**МИНИCTEPCTBO НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное учреждение высшего образования**

**«Северо-Кавказский федеральный университет» Кафедра инфокоммуникаций**

**Отчет по лабораторной работе №6**

**по дисциплине «Основы программной инженерии»**

|  |
| --- |
| Выполнил студент группы ПИЖ-б-о-20-1 |
| Ваньянц И.М. « » 20 г. |
| Подпись студента |
| Работа защищена « » \_20 г. |
| Проверил Воронкин Р.А.  (подпись) |

Ставрополь 2021

**ХОД РАБОТЫ**



Рисунок 1 – создание ветки “develop”

1. Пример 1

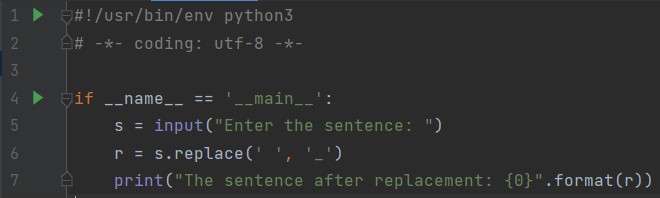


Рисунок 2 – код примера



Рисунок 3 – результат

1. Пример 2

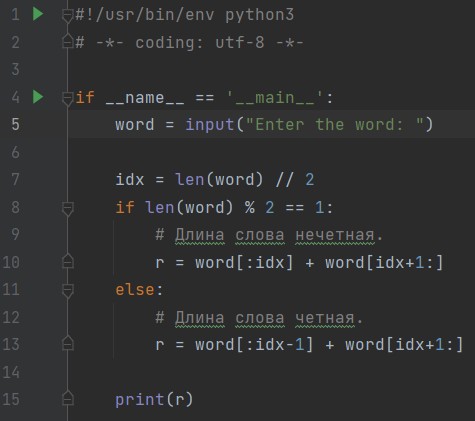


Рисунок 4 – код примера

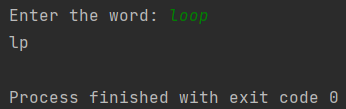


Рисунок 5 - вывод программы при четной длине слова

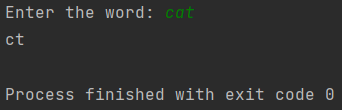
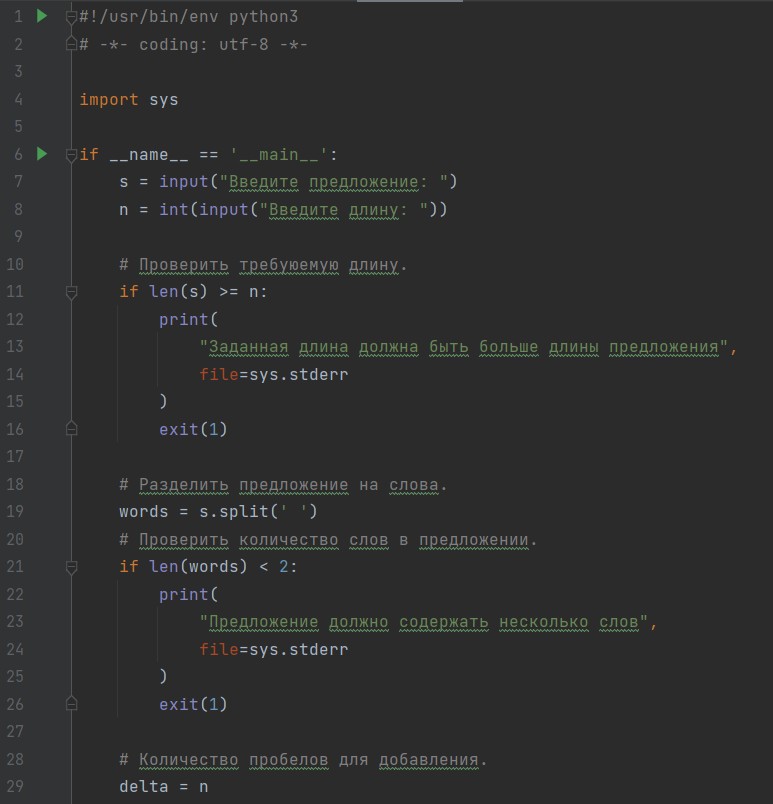
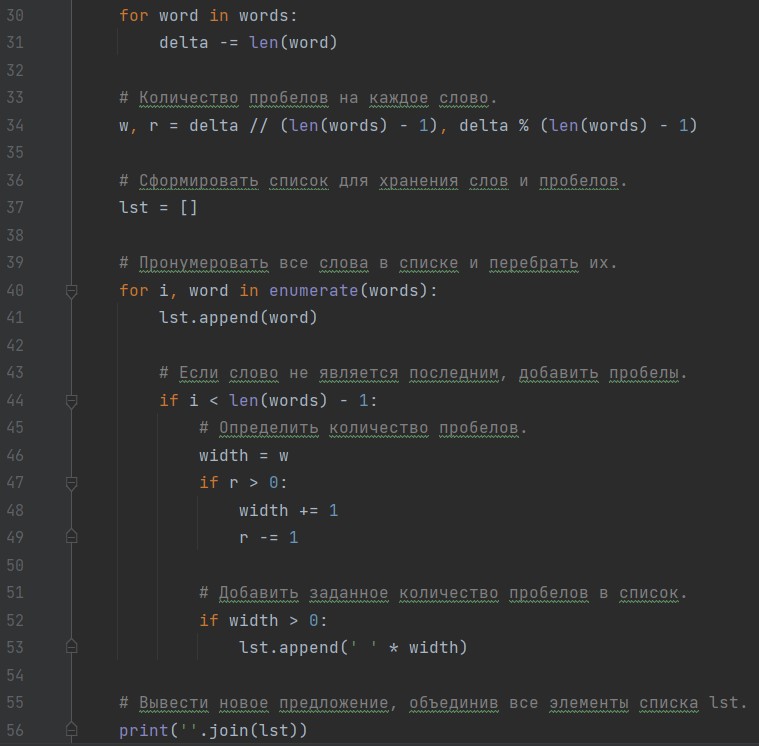


Рисунок 6 - вывод программы при нечетной длине слова

1. Пример 3.





Рисунки 7 и 8 – код программы

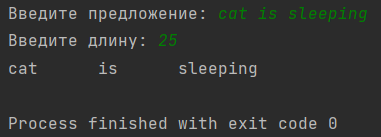


Рисунок 9 – вывод при верном вводе

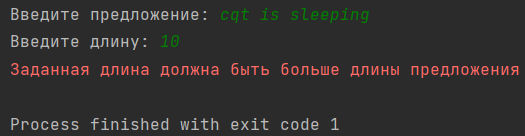


Рисунок 10 – вывод при неправильном вводе

1. Индивидуальное задание 1

Дано слово. Добавить к нему в начале и конце столько звездочек, сколько букв в этом слове.

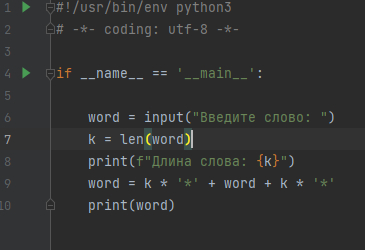


Рисунок 11 – код программы

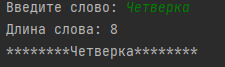


Рисунок 12 – результат

1. Индивидуальное задание 2

Даны два слова. Определить, сколько начальных букв первого слова совпадает с начальными буквами второго слова. Рассмотреть два случая:

известно, что слова разные;

слова могут быть одинаковыми.

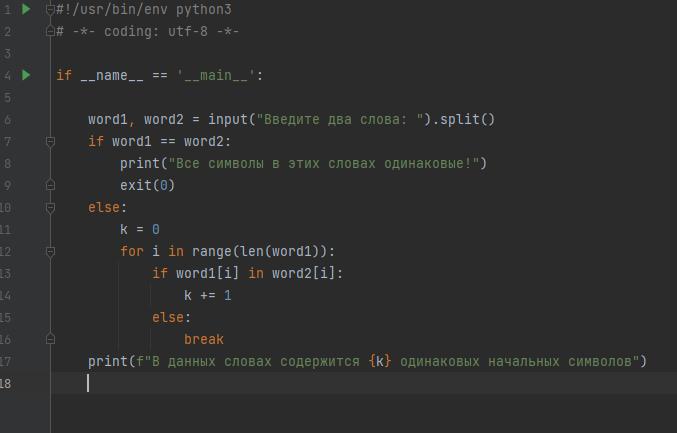


Рисунок 13 – код программы



Рисунок 14 – результат

1. Индивидуальное задание 3

Дано предложение. Удалить из него все буквы с (как в кириллице так и на латинице).

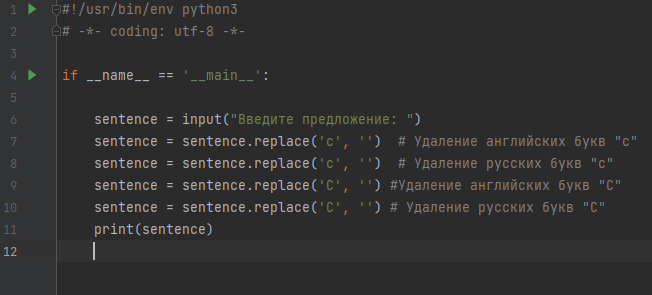


Рисунок 15 – код программы



Рисунок 16 – результат программы

1. Задание повышенной сложности

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.

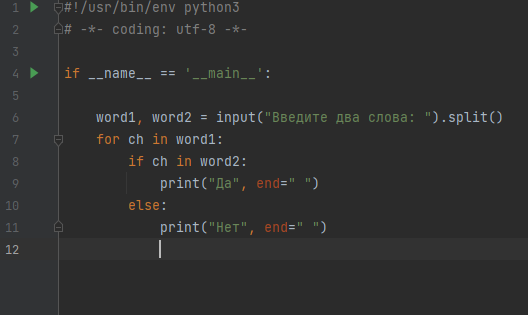


Рисунок 17 – код программы



Рисунок 18 – вывод для заданных условие слов



Рисунок 19 – вывод для произвольных слов

**ОТВЕТЫ НА КОНТРОЛЬНЫЕ ВОПРОСЫ**

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

3. Сложение, умножение, оператор принадлежности. Строковых функций в Python много, вот некоторые из них: chr() – Преобразует целое число в символ ord() – Преобразует символ в целое число len() – Возвращает длину строки str() – Изменяет тип объекта на string

4. В Python строки являются упорядоченными последовательностями символьных данных и могут быть проиндексированы. Доступ к отдельным символам в строке можно получить, указав имя строки, за которым следует число в квадратных скобках []. Индексация строк начинается с нуля: у первого символа индекс 0, следующего 1 и так далее. Индекс последнего символа в python — ‘‘длина строки минус один’’.

5. Если s это строка, выражение формы s[m:n] возвращает часть s , начинающуюся с позиции m , и до позиции n , но не включая позицию. Если пропустить первый индекс, срез начинается с начала строки. Аналогично, если опустить второй индекс s[n:], срез длится от первого индекса до конца строки.

6. Более легкое представление в памяти.

7. s.istitle()

8. if s1 in s2

9. s.find().

10. len(s)

11. s.count().

12. f-строки упрощают форматирование строк. Пример: print(f’’ This is {name}, he is {age} years old”)

13. string.find([, [, ]])

14. 'Hello, {}!'.format('Vasya')

15. string.isdigit()

16. 'foo.bar.baz.qux'.rsplit(sep='.') – пример разделения

17. string.islower()

18. s[0].isupper()

19. С точки зрения математической операции нельзя, можно лишь только вывести из без разделения друг от друга

20. s[::-1] – при помощи среза.

21. ‘–‘.join()

22. К верхнему – string.upper(), к нижнему – string.lower().

23. s[0].upper() s[len(s) – 1].upper()

24. s.isupper()

25. Если нужно сохранить символы, обозначающие конец слов.

26. s.replace(‘что заменить’, ‘на что заменить’)

27. string.endswith([, [, ]]), str.startswith(prefix[, start[, end]])

28. s.isspace()

29. Будет получена копия исходной строки в трёхкратном размере.

30. s.tittle()

31. s.partition() отделяет от s подстроку длиной от начала до первого вхождения . Возвращаемое значение представляет собой кортеж из трех частей: Часть s до Разделитель Часть s после

32. Когда нужен индекс последнего вхождения подстроки в строку.