

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное учреждение высшего
образования**

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития

Кафедра информационных систем и технологий

Отчет по лабораторной работе №6.

Дисциплина: «Основы программной инженерии»

Выполнил:

Студент группы ПИЖ-б-о-22-1,
направление подготовки: 09.03.04

«Программная инженерия»

ФИО: Джараян Арег Александрович

Проверил:

Воронкин Р. А.

Ставрополь 2022

Тема: Лабораторная работа 2.1 Основы языка Python

Цель работы: исследование процесса установки и базовых возможностей языка Python версии 3.x.

Выполнение работы:


1. Изучил теоретический материал работы.
2. Создала репозиторий на git.hub.

Создайте новый репозиторий

Репозиторий содержит все файлы проекта, включая историю изменений. У вас уже есть репозиторий проекта в другом месте? [Импортируйте репозиторий](#).

Обязательные поля отмечены звездочкой (*).

Владелец *

 мрядская ▾

Название репозитория *

/ lab6


✔ доступен lab6.

Отличные названия репозитория короткие и запоминающиеся. Вам нужно вдохновение? Как насчет [переработанного octo-waffle?](#)

Описание (необязательно)

☒  **Общедоступный**

Этот репозиторий может видеть любой пользователь Интернета. Вы сами выбираете, кто может совершать коммиты.

☐  **Приватный**

Вы сами выбираете, кто может просматривать этот репозиторий и вносить в него изменения.

Инициализируйте этот репозиторий с помощью:

☒ **Добавьте файл README**

Здесь вы можете написать длинное описание вашего проекта. [Узнайте больше о READMEs.](#)

Добавить файл .gitignore

Рисунок 1 – создание репозитория

3. Клонировала репозиторий.

```
C:\>cd git1

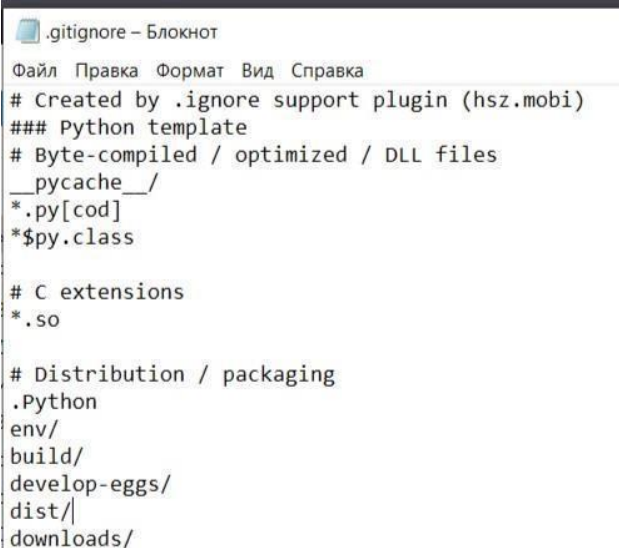
C:\git1>git clone https://github.com/mryadskaya/lab6.git
Cloning into 'lab6'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.

C:\git1>cd lab6

C:\git1\lab6>
```

Рисунок 2 – клонирование репозитория 4.

Дополнить файл gitignore необходимыми правилами.



```
.gitignore - Блокнот
Файл Правка Формат Вид Справка
# Created by .ignore support plugin (hsz.mobi)
### Python template
# Byte-compiled / optimized / DLL files
__pycache__/
*.py[cod]
*$py.class

# C extensions
*.so

# Distribution / packaging
.Python
env/
build/
develop-eggs/
dist/
downloads/
```

Рисунок 3 - .gitignore для IDE PyCharm

5. Организовать свой репозиторий в соответствии с моделью ветвления git-flow.

```

C:\git1\lab6>git branch develop

C:\git1\lab6>git push -u origin develop
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'develop' on GitHub by visiting:
remote:   https://github.com/mryadskaya/lab6/pull/new/develop
remote:
To https://github.com/mryadskaya/lab6.git
 * [new branch]      develop -> develop
branch 'develop' set up to track 'origin/develop'.

C:\git1\lab6>git checkout develop
Switched to branch 'develop'
Your branch is up to date with 'origin/develop'.

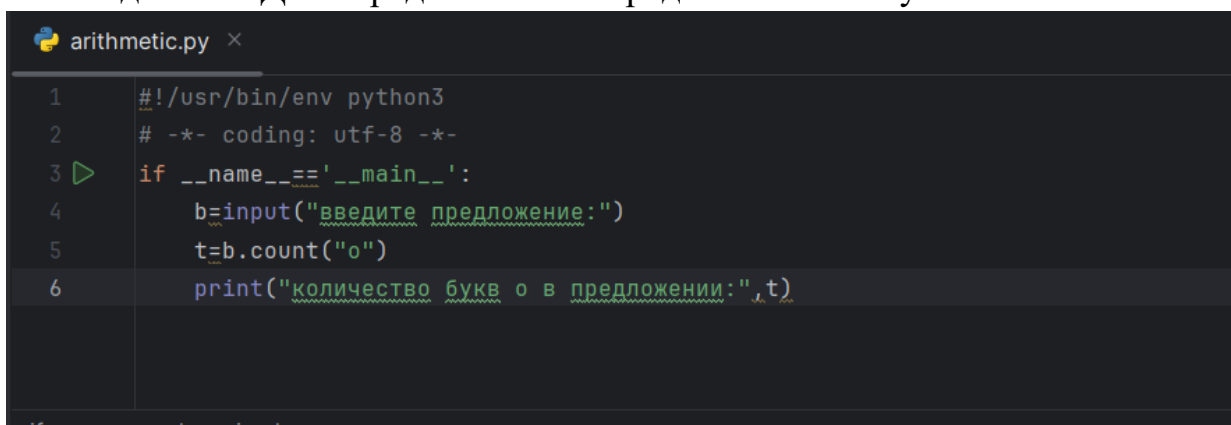
C:\git1\lab6>

```

Рисунок 4 – создание ветки develop

6. Проработать примеры лабораторной работы. Создать для каждого примера отдельный модуль языка Python. Зафиксировать изменения в репозитории. Привести в отчете скриншоты результатов выполнения каждой из программ примеров при различных исходных данных вводимых с клавиатуры.

Задание 1: Дано предложение. Определить число букв о в нем.



```

arithmetic.py x
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3  if __name__ == '__main__':
4      b=input("введите предложение:")
5      t=b.count("o")
6      print("количество букв о в предложении:",t)

```

Рисунок 5 –задание 1

```
C:\Users\ADMIN\PycharmProjects\pythonProject\venv\Scripts\python.exe 0
введите предложение:предложение о предложении
количество букв о в предложении: 3

Process finished with exit code 0
```

Рисунок 6 – примеры выполнения для задания1

Задание 2: Дано предложение. Все его символы, стоящие на четных местах, заменить буквой ы.

```
1      ex1.py      ex2.py x
1      #!/usr/bin/env python3
2      # -*- coding: utf-8 -*-
3      if __name__ == '__main__':
4          r=input("введите предложение:")
5          m=""
6          for i in range(len(r)):
7              if i%2==1:
8                  m+=r[i]
9              else:
10                 m+="ы"
11      print("исходное предложение",r)
12      print("изменённое предложение",m)
```

Рисунок 7 –задание 2

```

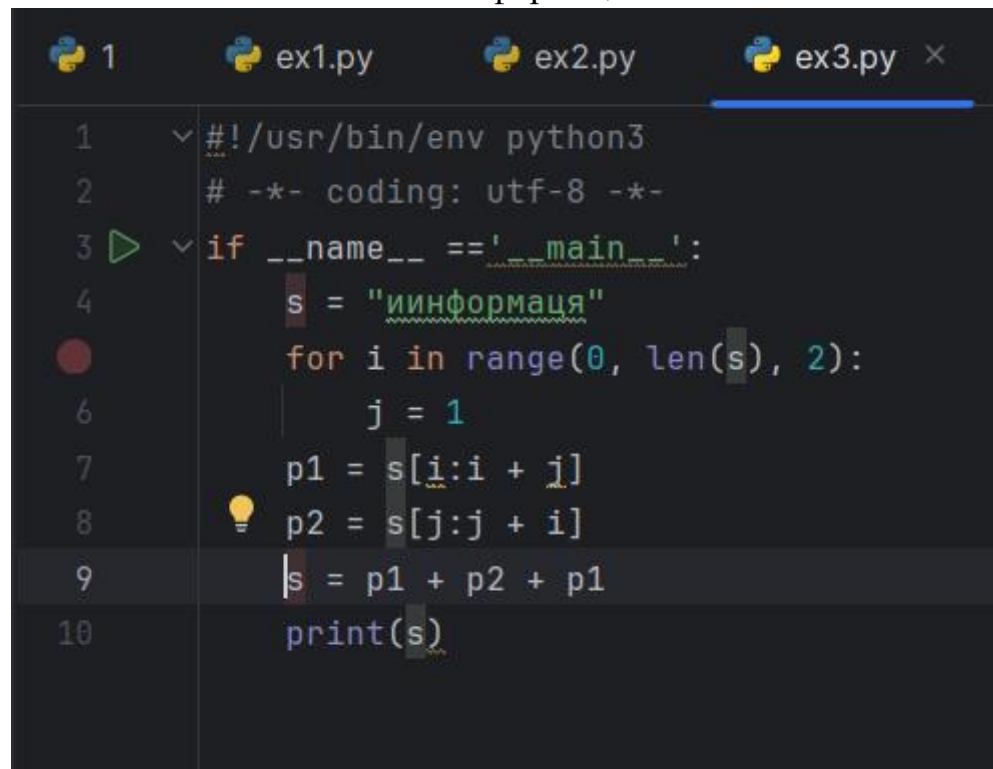
введите предложение:предложение
исходное предложение предложение
изменённое предложение ырыдыыыыыыы

Process finished with exit code 0

```

Рисунок 8 – примеры выполнения для задания 2

Задание 3: Дано ошибочно написанное слово и информация. Путём перемещения его букв получить слово информация.



```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3  if __name__ == '__main__':
4      s = "иинформация"
5      for i in range(0, len(s), 2):
6          j = 1
7          p1 = s[i:i + j]
8          p2 = s[j:j + i]
9          s = p1 + p2 + p1
10     print(s)

```

Рисунок 9 – пример 3

```

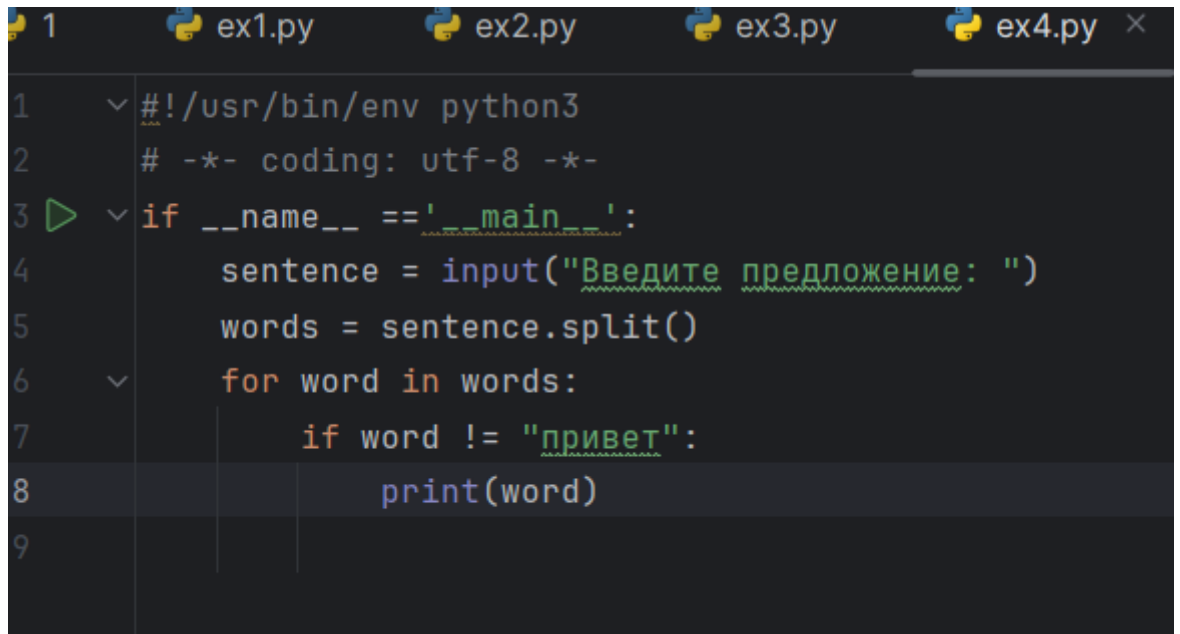
C:\Users\ADMIN\PycharmProjects\pythonProject\venv\Scripts\
информация

Process finished with exit code 0

```

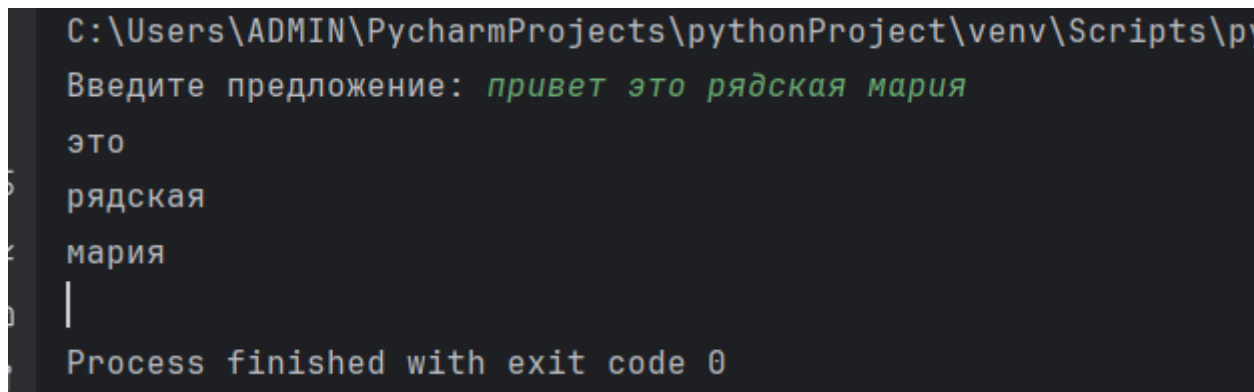
Рисунок 10 – примеры выполнения для примера 3

Задание повышенной сложности: 15 Дано предложение. Напечатать все его слова, отличные от слова привет.



```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3  if __name__ == '__main__':
4      sentence = input("Введите предложение: ")
5      words = sentence.split()
6      for word in words:
7          if word != "привет":
8              print(word)
```

Рисунок 11 – решение задания



```
C:\Users\ADMIN\PycharmProjects\pythonProject\venv\Scripts\python.exe
Введите предложение: привет это рядская мария
это
рядская
мария
|
Process finished with exit code 0
```

Рисунок 12– результат выполнения задания

9. Зафиксировал все изменения в github в ветке develop.

```
C:\git1\lab6>git add .
C:\git1\lab6>git status
On branch develop
Your branch is up to date with 'origin/develop'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   PyCharm/ex1.py
        new file:   PyCharm/ex2.py
        new file:   PyCharm/ex3.py
        new file:   PyCharm/ex4.py

C:\git1\lab6>git commit -m"сохранение изменений"
[develop df8c68b] сохранение изменений
 4 files changed, 36 insertions(+)
 create mode 100644 PyCharm/ex1.py
 create mode 100644 PyCharm/ex2.py
 create mode 100644 PyCharm/ex3.py
 create mode 100644 PyCharm/ex4.py

C:\git1\lab6>git push
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 2 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (7/7), 1.25 KiB | 182.00 KiB/s, done.
Total 7 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/mryadskaya/lab6.git
   b10f07b..df8c68b  develop -> develop

C:\git1\lab6>
```

Рисунок 19 – фиксация изменений в ветку develop

10.Слила ветки.


```
C:\git1\lab6>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

C:\git1\lab6>git merge develop
Updating b10f07b..df8c68b
Fast-forward
 PyCharm/ex1.py | 6  ++++++
 PyCharm/ex2.py | 12 ++++++++
 PyCharm/ex3.py | 10 ++++++++
 PyCharm/ex4.py | 8  ++++++
4 files changed, 36 insertions(+)
 create mode 100644 PyCharm/ex1.py
 create mode 100644 PyCharm/ex2.py
 create mode 100644 PyCharm/ex3.py
 create mode 100644 PyCharm/ex4.py

C:\git1\lab6>
```

Рисунок 20 – сливание ветки develop в ветку main

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?

Работа со строками в Python очень удобна. Существует несколько литералов строк. (строки в апострофах и в кавычках, экранированные последовательности - служебные символы, строки в тройных апострофах или кавычках).

3. Какие операции и функции существуют для строк?

Некоторые функции: chr(), ord(), len(), str(). Строки можно умножать на числа, строки можно прибавлять между собой.

4. Как осуществляется индексирование строк?

Индексирование строк осуществляется с использованием квадратных скобок [], и индексы начинаются с 0. Вы можете получить доступ к отдельным символам в строке или извлекать подстроки, указывая индексы. Можно указывать отрицательные индексы, тогда счёт пойдет с обратной стороны.

5. Как осуществляется работа со срезами для строк?

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

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 вы можете легко сгенерировать копию исходной строки с необходимыми изменениями. Есть минимум 2 способа сделать это в python.

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

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

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

Можно проверить оператором “in”.

9. Как найти индекс первого вхождения подстроки в строку?

string.find(<sub>[, <start>[, <end>]]) ищет в строке заданную подстроку. s.find(<sub>) - возвращает первый индекс в s который соответствует началу строки <sub>

10. Как подсчитать количество символов в строке?

Можно воспользоваться len(строка).

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

string.count(<sub>[, <start>[, <end>]]) -подсчитывает количество вхождений подстроки в строку.

12 Что такое f-строки и как ими пользоваться?

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.

13 Как найти подстроку в заданной части строки?

Для поиска подстроки в заданной части строки в Python вы можете использовать метод строки find(), который вернет индекс начала первого вхождения подстроки в заданной части строки. Если подстрока не найдена, метод вернет -1.

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

Переменную нужно указать внутри {}.

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?

Для разделения строки по заданному символу или подстроке в Python вы можете использовать метод строки `split()`. Этот метод разбивает строку на список подстрок с использованием указанного разделителя и возвращает этот список. `string.rsplit(sep=None, maxsplit=-1)` делит строку на список из подстрок.

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

`s.isupper()` возвращает `true` , если строка `s` не пустая, и все содержащиеся в ней буквенные

символы являются заглавными, и в `false` , если нет.

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

`str.islower()`. Этот метод возвращает `True`, если первый символ строки является строчной буквой, и `False` в противном случае.

19 Можно ли в Python прибавить целое число к строке?

при попытке выполнения подобной операции будет выдана ошибка `TypeError`.

20 Как «перевернуть» строку?

Можно указать `[::-1]`

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

Для объединения списка строк в одну строку, где элементы разделяются дефисами, вы можете использовать метод строки `join()`

22 Как привести всю строку к верхнему или нижнему регистру?

Для приведения строки к верхнему (заглавному) или нижнему (строчному) регистру в Python, вы можете использовать методы строк `upper()` и `lower()`, соответственно.

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

Для преобразования первого и последнего символов строки к верхнему регистру в Python, вы можете использовать методы строк `upper()` и `lower()` в сочетании с конкатенацией строк.

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

`s.islower()` возвращает `true` , если строка `s` не пустая, и все содержащиеся в нем буквенные

символы строчные, а `false` если нет.

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

Когда нужно делить `s` на строки и возвращать 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?

Для замены всех вхождений определенной подстроки в заданной строке в Python, вы можете использовать метод строки `replace()`. Этот метод заменяет все вхождения подстроки на другую подстроку и возвращает новую строку с заменами.

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

Можно сравнить нужную нам последовательность символов с индексом той части строки, в которой должна быть последовательность символов.

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

`s.isspace()` возвращает `True`, если `s` не пустая строка, и все символы являются пробельными, а `False`, если нет

29 Что случится, если умножить некую строку на 3?

Строка повторится 3 раза.

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

`s.title()` возвращает копию, `s` в которой первая буква каждого слова преобразуется в

верхний регистр, а остальные буквы — в нижний регистр:

31 Как пользоваться методом `partition()` ?

`partition()` - это метод строки в Python, который позволяет разделить строку на три части с использованием заданного разделителя. Метод возвращает кортеж, в котором первый элемент - это часть строки до первого вхождения разделителя, второй элемент - сам разделитель, и третий элемент - часть строки после первого вхождения разделителя.

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

Метод `rfind()` используется в Python для поиска последнего вхождения подстроки в строке. Он возвращает индекс последнего вхождения заданной подстроки в строке. Если подстрока не найдена, метод `rfind()` возвращает `-1`. `rfind()`. Когда нам нужно найти последнее вхождение определенной подстроки в строке, `rfind()` позволяет это сделать без необходимости итерации с конца строки.