

**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: Гуртовой Ярослав Дмитриевич

Проверил:

Воронкин Р. А.

Ставрополь 2023

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. Создал репозиторий на git.hub.
3. Клонировал репозиторий.

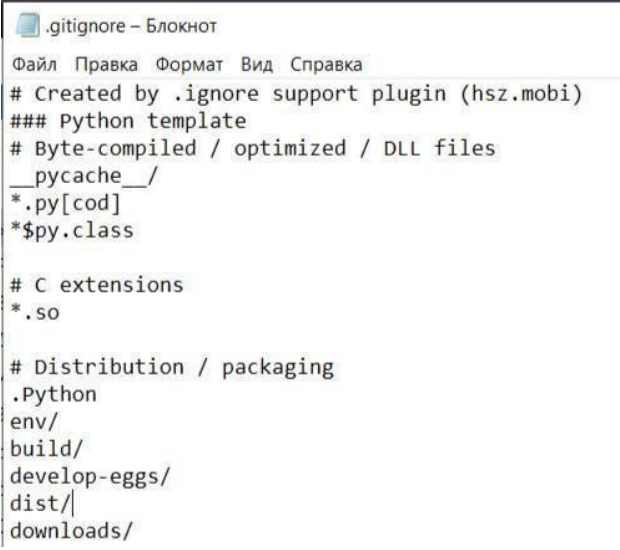
```

PS C:\> cd C:\Users\User\Desktop\учеба\Зсемак\эмй\lr_2.3
PS C:\Users\User\Desktop\учеба\Зсемак\эмй\lr_2.3> git clone https://github.com/KingItProgger/lr-2.3
Cloning into 'lr-2.3'...
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.
PS C:\Users\User\Desktop\учеба\Зсемак\эмй\lr_2.3>

```

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.

```

PS C:\Users\User\Desktop\учеба\Зсемак\эмй\lr_2.3\lr-2.3> git branch develop
PS C:\Users\User\Desktop\учеба\Зсемак\эмй\lr_2.3\lr-2.3> git checkout -b dev

```

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.

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
if __name__ == '__main__':
    s = input("Введите предложение: ")
    r = s.replace(' ', '_')
    print(f"Предложение после замены: {r}")
```

Рисунок 5 – пример 1

```
C:\Users\User\AppData\Local\Programs\Python
Введите предложение: g g g
Предложение после замены: g_g_g

Process finished with exit code 0
```

Рисунок 6 – примеры выполнения для примера 1

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
if __name__ == '__main__':
    word = input("Введите слово: ")
    idx = len(word) // 2
    if len(word) % 2 == 1:
        # Длина слова нечетная.
        r = word[:idx] + word[idx+1:]
    else:
        # Длина слова четная.
        r = word[:idx-1] + word[idx+1:]
    print(r)
```

Рисунок 7 – пример 2

```
C:\Users\User\AppData\Local\Programs\Py
Введите слово: dashboard
dashoard

Process finished with exit code 0
```

Рисунок 8 – примеры выполнения для 2 примера

```

import sys
if __name__ == '__main__':
    s = input("Введите предложение: ")
    n = int(input("Введите длину: "))
    # Проверить требуемую длину.
    if len(s) >= n:
        print(
            "Заданная длина должна быть больше длины предложения",
            file=sys.stderr
        )
        exit(1)
    # Разделить предложение на слова.
    words = s.split(' ')
    # Проверить количество слов в предложении.
    if len(words) < 2:
        print(
            "Предложение должно содержать несколько слов",
            file=sys.stderr
        )
        exit(1)
    # Количество пробелов для добавления.
    delta = n
    for word in words:
        delta -= len(word)
    # Количество пробелов на каждое слово.
    w, r = delta // (len(words) - 1), delta % (len(words) - 1)
    # Сформировать список для хранения слов и пробелов.
    lst = []
    # Пронумеровать все слова в списке и перебрать их.
    for i, word in enumerate(words):
        lst.append(word)
        # Если слово не является последним, добавить пробелы.
        if i < len(words) - 1:
            # Определить количество пробелов.
            width = w
            if r > 0:
                width += 1
                r -= 1
            # Добавить заданное количество пробелов в список.
            if width > 0:
                lst.append(' ' * width)
    # Вывести новое предложение, объединив все элементы списка lst.
    print(''.join(lst))

```

Рисунок 9 – пример 3

```
C:\Users\User\AppData\Local\Programs\Python\Pyt
Введите предложение: i am student of NCFU
Введите длину: 25
i am student of NCFU

Process finished with exit code 0
```

Рисунок 10 – примеры выполнения для примера 3

7. 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: ')
    l=len(word)
    res="*" * l
    print(res+word+res)
```

Рисунок 11 – решение задания 1

```
C:\Users\User\AppData\Local\Programs\
enter word: student
*****student*****

Process finished with exit code 0
```

Рисунок 12– результат выполнения задания 1

8. Даны два слова. Определить, сколько начальных букв первого слов 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 word1: '))
    word2 = (input('enter word2: '))
    if word1==word2:
        print('The same words!')
    else:
        if len(word1)<len(word2):
            minimal=len(word1)
        else:
            minimal=len(word2)
        count=0
        for i in range(minimal):
            if word1[i]==word2[i]:
                count+=1
            else: break
        if count==0:
            print('no same start symbols')
        else:
            print(f"{count} same start symbols")

```

Рисунок 13 – решение задания 2

```

C:\Users\User\AppData\Local\Programs
enter word1: yaroslav
enter word2: yaroslav_gurtovoy
8 same start symbols

Process finished with exit code 0

```

Рисунок 14 – результат выполнения задания 2

9. 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__':
    s=input('enter sentence: ')
    r=s.lower()
    while "c" in r:
        t=r.find('c')
        r=r[:t]+r[t+1:]

    while "c" in r:
        t=r.find('c')
        r=r[:t]+r[t+1:]

    print(r)
```

Рисунок 15 – решение задания 3

```
C:\Users\User\AppData\Local\Programs\Python\Python39\python.exe
enter sentence: counter counter
ounter ounter

Process finished with exit code 0
```

```
C:\Users\User\AppData\Local\Programs\Python\Python39\python.exe
enter sentence: Ставрополь
таврополь

Process finished with exit code 0
```

Рисунок 16 – результат выполнения задания 3

10. 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 1st word: ')
    word2 = input('enter 2nd word: ')
    for i in word1:
        if i in word2:
            print('yes')
        else:
            print('no')
```

Рисунок 17 – решение задания повышенной сложности

```
C:\Users\User\AppData\Local\Program
enter 1st word: height
enter 2nd word: width
yes
no
yes
no
yes
yes

Process finished with exit code 0
|
```

Рисунок 18 – результат выполнения задания повышенной сложности

Зафиксировал все изменения в github в ветке develop.

```
PS C:\Users\User\Desktop\учеба\Зсемак\змій\lr_2.3\lr-2.3> git commit -m [main 8954795] лаба
12 files changed, 145 insertions(+)
create mode 100644 pycharm/.idea/.gitignore
create mode 100644 pycharm/.idea/inspectionProfiles/profiles_settings.xml
create mode 100644 pycharm/.idea/lr 2.3.iml
create mode 100644 pycharm/.idea/misc.xml
create mode 100644 pycharm/.idea/modules.xml
create mode 100644 pycharm/ex1.py
create mode 100644 pycharm/ex2.py
create mode 100644 pycharm/ex3.py
create mode 100644 pycharm/task 1.py
create mode 100644 pycharm/task2.py
create mode 100644 pycharm/task3.py
create mode 100644 pycharm/task_super.py
PS C:\Users\User\Desktop\учеба\Зсемак\змій\lr_2.3\lr-2.3> git push
Enumerating objects: 18, done.
Counting objects: 100% (18/18), done.
Delta compression using up to 20 threads
Compressing objects: 100% (15/15), done.
Writing objects: 100% (17/17), 3.27 KiB | 3.27 MiB/s, done.
Total 17 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/KingItProgger/lr-2.3.git
1cbe633..8954795 main -> main
PS C:\Users\User\Desktop\учеба\Зсемак\змій\lr_2.3\lr-2.3>
```

Рисунок 19 – фиксация изменений в ветку

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 “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.

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.

8. 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, 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`. Когда нам нужно найти последнее вхождение определенной подстроки в строке, `rfind()` позволяет это сделать без необходимости итерации с конца строки.

