

**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 кроссплатформенного программирования»

Тема: «Работа со строками в языке Python»

Выполнил: студент 1 курса

группы ИВТ-б-о-21-1

Харченко Богдан Романович

Ставрополь 2022

Выполнение работы:

1. Создал репозиторий в GitHub «Lab_6» в который добавил .gitignore, который дополнил правила для работы с IDE PyCharm с ЯП Python, выбрал лицензию MIT, клонировал его на лок. сервер и организовал в соответствии с моделью ветвления git-flow.

Owner * / Repository name *

Konstellation / Lab_6 ✓

Great repository names are short and memorable. Need inspiration? How about **fuzzy-giggle**?

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 ▾

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

Create repository

Рисунок 1.1 Создание репозитория

```
C:\>git clone https://github.com/Konstellation/Lab_6.git
Cloning into 'Lab_6'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (4/4), done.
```

Рисунок 1.2 Клонирование репозитория

```
C:\Lab_6>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:/Lab_6/.git/hooks]
```

Рисунок 1.3 Организация репозитория в соответствии с моделью ветвления
git-flow

```
.idea/
# Created by https://www.toptal.com/developers/gitignore/api/python,pycharm
# Edit at https://www.toptal.com/developers/gitignore?templates=python,pycharm

### PyCharm ###
# Covers JetBrains IDEs: IntelliJ, RubyMine, PhpStorm, AppCode, PyCharm, CLion, Android Studio
# Reference: https://intellij-support.jetbrains.com/hc/en-us/articles/206544839

# User-specific stuff
.idea/**/workspace.xml
.idea/**/tasks.xml
.idea/**/usage.statistics.xml
.idea/**/dictionaries
.idea/**/shelf

# AWS User-specific
.idea/**/aws.xml
```

Рисунок 1.4 Изменение .gitignore

2. Создал проект PyCharm в папке репозитория, проработал примеры ЛР.

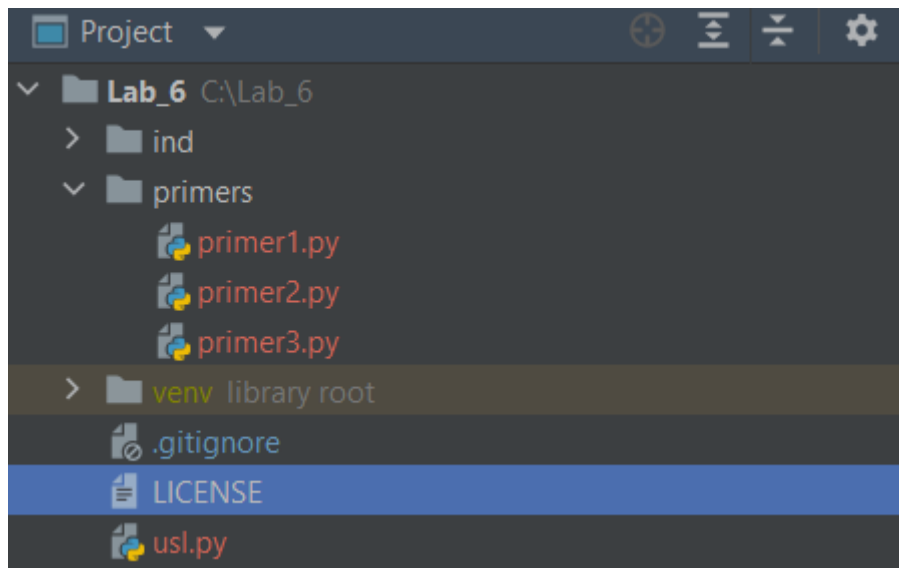


Рисунок 2.1 Создание проекта в PyCharm

```
Введите предложение: asdf asdf asdfnjk   asd s sasdf
Предложение после замены: asdf_asdf_asdfnjk___asd_s_sasdf
```

Рисунок 2.2 Рез-т выполнения программы

```
Введите слово: gashkdf
gaskdf
```

Рисунок 2.3 Рез-т выполнения программы

```
Введите предложение: qwe qweq qqw   er w a asdfda rt
Введите длину: 39
qwe   qweq   qqw   er   w   a   asdfda   rt
```

Рисунок 2.4 Рез-т выполнения программы

3. (21 вариант). Выполнил 3 индивидуальных задания и задание повышенной сложности.

```

1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 ▶ if __name__ == '__main__':
5     p = input("Введите предложение: ")
6     sl_2 = input("Введите буквосочетание из двух букв: ")
7     sl_n = input("Введите буквосочетание: ")
8
9     print('Количество буквосочетания po в предложении = ', p.count('po'))
10    print('Количество буквосочетания ', sl_2, ' в предложении = ', p.count(sl_2))
11    print('Количество буквосочетания ', sl_n, ' в предложении = ', p.count(sl_n))
12

```

Рисунок 3.1 Листинг программы для задания 1

```

Введите предложение: asda asdf asdf asdpo power odopa
Введите буквосочетание из двух букв: oa
Введите буквосочетание: asd
Количество буквосочетания po в предложении = 2
Количество буквосочетания oa в предложении = 0
Количество буквосочетания asd в предложении = 4

```

Рисунок 3.2 Выполнение программы

```

1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 ▶ if __name__ == '__main__':
5     w = str(input('Введите слово: '))
6     tmp = list(w)
7
8     s = tmp[1]
9     tmp[1] = tmp[4]
10    tmp[4] = s
11
12    w = ''.join(tmp)
13    print(w)
14

```

Рисунок 3.3 Листинг программы для задания 2

Введите слово: *adsfas*
aasfds

Рисунок 3.4 Выполнение программы

```
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 ▶ if __name__ == '__main__':
5     w = 'ИТЕРНЕТ'
6     tmp = list(w)
7
8     s = tmp[-1]
9     for i in range(len(w)-1, 1, -1):
10         tmp[i] = tmp[i-1]
11     tmp[1] = s
12
13     w = ''.join(tmp)
14     print(w)
15
```

Рисунок 3.5 Листинг программы для задания 3

ИТЕРНЕТ

Process finished with exit code 0

Рисунок 3.6 Выполнение программы

```

1  ▶  #!/usr/bin/env python3
2      # -*- coding: utf-8 -*-
3
4  ▶  if __name__ == '__main__':
5      p = input('Введите предложение: ')
6      tmp = p.split()
7      f = False
8      for i in tmp:
9          if len(i) > 10:
10             f = True
11             break
12      if f:
13          print('Самое длинное слово имеет больше 10 символов')
14      else:
15          print('Самое длинное слово НЕ имеет больше 10 символов')
16

```

Рисунок 3.7 Листинг программы усложненного задания

```

Введите предложение: asdf asdasdf h kjhashd sd asdfajlkghasd
Самое длинное слово имеет больше 10 символов

```

Рисунок 3.8 Выполнение программы

4. Сделал коммит, выполнил слияние с веткой main, и запустил изменения в уд. репозиторий.

```

C:\Users\adamk\OneDrive\Рабочий стол\rep_2.3>git add .
C:\Users\adamk\OneDrive\Рабочий стол\rep_2.3>git commit -m "formatted according to pep8"
[develop 3cb8760] formatted according to pep8
 4 files changed, 8 insertions(+), 4 deletions(-)

```

Рисунок 4.1 Фиксация и коммит файлов

```

C:\Lab_6>git merge develop
Updating 1b5d880..faba4fd
Fast-forward
 ind/ind1.py      | 11 ++++++
 ind/ind2.py      | 13 ++++++
 ind/ind3.py      | 14 ++++++
 primers/primer1.py | 7 +++++
 primers/primer2.py | 13 ++++++
 primers/primer3.py | 57 ++++++
 usl.py           | 15 ++++++
 7 files changed, 130 insertions(+)
 create mode 100644 ind/ind1.py
 create mode 100644 ind/ind2.py
 create mode 100644 ind/ind3.py
 create mode 100644 primers/primer1.py
 create mode 100644 primers/primer2.py
 create mode 100644 primers/primer3.py
 create mode 100644 usl.py

```


Рисунок 4.2 Слияние ветки develop с main

```
C:\Lab_6>git push
info: please complete authentication in your browser...
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 2.71 KiB | 2.71 MiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Konstellation/Lab_6.git
c8d75a6..1b5d880 main -> main
```

Рисунок 4.3 Пуш коммитов

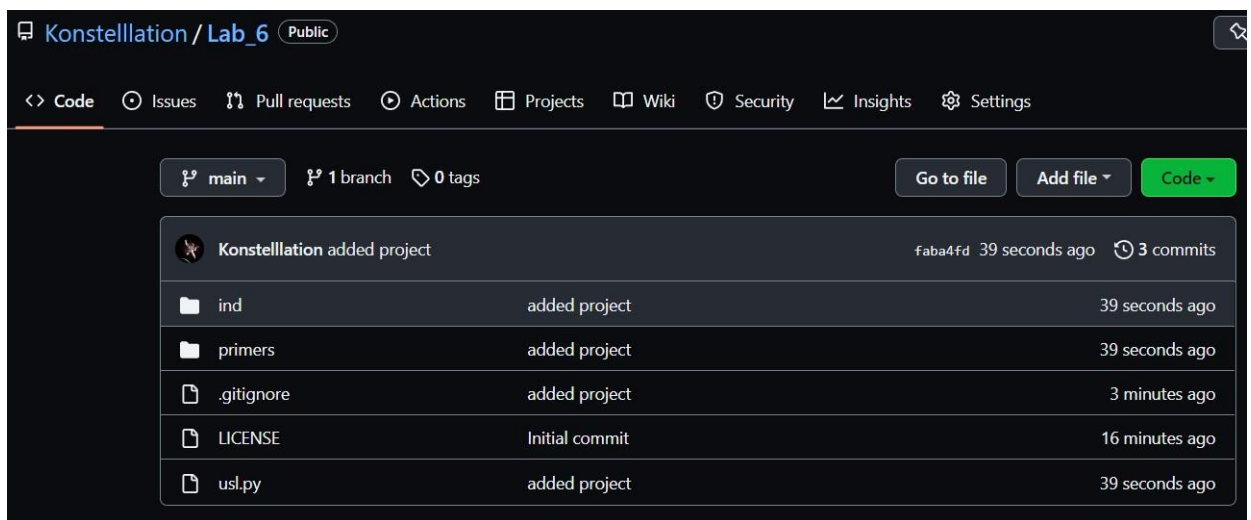


Рисунок 4.4 Изменения на уд. сервере

Контр. вопросы и ответы на них:

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 = 'Hello'`, то `S[0]=='H'`, `S[1]=='e'`, `S[2]=='l'`, `S[3]=='l'`, `S[4]=='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`-а 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?

`str.split()`

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

`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>`.