

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

Институт цифрового развития
Кафедра инфокоммуникаций

«Работа со строками в языке Python»

ОТЧЕТ
по лабораторной работе №6
дисциплины
«Основы программной инженерии»

Выполнил:

Мизин Глеб Егорович

2 курс, группа ПИЖ-б-о-21-1,

09.03.04 «Программная

инженерия», направленность

(профиль) «Разработка и

сопровождение программного

обеспечения», очная форма

обучения

(подпись)

Проверил:

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2022 г.

Проработка примеров из лабораторной работы:

```
1 ▶ #!/usr/bin/env python3
2   # -*- coding: utf-8 -*-
3
4 ▶ if __name__ == '__main__':
5     s = input("Введите предложение: ")
6     r = s.replace(' ', '_')
7     print(f"Предложение после замены: {r}")

if __name__ == '__main__':
Run: FirstExample x
F:\GitLaby\Lab2.3\PythonLab2.3-GitFlow\venv\Scripts\python.exe F:\GitLaby\Lab2.3\Py
Введите предложение: Дано предложение Все пробелы в нем заменить символом "_"
Предложение после замены: Дано_предложение_Все_пробелы_в_нем_заменить_символом_"_"
Process finished with exit code 0
```

Рисунок 1 – Код и результат работы программы примера №1

```
1 ▶ #!/usr/bin/env python3
2   # -*- coding: utf-8 -*-
3 ▶ if __name__ == '__main__':
4     word = input("Введите слово: ")
5
6     idx = len(word) // 2
7     if len(word) % 2 == 1:
8         # Длина слова нечетная.
9         r = word[:idx] + word[idx+1:]
10    else:
11        # Длина слова четная.
12        r = word[:idx-1] + word[idx+1:]
13
14    print(r)

if __name__ == '__main__':
SecondExample x
F:\GitLaby\Lab2.3\PythonLab2.3-GitFlow\venv
Введите слово: Программа
Программа
```

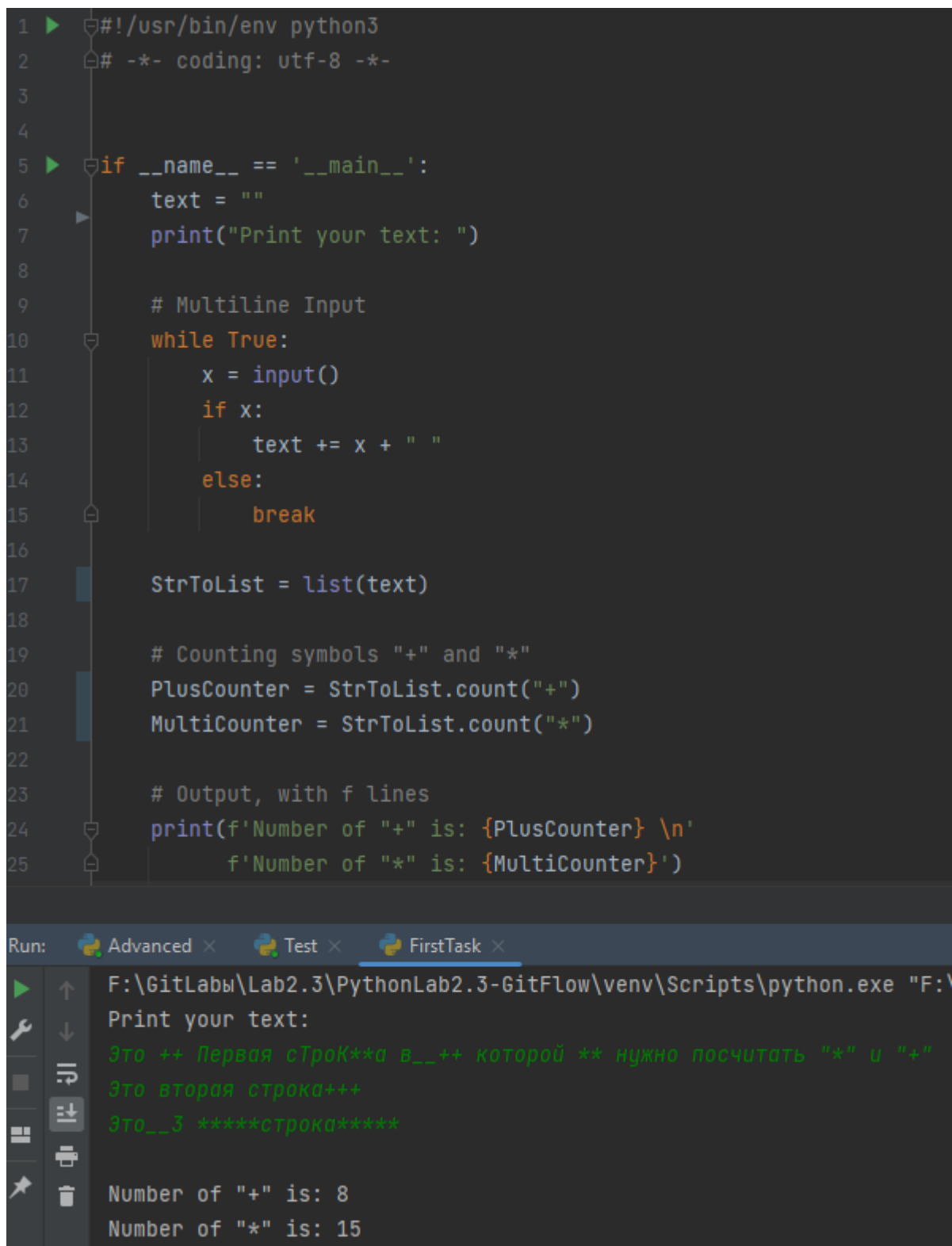
Рисунок №2 – Код и результат работы программы примера №2

```
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 import sys
5
6 ▶ if __name__ == '__main__':
7     s = input("Введите предложение: ")
8     n = int(input("Введите длину: "))
9
10    # Проверить требуемую длину.
11    if len(s) >= n:
12        print(
13            "Заданная длина должна быть больше длины предложения",
14            file=sys.stderr
15        )
16        exit(1)
17
18    # Разделить предложение на слова.
19    words = s.split(' ')
20    # Проверить количество слов в предложении.
21    if len(words) < 2:
22        if __name__ == '__main__':
23            ThirdExample ×
24            F:\GitLab\Lab2.3\PythonLab2.3-GitFlow\venv\Scripts\python.exe F:\G
25            Введите предложение: Тест примера номер 3
26            Введите длину: 20
27            Тест примера номер 3
```

Рисунок 3 – Код и результат работы программы примера №3

Индивидуальные задания:

Задание №1: дан текст. Сколько раз в нем встречается символ «+» и сколько раз символ «*»?



```
1  ▶  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  ▶  if __name__ == '__main__':
6      text = ""
7      print("Print your text: ")
8
9      # Multiline Input
10     while True:
11         x = input()
12         if x:
13             text += x + " "
14         else:
15             break
16
17     StrToList = list(text)
18
19     # Counting symbols "+" and "*"
20     PlusCounter = StrToList.count("+")
21     MultiCounter = StrToList.count("*")
22
23     # Output, with f lines
24     print(f'Number of "+" is: {PlusCounter} \n'
25           f'Number of "*" is: {MultiCounter}')
```

Run: Advanced × Test × FirstTask ×

F:\GitLab\Lab2.3\PythonLab2.3-GitFlow\venv\Scripts\python.exe "F:\GitLab\Lab2.3\PythonLab2.3-GitFlow\venv\Scripts\python.exe" "F:\GitLab\Lab2.3\PythonLab2.3-GitFlow\venv\Scripts\python.exe"

Print your text:

Это ++ Первая строка**а в__++ которой ** нужно посчитать "*" и "+"

Это вторая строка+++

Это__3 *****строка*****

Number of "+" is: 8

Number of "*" is: 15

Рисунок 4 – Код и результат работы программы для задания №1

Задание №2: дано предложение. Заменить в нем все вхождения буквосочетания про на нет.

```
1  ▶  #!/usr/bin/env python3
2      # -*- coding: utf-8 -*-
3
4
5  ▶  if __name__ == '__main__':
6      Sentence = str(input("Enter your sentence: "))
7      Sen_List = list(Sentence)
8
9      for i in range(len(Sentence)-2):
10
11         # Checking 3 symbols one by one
12         if Sen_List[i] == "н" and Sen_List[i+1] == "п" and Sen_List[i+2] == "о":
13             Sen_List[i] = "н"
14             Sen_List[i+1] = "е"
15             Sen_List[i+2] = "т"
16
17         # Connecting parts of list to string
18         NewSentence = "".join(Sen_List)
19         print(NewSentence)
20
if __name__ == '__main__':
SecondTask x
F:\GitLaby\Lab2.3\PythonLab2.3-GitFlow\venv\Scripts\python.exe "F:\GitLaby\Lab2.3\Py
Enter your sentence: He про а противник
He нет а неттивник
```

Рисунок 5 – Код и результат работы программы для задания №2

Задание №3: Дано ошибочно написанное слово роцессорп. Путем перемещения его букв получить слово процессор.

```
1  ▶  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  ▶  if __name__ == '__main__':
6      WrongWord = 'роцессорп'
7
8      # Take last 3 symbols
9      LastPart = WrongWord[-3:]
10
11     # Revers of received characters
12     LastPartR = LastPart[::-1]
13
14     # Printing 3 symbols and right part of the word without last symbol
15     print(f'Right word: {LastPartR}{WrongWord[2:-1]}')
16
```

ThirdTask ×

↑ F:\GitLaby\Lab2.3\PythonLab2.3-GitFlow\venv\Scripts\python.exe "F:\GitLaby\
↓ Right word: процессор
⏏ Process finished with exit code 0

Рисунок 6 – Код и результат работы программы для задания №3

Задание повышенной сложности: дано предложение. Найти длину его самого короткого слова.

```
1  ▶ #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  ▶ if __name__ == '__main__':
6      MinWordLen = float('inf')
7
8      # List of punctuation marks
9      PunctMarks = [' ', ',', '-', ';', ':', '?', '!', '.']
10     Counter = 0
11
12     UserStr = str(input("Enter your sentence: "))
13     StrList = list(UserStr)
14
15     # Loop through all characters of the input string
16     for i in range(len(StrList)):
17
18         # Checking if a symbol is a punctuation mark
19         if StrList[i] in PunctMarks:
20             WordLen = Counter
21             Counter = 0
22
23         # Determine the minimum length of a word
24         if 0 < WordLen < MinWordLen:
25             MinWordLen = WordLen
26         else:
27             Counter += 1
28
29     # Checking the last word, in case of a forgotten dot at the end
30     WordLen = Counter
31     if 0 < WordLen < MinWordLen:
32         MinWordLen = WordLen
33
34     print(MinWordLen)
35
```

Рисунок 7.1 – Код программы для задания повышенной сложности

```
↑ F:\GitLab\Lab2.3\PythonLab2.3-GitFlow\venv\Scripts\python.exe
↓ Enter your sentence: Какая, длина !самого! короткого... слова?
↕ 5
⏮ Process finished with exit code 0
⏭
```

Рисунок 7.2 – Результат работы программы для задания повышенной сложности

Контрольные вопросы

1. Что такое строки в языке Python?

Строки в Python - упорядоченные последовательности символов, используемые для хранения и представления текстовой информации.

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

Строки в апострофах и в кавычках, экранированные последовательности - служебные символы, "Сырые" строки - подавляют экранирование, строки в тройных апострофах или кавычках

3. Какие операции и функции существуют для строк?

Оператор сложения строк +, оператор умножения строк *, оператор принадлежности подстроки in.

Функция ord(c) возвращает числовое значение для заданного символа, Функция chr(n) возвращает символьное значение для данного целого числа, Функция len(s) возвращает длину строки, функция str(obj) возвращает строковое представление объекта.

4. Как осуществляется индексирование строк?

```
>>> s = 'foobar'
>>> s[0]='f'
>>> s[1]='o'
>>> s[3]='b'
>>> s[5]='r'
```

5. Как осуществляется работа со срезами для строк?

```
>>> s = 'python'
>>> s[2:5]
'tho'
```


6. Почему строки Python относятся к неизменяемому типу данных?

Строки в Python относятся к неизменяемому типу данных т.к при смене любого элемента строки, изменяется идентификатор

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

`str. istitle()` возвращает `True` , если каждое слово в строке `str` начинается с заглавной буквы и в ней есть хотя бы один символ в верхнем регистре. Возвращает `False` в противном случае.

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

Для проверки, содержится ли указанная строка в другой строке, в Python можно использовать оператор `in` или метод `find`.

9. Как найти индекс первого вхождения подстроки в строку?

Метод `find()` помогает найти индекс первого совпадения подстроки в данной строке. Возвращает `-1`, если подстрока не была найдена. В метод передаются три параметра: подстрока, которую нужно найти, `start` со значением по умолчанию равным `0` и `end` со значением по умолчанию равным длине строки.

10. Как подсчитать количество символов в строке?

Узнать количество символов (длину строки) можно при помощи функции `len`.

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

Метод `count()`, возвращает количество вхождений в строку заданного символа.

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

f-строки. Способ похожий на `format()`, но более гибкий и читабельный. Они поддерживают расширенное форматирование чисел, могут форматировать дату без метода `strftime()`, поддерживают базовые арифметические операции прямо в строках, позволяют обращаться к значениям списков по индексам, к элементам словарей по ключу, вызывать функции и методы объектов.

13. Как найти подстроку в заданной части строки?

Метод `index()` можно вызывать, передавая ему необязательные аргументы, представляющие индекс начального и конечного фрагмента строки, в пределах которых и нужно осуществлять поиск подстроки.

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

Метод `format()` позволяет добиваться результатов, сходных с теми, которые можно получить, применяя f-строки. Но, использовать `format()` не так удобно, так как все переменные приходится указывать в качестве аргументов `format()`.

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

Существует метод `isnumeric()`, который возвращает `True` в том случае, если все символы, входящие в строку, являются цифрами.

16. Как разделить строку по заданному символу?

Методом `split()`, который разбивает строку по заданному символу или по нескольким символам.

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

Метод `islower()` возвращает `True` только в том случае, если строка составлена исключительно из строчных букв.

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

Сделать это можно, вызвав метод `islower()` для первого символа строки.

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

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

20. Как «перевернуть» строку?

Для того чтобы «перевернуть» строку, её можно разбить, представив в виде списка символов, «перевернуть» список, и, объединив его элементы, сформировать новую строку. `"".join(reversed("hello world"))`

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

Метод `join()` умеет объединять элементы списков в строки, разделяя отдельные строки с использованием заданного символа

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

Для решения этих задач можно воспользоваться методами `upper()` и `lower()`, которые, соответственно, приводят все символы строк к верхнему и нижнему регистрам.

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

Мы будем обращаться к символам строки по индексам. Строки в Python иммутабельны, поэтому мы будем заниматься сборкой новой строки на основе существующей.

```
animal = 'fish'
```

```
animal[0].upper() + animal[1:-1] + animal[-1].upper()
```

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

Имеется метод `isupper()`, возвращает `True` только в том случае, если вся строка состоит из прописных букв.

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

Когда необходимо разделить строки по символам разрыва строки.

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

Для решения этой задачи можно воспользоваться методом `replace()`

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

Для ответа на этот вопрос можно прибегнуть к методам `Startswith()` и `Endswith()`

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

Есть метод `isspace()`, который возвращает `True` только в том случае, если строка состоит исключительно из пробелов.

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

Строка повторится трижды

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

Существует метод `title()`, приводящий к верхнему регистру первую букву каждого слова в строке.

31. Как пользоваться методом `partition()` ?

Метод `partition()` разбивает строку по заданной подстроке. После этого результат возвращается в виде кортежа. При этом подстрока, по которой осуществлялась разбивка, тоже входит в кортеж.

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

Метод `rfind()` похож на метод `find()`, но он, в отличие от `find()`, просматривает строку не слева направо, а справа налево, возвращая индекс первого найденного вхождения искомой подстроки.