

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии
Департамент цифровых, робототехнических систем и электроники

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №2.3
дисциплины «Искусственный интеллект в профессиональной сфере»

#

Выполнил:
Оганесов Артур Витальевич
3 курс, группа ЭНЭ-б-о-22-1,
11.03.04 «Электроника и
наноэлектроника», направленность
(профиль) «Промышленная
электроника», очная форма обучения

(подпись)

Проверил:
Воронкин Роман Александрович,
доцент

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2024 г.

Тема работы: работа со строками в языке Python.

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

Аппаратура и материалы: ПК, операционная система Windows 10, Git, браузер для доступа к web-сервису GitHub, PyCharm Community Edition.

Ход работы:

1. Изучил теоретический материал работы.
2. На основе полученных знаний создал общедоступный репозиторий на GitHub, в котором использована лицензия MIT и выбранный мной язык программирования (python).

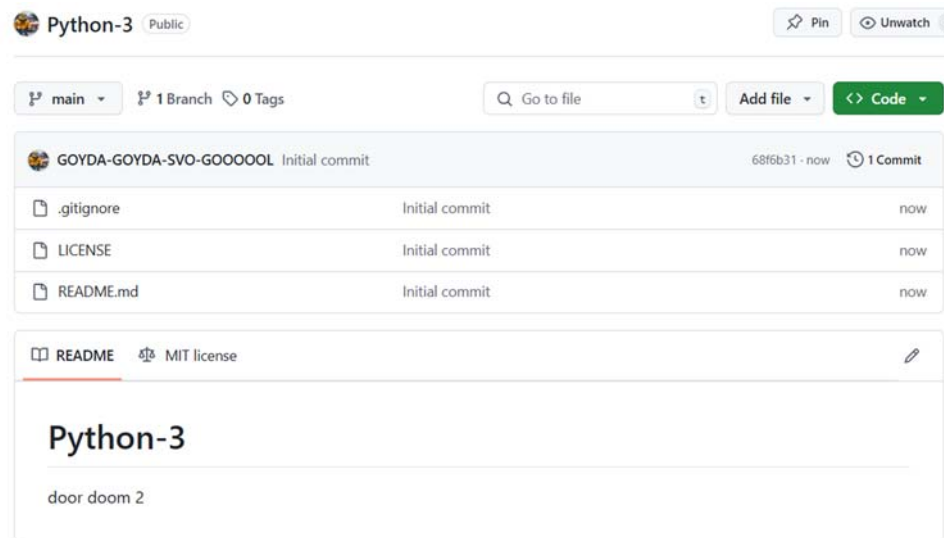
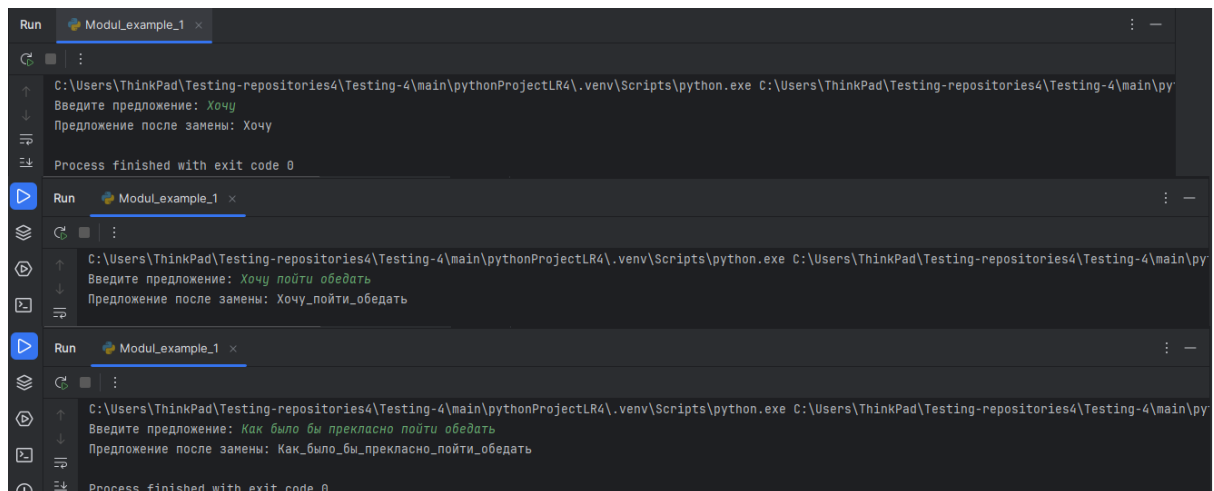


Рисунок 1 – Новый репозиторий

3. Привел скриншоты результатов выполнения каждой из программ примеров при различных исходных данных, вводимых с клавиатуры.



```
Run ModulExample_1 x
C:\Users\ThinkPad\Testing-repositories4\Testing-4\main\pythonProjectLR4\.venv\Scripts\python.exe C:\Users\ThinkPad\Testing-repositories4\Testing-4\main\py
Введите предложение: Хочу
Предложение после замены: Хочу
Process finished with exit code 0

Run ModulExample_1 x
C:\Users\ThinkPad\Testing-repositories4\Testing-4\main\pythonProjectLR4\.venv\Scripts\python.exe C:\Users\ThinkPad\Testing-repositories4\Testing-4\main\py
Введите предложение: Хочу пойти обедать
Предложение после замены: Хочу пойти обедать

Run ModulExample_1 x
C:\Users\ThinkPad\Testing-repositories4\Testing-4\main\pythonProjectLR4\.venv\Scripts\python.exe C:\Users\ThinkPad\Testing-repositories4\Testing-4\main\py
Введите предложение: Как было бы прекрасно пойти обедать
Предложение после замены: Как было бы прекрасно пойти обедать
Process finished with exit code 0
```

Рисунок 2 – Пример модуля 1



```
Run Alt+4 ModulExample_2 x
C:\Users\ThinkPad\Testing-repositories4\Testing-4\main\pythonProjectLR4\.venv\Scripts\python.exe C:\Users\ThinkPad\Testing-repositories4\Testing-4\main\py
Введите слово: YOU_WORK_?
YOU_WORK_?
Process finished with exit code 0

Run ModulExample_2 x
C:\Users\ThinkPad\Testing-repositories4\Testing-4\main\pythonProjectLR4\.venv\Scripts\python.exe C:\Users\ThinkPad\Testing-repositories4\Testing-4\main\py
Введите слово: нечет
нечет
Process finished with exit code 0
```

Рисунок 3 – Пример модуля 2



```
Run ModulExample_3 x
C:\Users\ThinkPad\Testing-repositories4\Testing-4\main\pythonProjectLR4\.venv\Scripts\python.exe C:\Users\ThinkPad\Testing-repositories4\Testing-4\main\py
Введите предложение: Ну заработай уже, бога ради
Введите длину: 33
Ну заработай уже, бога ради
|
Process finished with exit code 0

Run ModulExample_3 x
C:\Users\ThinkPad\Testing-repositories4\Testing-4\main\pythonProjectLR4\.venv\Scripts\python.exe C:\Users\ThinkPad\Testing-repositories4\Testing-4\main\py
Введите предложение: Ура, можно взять перерыв
Введите длину: 30
Ура, можно взять перерыв
Process finished with exit code 0
```

Рисунок 4 –Пример модуля 3

4. Выполнил индивидуальные задания, 10 вариант. Привел в отчете скриншоты работы программ.

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

Рисунок 5 – Задание 1

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

Рисунок 6 – Задание 2

10. Дано слово, оканчивающее символом «.». Вставить заданную букву после первой буквы и.

Рисунок 7 – Задание 3

```
Python Console>>>
>>>
>>> sentence = "искусственный интеллект - большая сила, способная решить множество бытовых
... letters = []
... for char in sentence:
...     if char == 'м' or char == 'н':
...         letters.append(char)
... print(letters)
['н', 'н', 'н', 'н', 'н', 'м', 'н', 'н', 'н', 'н', 'н', 'н', 'н']
```

Рисунок 8 – Решение 1

```
>>> words = ["жизненный", "шикарный", "ширинка", "зажыгалка", "жестяной", "жираф", "шыроченный"]
... def check_ji_shi(words):
...     incorrect_words = []
...     for word in words:
...         if "шы" in word or "жы" in word:
...             incorrect_words.append(word)
...     return incorrect_words
... incorrect_words = check_ji_shi(words)
... if incorrect_words:
...     print("Неправильно написанные слова:", incorrect_words)
... else:
...     print("Все слова написаны правильно.")
...
Неправильно написанные слова: ['ширинка', 'зажыгалка', 'шыроченный']
```

Рисунок 9 – Решение 2

```

...
... words = ["мир.", "сила.", "интеллект.", "разум.", "силовик."]
... def insert_letter_after_i(words, letter):
...     modified_words = []
...     for word in words:
...         index = word.find('и')
...         if index != -1 and index < len(word) - 1:
...             modified_word = word[:index + 1] + letter + word[index + 1:]
...             modified_words.append(modified_word)
...         else:
...             modified_words.append(word)
...     return modified_words
... letter_to_insert = 'x'
... modified_words = insert_letter_after_i(words, letter_to_insert)
... print(modified_words)
['михр.', 'сихла.', 'ихнтеллект.', 'разум.', 'сихловик.']

```

Рисунок 10 – Решение 3

Ответы на вопросы:

1. Что такое строки в языке Python?

Строки в языке Python — это упорядоченные последовательности символов, используемые для хранения и представления текстовой информации. Они позволяют описывать всё, что представлено в текстовой форме.

2. Какие существуют способы задания строковых литералов в языке Python?

Существуют следующие способы задания строковых литералов в языке Python: одинарные и двойные кавычки: строки, заключённые в одинарные или двойные кавычки, являются строковыми литералами; многострочные строки: можно использовать чёрную косую черту в конце каждой строки или тройные кавычки для создания многострочных строк.

3. Какие операции и функции существуют для строк?

Для строк в языках программирования существуют следующие операции и функции: вычисление длины строки; выделение подстроки (префикса, суффикса, произвольного фрагмента); инвертирование строки

(редко используется для текстовых строк, но встречается для числовых строк); конкатенация строк (соединение нескольких строк в одну); сравнение строк (обычно используется лексикографическое упорядочение); поиск образца (поиск подстроки в другой строке); подстановка (фрагмент строки заменяется другим значением); поиск максимального совпадения двух строк; нахождение редакционного расстояния (мера различия между двумя строками).

4. Как осуществляется индексирование строк?

Индексирование строк в Python осуществляется с помощью номеров индексов, которые начинаются с 0 и увеличиваются на 1 для каждого символа. Доступ к символам строки можно получить, указав индекс символа в квадратных скобках после имени переменной, хранящей строку. Например, для доступа.

5. Как осуществляется работа со срезами для строк?

Работа со срезами для строк в Python осуществляется следующим образом: Slicing с одним параметром: `S[start:end]` возвращает подстроку из символов, начинающихся с индекса `start` и заканчивая индексом `end`, не включая его. Например, `S[1:4] == 'ell'`. Slicing с двумя параметрами: `S[a:b]` возвращает подстроку из `b-a` символов, начиная с символа с индексом `a` и заканчивая символом с индексом `b`, не включая его. Например, `S[1:4] == 'ell'`. Отрицательные индексы: если указать отрицательное значение индекса, то номер будет отсчитываться с конца, начиная с номера -1. Например, `S[-1] == 'o'`, `S[-2] == 'l'`, `S[-3] == 'l'`, `S[-4] == 'e'`, `S[-5] == 'H'`.

6. Почему строки Python относятся к неизменяемому типу данных?

Строки Python относятся к неизменяемому типу данных, потому что их содержимое не может быть изменено после создания. Это сделано для повышения производительности и безопасности, так как предотвращает случайные ошибки и упрощает оптимизацию кода.

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

Если каждое слово в строке начинается с заглавной буквы, метод `istitle()` вернёт `True`, в противном случае — `False`.

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

Чтобы проверить строку на вхождение другой строки, используется метод `string.contains()`.

9. Как найти индекс первого вхождения подстроки в строку?

Чтобы найти индекс первого вхождения подстроки в строку, используется метод `string.indexOf()`.

10. Как подсчитать количество символов в строке?

Чтобы подсчитать количество символов в строке, используется метод `string.length()`.

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

Чтобы подсчитать количество раз, когда определённый символ встречается в строке, используется метод `string.count()`.

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

F-строки (f-strings) — это новый способ форматирования строк в Python, представленный в версии 3.6. Они позволяют встраивать выражения прямо в строку, используя фигурные скобки.

13. Как найти подстроку в заданной части строки?

Чтобы найти подстроку в заданной части строки, используется метод `string.find()`.

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

Пример: `msg = "Меня зовут {0}, мне {1} и я люблю Python".format(name, age)`

15. Как узнать о том, что в строке содержатся только цифры?

Чтобы узнать, содержит ли строка только цифры, используется метод `isdigit()`.

16. Как разделить строку по заданному символу?

Чтобы разделить строку по заданному символу, используется метод `split()`.

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

Чтобы проверить строку на то, что она составлена только из строчных букв, используется метод `islower()`:

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

Чтобы проверить, начинается ли строка с маленькой буквы, используется метод `startswith()`.

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

Да, в Python можно прибавить целое число к строке.

20. Как «перевернуть» строку?

Чтобы «перевернуть» строку, используется метод `reverse()`.

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

Чтобы объединить список строк в одну строку, элементы которой разделены дефисами, используется метод `join()`.

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

Чтобы привести всю строку к верхнему или нижнему регистру, используются методы `upper()` или `lower()`.

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

Чтобы преобразовать первый и последний символы строки к верхнему регистру, используются методы `capitalize()` или `rjust()`.

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

Чтобы проверить строку на то, что она составлена только из прописных букв, используется метод `isupper()`.

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

Метод `splitlines()` используется для разделения строки на отдельные строки по разделителям (например, новые строки). Можно бы воспользоваться этим методом, если нужно проанализировать текстовый файл или документ, содержащий несколько строк текста.

26. Как в заданной строке заменить на что-либо все вхождения некоей подстроки?

Чтобы заменить все вхождения подстроки в заданной строке, используется метод `replace()`.

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

Чтобы проверить, начинается ли строка с заданной последовательности символов, используется метод `startswith()`.

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

Чтобы узнать, содержит ли строка только пробелы, используется метод `isspace()` из библиотеки `string`.

29. Что случится, если умножить некую строку на 3?

Умножение строки на 3 приведёт к повторению исходной строки три раза.

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

Чтобы привести к верхнему регистру первый символ каждого слова в строке, используется метод `transform()`.

31. Как пользоваться методом `partition()`?

Метод `partition()` разбивает строку на подстроки на основе указанного разделителя и возвращает кортеж из разделённой строки, разделителя и оставшейся части строки.

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

Метод `rfind()` находит последнее вхождение подстроки в строке и возвращает позицию найденного вхождения.

Вывод: в ходе выполнения работы были получены навыки работы со строками при написании программ с помощью языка программирования Python версии 3.x.