

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ  
АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО  
ОБРАЗОВАНИЯ «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»  
ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ**

**Отчёт о лабораторной работе №6 по дисциплине основы программной  
инженерии**

Выполнила:

Нестеренко Тамара Антоновна,  
2 курс, группа ПИЖ-б-о-20-1,

Проверил:

Доцент кафедры инфокоммуникаций,  
Воронкин Р.А.

Ставрополь, 2021 г.

## ВЫПОЛНЕНИЕ

### 1. Практическая часть

```
S = 'spam"s'  
S = "spam's"
```

Рисунок 1 – Строки в апострофах и в кавычках

```
S = r'C:\newt.txt'
```

Рисунок 2 – «Сырые» строки – подавляют экранирование

```
S = r'\n\n\'[: -1]  
S = r'\n\n' + '\\'  
S = '\\n\n'
```

Рисунок 3 – Пример «сырых» строк – подавляют экранирование

```
>>> s = 'py'  
>>> t = 'th'  
>>> u = 'on'  
>>> s + t  
'pyth'  
>>> s + t + u  
'python'  
>>> print('Привет, ' + 'Мир!')  
Привет, Мир!
```

Рисунок 4 – Пример работы оператора сложения строк +

```
>>> s = 'ру'  
>>> s * 4  
'руруруру'  
>>> 4 * s  
'руруруру'  
>>>
```

Рисунок 5 – Пример работы оператора умножения строк \*

```
>>> 'ру' * -6  
,,  
>>>
```

Рисунок 6 – Пример работы оператора умножения строк \* с отрицательным множителем

```
>>> s = 'Python'
>>> s in 'I love Python.'
True
>>> s in 'I love Java'
False
```

Рисунок 7 – Пример работы оператора принадлежности подстроки in

```
>>> 'z' not in 'abc'
True
>>> 'z' not in 'xyz'
False
>>>
```

Рисунок 8 – Пример работы оператора not in

```
>>> ord('a')
97
>>> ord('#')
35
```

Рисунок 9 – Пример работы функции ord()

```
>>> chr(97)
'a'
>>> chr(35)
'#'
>>> chr(8364)
'€'
>>> chr(8721)
'Σ'
>>>
```

Рисунок 10 – Пример работы функции chr()

```
>>> s = 'Простая строка.'
>>> len(s)
15
```

Рисунок 11 – Пример работы функции len(s)

```
>>> str(49.2)
'49.2'
>>> str(3+4j)
'(3+4j)'
>>> str(3 + 29)
'32'
>>> str('py')
'py'
>>>
```

Рисунок 12 – Пример работы функции str(obj)

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

Рисунок 13 – Пример работы индексации строк

```
>>> s[6]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IndexError: string index out of range
```

Рисунок 14 – Пример попытки обращения по индексу большему, чем количество букв в слове

```
>>> s = 'foobar'
>>> s[-1]
'r'
>>> s[-2]
'a'
>>> len(s)
6
>>> s[-len(s)]
'f'
```

Рисунок 15 – Пример обращения по индексу меньшему чем  $-\text{len}(s)$

```
>>> s[-7]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IndexError: string index out of range
```

Рисунок 16 – Пример обращения по индексу меньшему чем  $-\text{len}(s)$ , приводит к ошибке

```
>>> s = 'python'
>>> s[2:5]
'tho'
>>> s[:4]
'pyth'
>>> s[0:4]
'pyth'
>>> s[2:0]
''
>>> s[2:]
'thon'
>>> s[2:len(s)]
'thon'
>>> s[:4] + s[4:]
'python'
>>> s[:4] + s[4:] == s
True
>>> t = s[:]
>>> id(s)
2690191210544
>>> id(t)
2690191210544
>>> s is t
True
```

Рисунок 17 – Пример среза строки

```

>>> s[2:2]
''
>>> s[4:2]
''
>>> s[-5:-2]
'yth'
>>> s[1:4]
'yth'
>>> s[-5:-2] == s [1:4]
True

```

Рисунок 18 – Пример работы различных форматов индексов в срезе строки

```

>>> s = 'foobar'
>>> s[0:6:2]
'foa'
>>> s[1:6:2]
'obr'

```

Рисунок 19 – Пример работы шага для среза строки

```

>>> s = '12345' * 5
>>> s
'1234512345123451234512345'
>>> s[:5]
'11111'
>>> s[4:5]
'55555'
>>> s[:-5]
'55555'
>>>

```

Рисунок 20 – Пример работы шага для среза строки

```

>>> s = 'python'
>>> s[5:0:-2]
'nhy'
>>>

```

Рисунок 21 – Пример работы шага для среза строки

```

>>> s = 'Если так говорит товарищ Наполеон, значит так оно и есть.'
>>> s[::-1]
'.ьтсе и оно кат тичанз ,ноелопан щиравог тировог кат илсЕ'
>>>

```

Рисунок 22 – Пример работы общей парадигмы для разворота (reverse) строки

```
>>> 'Hello, {}!'.format('vasya')  
'Hello, vasya!'
```

Рисунок 23 – Пример работы форматирования строк с помощью метода format

```
>>> '{0}, {1}, {2}'.format('a', 'b', 'c')  
'a, b, c'  
>>> '{}', {}, {}'.format('a', 'b', 'c')  
'a, b, c'  
>>> '{2}, {1}, {0}'.format('a', 'b', 'c')  
'c, b, a'  
>>> '{2}, {1}, {0}'.format(*'abc')  
'c, b, a'  
>>> '{0}{1}{0}'.format('abra', 'cad')  
'abracadabra'
```

Рисунок 24 – Пример работы форматирования строк с помощью метода format

```
>>> "Units destroyed: {players[0]}".format(players = [1, 2, 3])  
'Units destroyed: 1'  
>>> "Units destroyed: {players[0]!r}".format(players = ['1', '2', '3'])  
"Units destroyed: '1'"
```

Рисунок 25 – Пример работы форматирования строк с помощью метода format

```

>>> coord = (3, 5)
>>> 'x: {0[0]}; y: {0[1]}'.format(coord)
'x: 3; y: 5'
>>> "repr() shows quotes: {!r}; str() doesn't: {!s}".format('test1', 'test2')
"repr() shows quotes: 'test1'; str() doesn't: test2"
>>> '{:<30}'.format('left aligned')
'left aligned'
>>> '{:<30}'.format('right aligned')
'right aligned'
>>> '{:>30}'.format('right aligned')
'right aligned'
>>> '{:^30}'.format('centered')
'centered'
>>> '{:*^30}'.format('centered')
'*****centered*****'
>>> '{:+f}; {:+f}'.format(3.14, -3.14)
'+3.140000; -3.140000'
>>> '{: f}; {: f}'.format(3.14, -3.14)
' 3.140000; -3.140000'
>>> '{: -f}; {: -f}'.format(3.14, -3.14)
'3.140000; -3.140000'
>>> "int: {0:d}; hex: {0:x}; oct: {0:o}; bin: {0:b}".format(42)
'int: 42; hex: 2a; oct: 52; bin: 101010'
>>> "int: {0:d}; hex: {0:#x}; oct: {0:#o}; bin: {0:#b}".format(42)
'int: 42; hex: 0x2a; oct: 0o52; bin: 0b101010'
>>> points = 19.5
>>> total = 22
>>> 'Correct answer: {:.2%}'.format(points/total)
'Correct answer: 88.64%'
>>>

```

Рисунок 26 – Пример работы форматирования строк с помощью метода format

```

>>> n = 20
>>> m = 25
>>> prod = n * m
>>> print('Произведение', n, 'на', m, 'равно', prod)
Произведение 20 на 25 равно 500
>>> print(f'Произведение {n} на {m} равно {prod}')
Произведение 20 на 25 равно 500
>>>

```

Рисунок 27 – Пример форматирования с помощью f-строк



```

>>> var = 'Гав'
>>> print(f'Собака говорит {var}!')
Собака говорит Гав!
>>> print(f"Собака говорит {var}!")
Собака говорит Гав!
>>> print(f'''Собака говорит {var}!''')
Собака говорит Гав!
>>>

```

Рисунок 28 – Пример форматирования с помощью f-строк

```

>>> s = 'python'
>>> s[3] = 't'
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'str' object does not support item assignment
>>> s = s[:3] + 't' + s[4:]
>>> s
'pytton'
>>> s = 'python'
>>> s = s.replace('h', 't')
>>> s
'pytton'
>>>

```

Рисунок 29 – Пример изменения строк

```

>>> s = 'everyThing you Can Imagine is rEAl'
>>> s.capitalize()
'Everything you can imagine is real'
>>> s = 'follow us @PYTHON'
>>> s.capitalize()
'Follow us @python'
>>> 'everyThing you Can Imagine is rEAl'.lower()
'everything you can imagine is real'
>>> 'everyThing you Can Imagine is rEAl'.swapcase()
'EVERYTHING YOU cAN iMAGINE IS Real'
>>> 'the sun also rises'.title()
'The Sun Also Rises'
>>> 'follow us @PYTHON'.title()
'Follow Us @Python'
>>> 'follow us @PYTHON'.upper()
'FOLLOW US @PYTHON'

```

Рисунок 30 – Пример изменения регистра строки

```
>>> 'foo goo moo'.count('oo')
3
>>> 'foo goo moo'.count('oo', 0, 8)
2
```

Рисунок 31 – Пример нахождения и замены подстроки в строке

```
>>> 'python'.endswith('on')
True
>>> 'python'.endswith('or')
False
>>> 'python'.endswith('yt', 0, 4)
False
>>> 'python'.endswith('yt', 2, 4)
False
>>> 'Follow us @Python'.find('Us')
-1
>>> 'Follow Us @Python'.find('Us')
7
>>> 'Follow Us @Python'.find('you')
-1
>>> 'Follow Us @Python'.find('Us', 4)
7
>>> 'Follow Us @Python'.find('Us', 4, 7)
-1
>>> 'Follow Us @Python'.index('you')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: substring not found
>>> 'Follow Us @Python'.rfind('o')
15
```

Рисунок 32 – Пример нахождения и замены подстроки в строке

```

>>> 'Follow Us @Python'.startswith('Fol')
True
>>> 'Follow Us @Python'.startswith('Go')
False
>>> 'Follow Us @Python'.startswith('Us', 7)
True
>>> 'Follow Us @Python'.startswith('Us', 8, 16)
False
>>> 'abc123'.isalnum()
True
>>> 'abc$123'.isalnum()
False
>>> ''.isalnum()
False
>>> 'ABCabc'.isalpha()
True
>>> 'abc123'.isalpha()
False
>>>

```

Рисунок 33 – Пример нахождения и замены подстроки в строке

```

>>> 'foo32'.isidentifier()
True
>>> '32foo'.isidentifier()
False
>>> 'foo$32'.isidentifier()
False
>>> 'and'.isidentifier()
True
>>> from keyword import iskeyword
>>> iskeyword('and')
True
>>> 'abc'.islower()
True
>>> 'abc1$d'.islower()
True
>>> 'Abc1$d'.islower()
False
>>> 'a\tb'.isprintable()
False
>>> 'a b'.isprintable()
True
>>> ''.isprintable()
True

```

Рисунок 34 – Пример нахождения и замены подстроки в строке

```
>>> 'a\nb'.isprintable()
False
>>> ' \t \n '.isspace()
True
>>> ' a '.isspace()
False
>>> '\f\u2005\r'.isspace()
True
>>> 'This Is A Title'.istitle()
True
>>> 'This is a title'.istitle()
False
>>> 'Give Me The $$$@ Ball!'.istitle()
True
>>> 'ABC'.isupper()
True
>>> 'ABC1$D'.isupper()
True
>>> 'Abc1$D'.isupper()
False
>>>
```

Рисунок 35 – Пример нахождения и замены подстроки в строке

```
>>> 'py'.center(10)
'   py   '
>>> 'py'.center(10, '-')
'----py----'
>>> 'python'.center(2)
'python'
>>> 'a\tb\tc'.expandtabs()
'a      b      c'
>>> 'aaa\tbbb\tc'.expandtabs()
'aaa      bbb      c'
>>> 'a\tb\tc'.expandtabs(4)
'a  b  c'
>>> 'aaa\tbbb\tc'.expandtabs(tabsize=4)
'aaa bbb c'
>>> 'python'.ljust(10)
'python '
>>> 'python'.ljust(10, '-')
'python----'
>>> 'python'.ljust(2)
'python'
>>> '   foo bar baz   '.rstrip()
'foo bar baz '
>>> '\t\nfoo\t\nbar\t\nbaz'.rstrip()
'foo\t\nbar\t\nbaz'
>>> 'https://www.pythonru.com'.rstrip('/:pths')
'www.pythonru.com'
```

Рисунок 36 – Пример выравнивания строк, отступов

```

>>> 'I hate python! I hate python! I hate python! I hate python!'.replace('h
ate', 'love')
'I love python! I love python! I love python! I love python!'
>>> 'I hate python! I hate python! I hate python! I hate python!'.replace('h
ate', 'love', 2)
'I love python! I love python! I hate python! I hate python!'
>>> 'python'.rjust(10)
'      python'
>>> 'python'.rjust(10, '-')
'-----python'
>>> 'python'.rjust(2)
'python'
>>> '    foo bar baz    '.rstrip()
'    foo bar baz'
>>> 'foo\t\nbar\t\nbaz\t\n'.rstrip()
'foo\t\nbar\t\nbaz'
>>> 'foo.$$$;'.rstrip(';$.')
'foo'
>>> s = '    foo bar baz\t\t\t'
>>> s = s.lstrip()
>>> s = s.rstrip()
>>> s
'foo bar baz'
>>> 'www.pythonru.com'.strip('w.moc')
'pythonru'
>>> '    foo bar baz\t\t\t'.lstrip().rstrip()
'foo bar baz'
>>> '    foo bar baz\t\t\t'.strip()
'foo bar baz'
>>> 'www.puthonru.com'.lstrip('w.').rstrip('.moc')
'puthonru'
>>> 'www.puthonru.com'.strip('.moc')
'www.puthonru'
>>> '42'.zfill(5)
'00042'
>>> '+42'.zfill(8)
'+0000042'
>>> '-42'.zfill(8)
'-0000042'
>>> '-42'.zfill(3)
'-42'
>>> 'foo'.zfill(6)
'000foo'
>>>

```

Рисунок 37 – Пример выравнивания строк, отступов

```

>>> ','.join(['foo', 'bar', 'baz', 'qux'])
'foo, bar, baz, qux'
>>> list('corge')
['c', 'o', 'r', 'g', 'e']
>>> ':'.join('corge')
'c:o:r:g:e'
>>> '---'.join(['foo', 23, 'bar'])
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: sequence item 1: expected str instance, int found
>>> '---'.join(['foo', str(23), 'bar'])
'foo---23---bar'
>>> 'foo.bar'.partition('.')
('foo', '.', 'bar')
>>> 'foo@@bar@@baz'.partition('@@')
('foo', '@@', 'bar@@baz')

```

Рисунок 38 – Пример методов преобразования строк в список

```

>>> 'foo.bar'.partition('@@')
('foo.bar', '', '')
>>> 'foo@@bar@@baz'.partition('@@')
('foo', '@@', 'bar@@baz')
>>> 'foo@@bar@@baz'.rpartition('@@')
('foo@@bar', '@@', 'baz')
>>> 'foo bar baz qux'.rsplit()
['foo', 'bar', 'baz', 'qux']
>>> 'foo\n\tbar\t\tbaz\r\fqux'.rsplit()
['foo', 'bar', 'baz', 'qux']
>>> 'foo.bar.baz.qux'.rsplit(sep='.')
['foo', 'bar', 'baz', 'qux']
>>> 'foo...bar'.rsplit(sep='.')
['foo', '', '', 'bar']
>>> 'foo\t\t\t\t\tbar'.rsplit()
['foo', 'bar']
>>> 'www.pythonru.com'.rsplit(sep='.', maxsplit=1)
['www.pythonru', 'com']
>>> 'www.pythonru.com'.rsplit(sep='.', maxsplit=-1)
['www', 'pythonru', 'com']
>>> 'www.pythonru.com'.rsplit(sep='.')
['www', 'pythonru', 'com']
>>> 'www.pythonru.com'.split('.', maxsplit=1)
['www', 'pythonru.com']
>>> 'www.pythonru.com'.rplit('.', maxsplit=1)

```

Рисунок 39 – Пример методов преобразования строк в список

```

>>> 'www.pythonru.com'.rsplit('.', maxsplit=1)
['www.pythonru', 'com']
>>> 'foo\nbar\r\nbaz\fqux\u2028quux'.splitlines()
['foo', 'bar', 'baz', 'qux', 'quux']
>>> 'foo\f\f\fbar'.splitlines()
['foo', '', '', 'bar']
>>> 'foo\nbar\r\nbaz\fqux\nqux'.splitlines(True)
['foo\n', 'bar\r\n', 'baz\x0c', 'qux\n', 'qux']
>>> 'foo\nbar\r\nbaz\fqux\nqux'.splitlines(8)
['foo\n', 'bar\r\n', 'baz\x0c', 'qux\n', 'qux']
>>>

```

Рисунок 40 – Пример методов преобразования строк в список

```

#!/usr/bin/env python3
# -*- coding: utf-8 -*-

if __name__ == '__main__':
    s = input("Введите предложение: ")
    r = s.replace(' ', '_')
    print("Предложение после замены: {r}")

```

Рисунок 41 – Пример программы заменяющей пробелы на символ «\_»



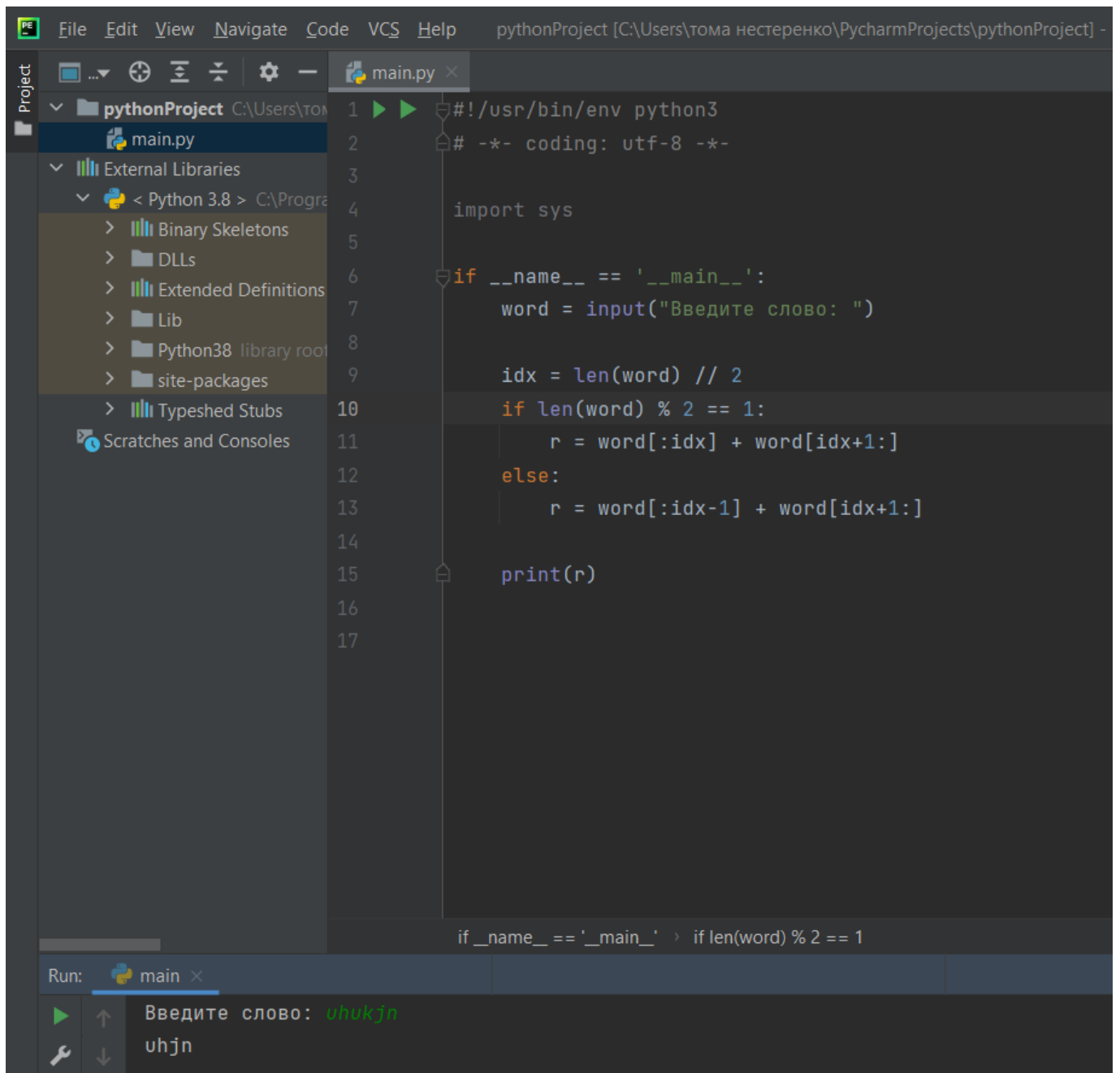


Рисунок 42 – Пример программы удаляющей буквы в слове

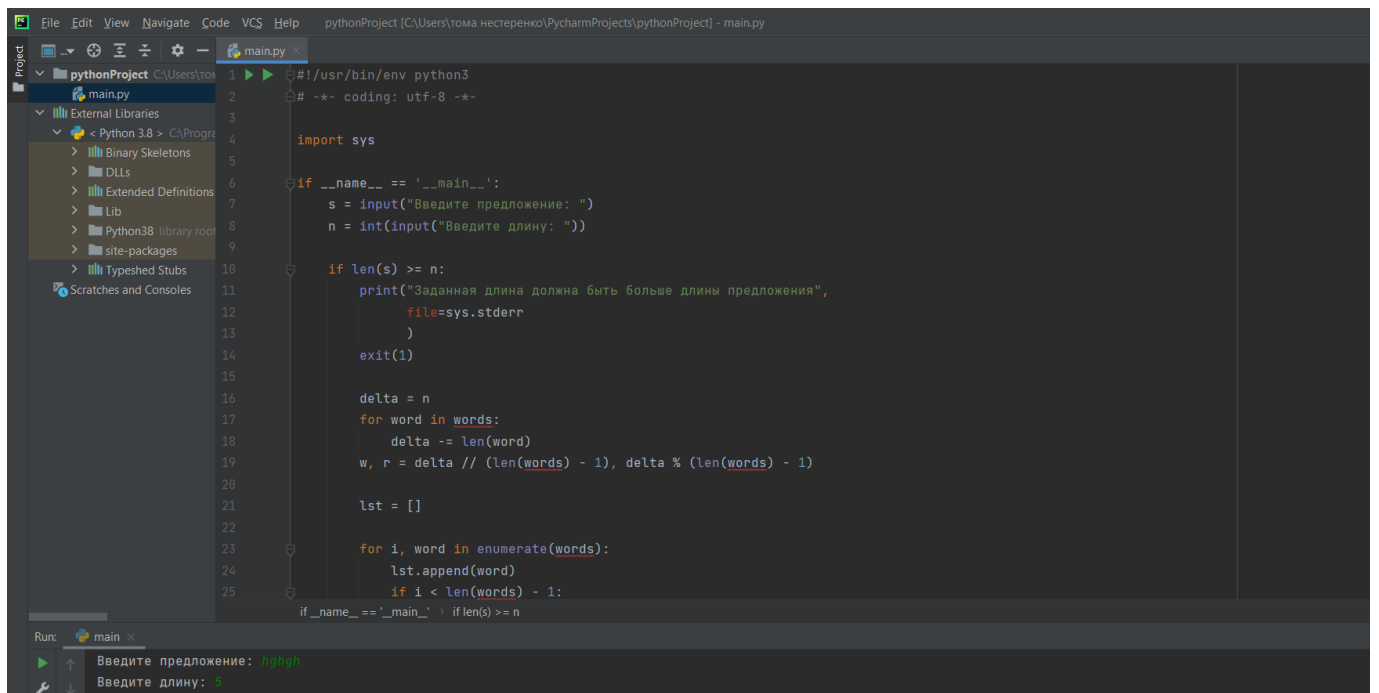


Рисунок 43 – Пример программы, вставляющей пробелы