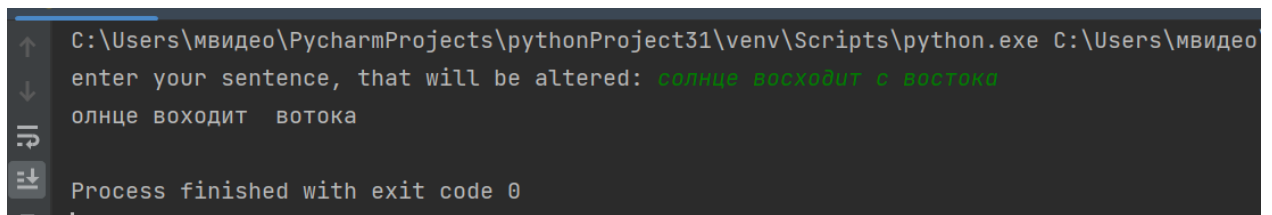
### Задание 3

Дано предложение. Удалить из него все буквы с (как в кириллице так и на латинице).

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

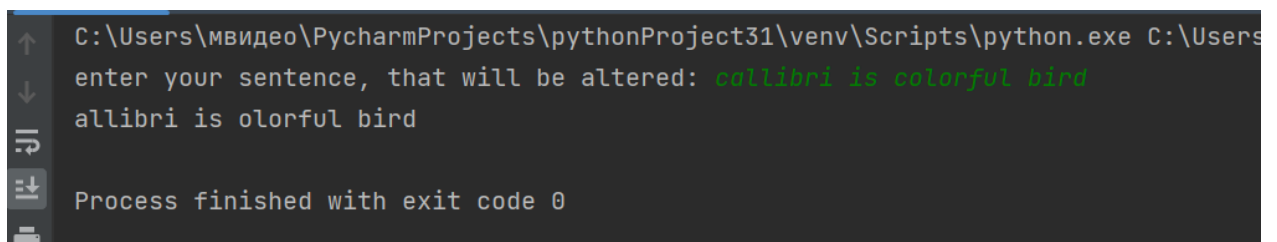
if __name__ == '__main__':
    sentence = input("enter your sentence, that will be altered: ")
    sentence = sentence.replace('с', '') # for eng
```

```
sentence = sentence.replace('с', ' ') # for rus
print(sentence)
```



```
C:\Users\мвидео\PycharmProjects\pythonProject31\venv\Scripts\python.exe C:\Users\мвидео
enter your sentence, that will be altered: солнце восходит с востока
олнце входит востока
Process finished with exit code 0
```

Рисунок 3.4 – Результат работы программы с предложением на русском



```
C:\Users\мвидео\PycharmProjects\pythonProject31\venv\Scripts\python.exe C:\Users\мвидео
enter your sentence, that will be altered: collibri is colorful bird
allibri is olorful bird
Process finished with exit code 0
```

Рисунок 3.5 – Результат работы программы с предложением на английском

4. Было выполнено задание повышенной сложности согласно варианту. Были зафиксированы изменения и слита ветка develop с веткой main.

Даны два слова. Для каждой буквы первого слова (в том числе для повторяющихся в этом слове букв) определить, входит ли она во второе слово. Например, если заданные слова информация и процессор, то для букв первого из них ответом должно быть: нет нет нет да да нет нет да нет нет.

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

if __name__ == '__main__':
    word1 = input('enter first word: ')
    word2 = input('enter second word: ')
    k = 0
    for i in word1:
        print(k + 1, " letter ", i in word2)
        k += 1
```

```
C:\Users\мвидео\PycharmProjects\pythonProject32\venv\Scripts\python.exe C:\Use
enter first word: python
enter second word: anaconda
1 letter False
2 letter False
3 letter False
4 letter False
5 letter True
6 letter True

Process finished with exit code 0
```

Рисунок 4.1 – Результат работы программы

```
c:\lbrtt_2.3\lbrtt_2.3>git add .
c:\lbrtt_2.3\lbrtt_2.3>git commit -m "completed tasks"
[develop 66ebd77] completed tasks
6 files changed, 101 insertions(+)
create mode 100644 indiv_tasks/indiv1.py
create mode 100644 indiv_tasks/indiv3.py
create mode 100644 pycharm/example1.py
create mode 100644 pycharm/example2.py
create mode 100644 pycharm/example3.py
create mode 100644 up_diff/up_diff5.py
c:\lbrtt_2.3\lbrtt_2.3>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
c:\lbrtt_2.3\lbrtt_2.3>git merge develop
Updating b7e6f54..66ebd77
Fast-forward
 indiv_tasks/indiv1.py | 6 ++++++
 indiv_tasks/indiv3.py | 8 +++++++
 pycharm/example1.py   | 7 ++++++
 pycharm/example2.py   | 13 ++++++++
 pycharm/example3.py   | 57 ++++++++
 up_diff/up_diff5.py   | 10 ++++++++
6 files changed, 101 insertions(+)
create mode 100644 indiv_tasks/indiv1.py
create mode 100644 indiv_tasks/indiv3.py
create mode 100644 pycharm/example1.py
create mode 100644 pycharm/example2.py
create mode 100644 pycharm/example3.py
create mode 100644 up_diff/up_diff5.py
c:\lbrtt_2.3\lbrtt_2.3>_
```

Рисунок 4.2 – Коммит изменений и слияние ветки develop с main

```
c:\lbrtt_2.3\lbrtt_2.3>git push
Enumerating objects: 12, done.
Counting objects: 100% (12/12), done.
Delta compression using up to 8 threads
Compressing objects: 100% (10/10), done.
Writing objects: 100% (11/11), 2.12 KiB | 1.06 MiB/s, done.
Total 11 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/meiokava/lbrtt_2.3.git
 b7e6f54..66ebd77 main -> main
c:\lbrtt_2.3\lbrtt_2.3>_
```

Рисунок 4.3 – 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, 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>`.