

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

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №6
дисциплины «Основы программной инженерии»

Выполнил:
Мелтонян Одиссей Гарикович
2 курс, группа ПИЖ-б-о-22-1,
09.03.04 «Программная инженерия»,
очная форма обучения

(подпись)

Проверил: Воронкин Р.А.

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

Тема: Работа со строками в языке Python.

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

Ход работы:

1. Изучил теоретический материал работы.
2. Создал общедоступный репозиторий на GitHub, в котором использована лицензия MIT и язык программирования Python.

Create a new repository

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

Required fields are marked with an asterisk ().*

Owner * **Repository name ***

odik8 / BSE6

✔ BSE6 is available.

Great repository names are short and memorable. Need inspiration? How about **upgraded-octo-happiness** ?

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:

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

Add .gitignore

.gitignore template: **Python**

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: **MIT License**

Рисунок 1. – Создание репозитория

3. Выполнил клонирование созданного репозитория.

```
MINGW64:/c:/Users/varfe/OneDrive/Рабочий стол/Воронкин/ЛР6/bse6
varfe@DESKTOP-8SV1DU3 MINGW64 ~/OneDrive/Рабочий стол/Воронкин/ЛР6
$ git clone https://github.com/odik8/BSE6.git
Cloning into 'BSE6'...
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
Receiving objects: 100% (4/4), done.
```

Рисунок 2. – Клонирование репозитория

4. Дополнил файл .gitignore необходимыми правилами для работы с IDE PyCharm.

```
Code Blame 160 lines (131 loc) · 3.01 KB Code 55% faster with GitHub Copilot

1  # Byte-compiled / optimized / DLL files
2  __pycache__/
3  *.py[cod]
4  *$py.class
5
6  # C extensions
7  *.so
8
9  # Distribution / packaging
10 .Python
11 build/
12 develop-eggs/
13 dist/
14 downloads/
15 eggs/
16 .eggs/
17 lib/
18 lib64/
19 parts/
20 sdist/
21 var/
22 wheels/
23 share/python-wheels/
24 *.egg-info/
```

Рисунок 3. – Файл .gitignore

5. Организовал свой репозиторий в соответствии с моделью ветвления git-flow. Для этого создал ветку develop, в которую будут сливаться ветки features.

```
varfe@DESKTOP-8SV1DU3 MINGW64 ~/OneDrive/Рабочий стол/Воронкин/ЛР6/bse6 (main)
$ git checkout -b develop
Switched to a new branch 'develop'
```

Рисунок 4. – Создание ветки develop

6. Создал проект PyCharm в папке репозитория.

7. Выполнил индивидуальные задания, согласно своему варианту.

Вариант 11. Задание 1. Дано предложение. Составить программу, которая выводит все вхождения в предложение двух заданных символов.

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3  usage new *
4  def main():
5      sentence = str(input("Enter the sentence: "))
6      symbol1 = str(input("Enter first symbol to find: "))
7      symbol2 = str(input("Enter second symbol to find: "))
8
9      return f"Count of '{symbol1}' symbol and '{symbol2}' symbol is {sentence.count(symbol1) + sentence.count(symbol2)}"
10
11 if __name__ == "__main__":
12     print(main())
```

Рисунок 4 – Код решения

```
C:\Users\varfe\AppData\Local\Programs\Python\Python312\pytho
Enter the sentence: Kawasaki Cago Cricko Estriper
Enter first symbol to find: a
Enter second symbol to find: e
Count of 'a' symbol and 'e' symbol is 5

Process finished with exit code 0
```

Рисунок 5 – Результат выполнения кода

Задание 2. Дана последовательность слов. Проверить, правильно ли в ней записаны буквосочетания ча и ща. Исправить ошибки.

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  usage new *
5
6  def check_words(sequence):
7      correct_words = []
8      incorrect_words = []
9
10     for word in sequence:
11         if "ча" in word or "ща" in word:
12             correct_words.append(word)
13         else:
14             incorrect_words.append(word)
15
16     print("\nСлова с ошибкой в записи буквосочетания \"ча\" или \"ща\":")
17     for word in incorrect_words:
18         print(f' "{word}"')
19
20 if __name__ == "__main__":
21     word_sequence = [
22         "Чай", "Чайник", "Чашка", "Чаша", "Роща", "Щавель",
23         "Выручай", "Гуща", "Часто", "Час", "Часы", "Часовой",
24         "Чайка", "Туча", "Чугун", "Хочу", "Часть", "Чуб",
25         "Чудо", "Учу", "Щука", "Щуплый"
26     ]
27
28     check_words(word_sequence)
```

Рисунок 6 – Код решения задачи

```

C:\Users\varfe\AppData\Local\Programs\Python\Python312\python

Слова с ошибкой в записи буквосочетания "ча" или "ща":
    "чяй"
    "чящя"
    "щявель"
    "гущя"
    "чясы"
    "тучя"

Исправленные слова:
    "чай"
    "чаща"
    "щавель"
    "гуща"
    "часы"
    "туча"

```

Рисунок 7 – Результат выполнения кода

Задание 3. Дано предложение, оканчивающееся символом «.». Вставить заданную букву перед последней буквой и «.»

```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  1 usage new *
5  def paste_the_char():
6      sentence = "Тестовое предложения для вставки вашей буквы"
7      index_of_i = sentence.rfind('и')
8      char = str(input("Введите символ: "))
9      return sentence[:index_of_i] + char + sentence[index_of_i:]
10
11  if __name__ == "__main__":
12      print(paste_the_char())
13

```

Рисунок 8 – Код решения задачи

```
C:\Users\varfe\AppData\Local\Programs\Python\Python
Введите символ: 5
Тестовое предложение для вставки вашей буквы

Process finished with exit code 0
|
```

Рисунок 9 – Результат выполнения кода

Задание повышенной сложности:

Задание 1. Даны три слова. Напечатать их общие буквы. Повторяющиеся буквы каждого слова не рассматривать.

```
1  #!/usr/bin/env python3
2
3  1 usage new *
4  def common_letters(word1, word2, word3):
5      set1 = set(word1)
6      set2 = set(word2)
7      set3 = set(word3)
8
9      common_letters_set = set1.intersection(*set2, set3)
10
11      print("Общие буквы:", ", ".join(common_letters_set))
12
13  if __name__ == "__main__":
14      common_letters(word1: "1Kawazaki", word2: "1iCago", word3: "1Kricko")
15
```

Рисунок 10 – Код решения задачи

```
C:\Users\varfe\AppData\Local\Programs\Python\Python
Общие буквы: 1, i

Process finished with exit code 0
```

Рисунок 11 – Код решения

8. Зафиксировал сделанные изменения в репозитории.
9. Добавил отчет по лабораторной работе в формате PDF в папку doc репозитория. Зафиксировал изменения.
10. Выполнил слияние ветки для разработки с веткой main.
11. Отправил сделанные изменения на сервер GitHub.
12. Отправил адрес репозитория GitHub на электронный адрес преподавателя

Вопросы для защиты:

1. **Что такое строки в языке Python?** – Строки в языке Python представляют собой последовательности символов и используются для хранения текстовой информации.
2. **Какие существуют способы задания строковых литералов в языке Python?** – Строковые литералы можно задавать с использованием одинарных ('), двойных (") и тройных ('' или ''') кавычек.
3. **Какие операции и функции существуют для строк?** – Для строк существуют различные операции, такие как конкатенация (+), умножение (*), разделение (`split()`), обрезка (`strip()`) и многие другие.
4. **Как осуществляется индексирование строк?** – Индексирование строк в Python начинается с 0, и элементы строки можно получить с использованием квадратных скобок, например, `my_string[0]` вернет первый символ строки.
5. **Как осуществляется работа со срезами для строк?** – Срезы в Python задаются с использованием двоеточия внутри квадратных скобок, например, `my_string[1:4]` вернет подстроку с индексами от 1 до 3.

6. Почему строки Python относятся к неизменяемому типу данных?

– Строки являются неизменяемыми в Python, потому что после создания строки нельзя изменить ее содержимое, только создать новую строку.

7. Как проверить то, что каждое слово в строке начинается с заглавной буквы? – Можно воспользоваться методом `istitle()`, который возвращает `True`, если каждое слово начинается с заглавной буквы.

8. Как проверить строку на вхождение в неё другой строки? – Можно использовать оператор `in`, например, `substring in my_string`.

9. Как найти индекс первого вхождения подстроки в строку? – Метод `find()` возвращает индекс первого вхождения подстроки, а `-1`, если подстрока не найдена.

10. Как подсчитать количество символов в строке? – Можно воспользоваться функцией `len()`, например, `len(my_string)` вернет количество символов в строке.

11. Как подсчитать то, сколько раз определённый символ встречается в строке? – Можно воспользоваться методом `count()`, например, `my_string.count('a')` вернет количество вхождений символа 'a'.

12. Что такое f-строки и как ими пользоваться? – F-строки (f-strings) позволяют вставлять значения переменных в строки. Используются с префиксом `f` или `F`, например, `f"Привет, {name}"`.

13. Как найти подстроку в заданной части строки? – Можно использовать метод `find()` с указанием диапазона индексов.

14. Как вставить содержимое переменной в строку, воспользовавшись методом format()? – Можно использовать `{}` в строке и метод `format()`, например, `"Привет, {}".format(name)`.

15. **Как узнать о том, что в строке содержатся только цифры?** – Можно воспользоваться методом **isdigit()**, который возвращает True, если все символы строки являются цифрами.

16. **Как разделить строку по заданному символу?** – Метод **split()** разделит строку на подстроки по указанному символу.

17. **Как проверить строку на то, что она составлена только из строчных букв?** – Метод **islower()** возвращает True, если все буквы в строке строчные.

18. **Как проверить то, что строка начинается со строчной буквы?** – Метод **islower()** также может быть использован для проверки, но вернет True только если все символы строчные.

19. **Можно ли в Python прибавить целое число к строке?** – Нет, строки и числа в Python являются разными типами данных. Операция "+" для них не определена.

20. **Как «перевернуть» строку?** – Можно воспользоваться срезами, например, **my_string[::-1]** вернет строку в обратном порядке.

21. **Как объединить список строк в одну строку, элементы которой разделены дефисами?** – Можно использовать метод **join()**, например, **' '.join(my_list)**.

22. **Как привести всю строку к верхнему или нижнему регистру?** – Методы **upper()** и **lower()** соответственно приводят всю строку к верхнему или нижнему регистру.

23. **Как преобразовать первый и последний символы строки к верхнему регистру?** – Можно воспользоваться методами **capitalize()** для первого символа и **title()** для каждого слова.

24. **Как проверить строку на то, что она составлена только из прописных букв?** – Метод **isupper()** возвращает True, если все буквы в строке прописные.

25. **В какой ситуации вы воспользовались бы методом splitlines()?–** Метод **splitlines()** используется для разделения строки на список строк по символам новой строки (**\n**).

26. **Как в заданной строке заменить на что-либо все вхождения некоей подстроки?** – Метод **replace()** позволяет заменить все вхождения подстроки на другую строку.

27. **Как проверить то, что строка начинается с заданной последовательности символов, или заканчивается заданной последовательностью символов?** – Методы **startswith()** и **endswith()** соответственно проверяют начало и конец строки.

28. **Как узнать о том, что строка включает в себя только пробелы?** – Метод **isspace()** возвращает True, если строка состоит только из пробельных символов.

29. **Что случится, если умножить некую строку на 3?** – Строка будет повторена три раза, например, **'abc' * 3** вернет **'abcabcabc'**.

30. **Как привести к верхнему регистру первый символ каждого слова в строке?** – Можно воспользоваться методом **title()**.

31. **Как пользоваться методом partition()?–** Метод **partition()** разбивает строку на три части по первому вхождению указанной подстроки, возвращая кортеж из трех элементов.

32. **В каких ситуациях пользуются методом rfind()?–** Метод **rfind()** используется для поиска последнего вхождения подстроки в строке, возвращая индекс этого вхождения.