

Лабораторная работа № 4

Python: строки

Содержание

1 Цель работы.....	2
2 Краткие теоретические сведения.....	2
3 Оборудование рабочего места.....	2
4 Задание.....	2
5 Содержание отчета.....	2
6 Порядок выполнения работы.....	3
6.1 Понятие строки.....	3
6.2 Основы работы со строками.....	4
6.3 Индексы элементов. Слайсинг.....	6
6.4 Методы работы со строками.....	7
6.5 Самостоятельная работа.....	7

1 Цель работы

Изучить порядок работы со строками в языке Python.

2 Краткие теоретические сведения

Программирование — это процесс проектирования, написания и отладки программ. Что такое программа? Программа — это последовательность инструкций, предназначенная для исполнения устройством управления вычислительной машины [1]. Есть, также, официальные трактовки термина «Программа»:

Программа — данные, предназначенные для управления конкретными компонентами системы обработки информации в целях реализации определённого алгоритма. [2]

Программа — представленная в объективной форме совокупность данных и команд, предназначенных для функционирования ЭВМ и других компьютерных устройств с целью получения определённого результата, включая подготовительные материалы, полученные в ходе разработки программы для ЭВМ, и порождаемые ею аудиовизуальные отображения. [3]

3 Оборудование рабочего места

Персональный компьютер под управлением ОС Linux или Windows с установленной средой программирования IDLE и языком программирования Python версии 3.4 и выше.

4 Задание

- Ознакомиться с определением «строка».
- Узнать область использования строк.
- Выполнить примеры, приведенные в разделе «Порядок выполнения работы».
- Решить примеры для самостоятельной работы.

5 Содержание отчета

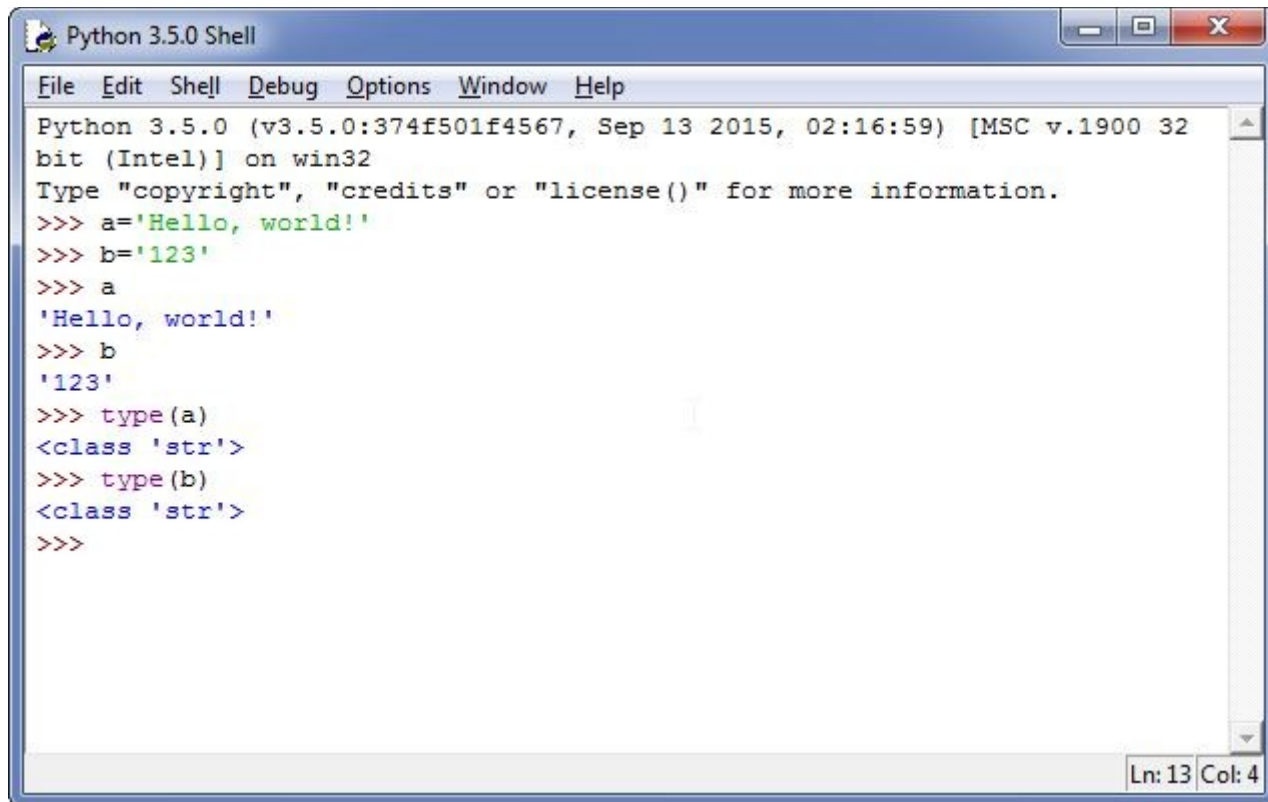
Отчет представляется в виде текстового документа, содержащего скриншоты окна редактора, содержащего код программы и командного окна с результатом выполнения программы для всех самостоятельных заданий.

6 Порядок выполнения работы

6.1 Понятие строки

Слово «строка» в программировании может применяться как к набору символов, так и к соответствующему типу данных. В Python в оригинале этот тип данных называется String.

Строка в Python заключается в одинарные или двойные кавычки. Рассмотрим примеры строк:

A screenshot of a Python 3.5.0 Shell window. The window has a title bar with the text 'Python 3.5.0 Shell' and standard Windows window controls (minimize, maximize, close). Below the title bar is a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Window', and 'Help'. The main area of the window contains a text editor with the following text:

```
Python 3.5.0 (v3.5.0:374f501f4567, Sep 13 2015, 02:16:59) [MSC v.1900 32
bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> a='Hello, world!'
>>> b='123'
>>> a
'Hello, world!'
>>> b
'123'
>>> type(a)
<class 'str'>
>>> type(b)
<class 'str'>
>>>
```

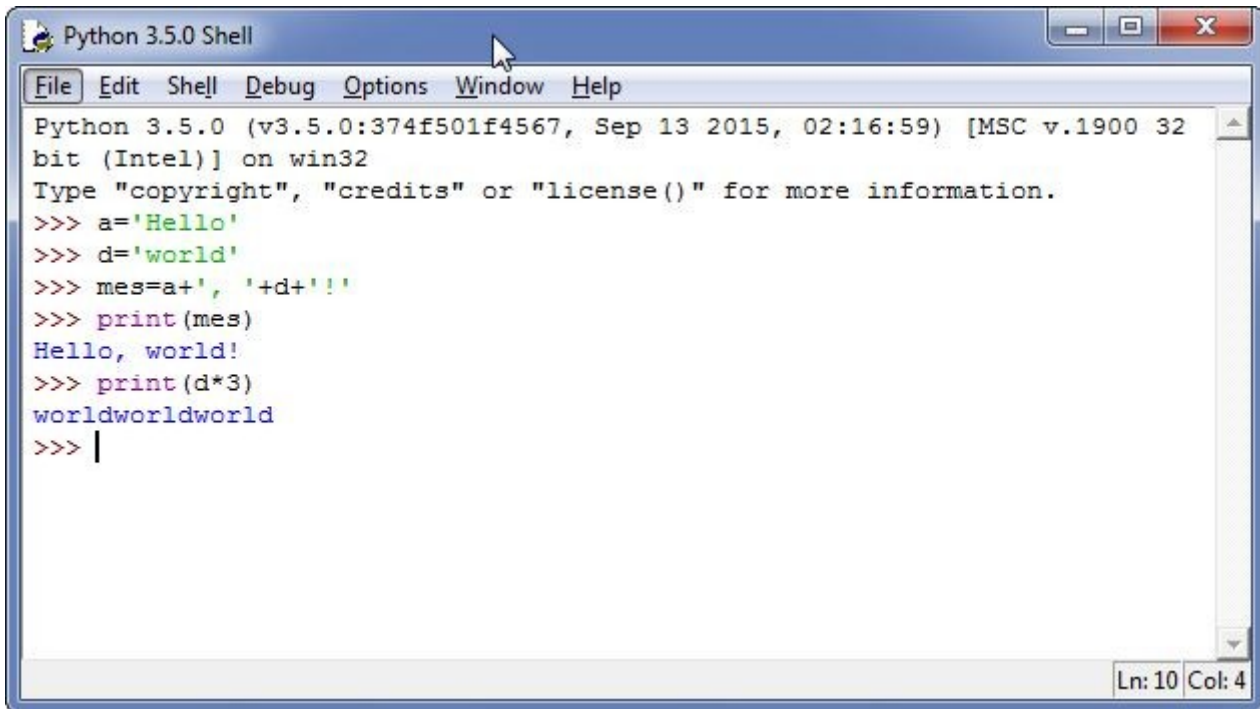
The text is color-coded: keywords like 'Python', 'type', and '<class' are in blue; string literals like 'Hello, world!' and '123' are in green; and the prompt '>>>' is in red. At the bottom right of the window, there is a status bar showing 'Ln: 13 Col: 4'.

При выводе строка также заключается в двойные кавычки. Поэтому '123' – это, однозначно, строка. И для работы с числом 123 необходимо сначала выполнить смену типа данных (функции `int` или `float`).

6.2 Основы работы со строками

Примеры операций со строками

Со строками можно проводить, например, операции конкатенации (сложения) и дублирования (умножения):

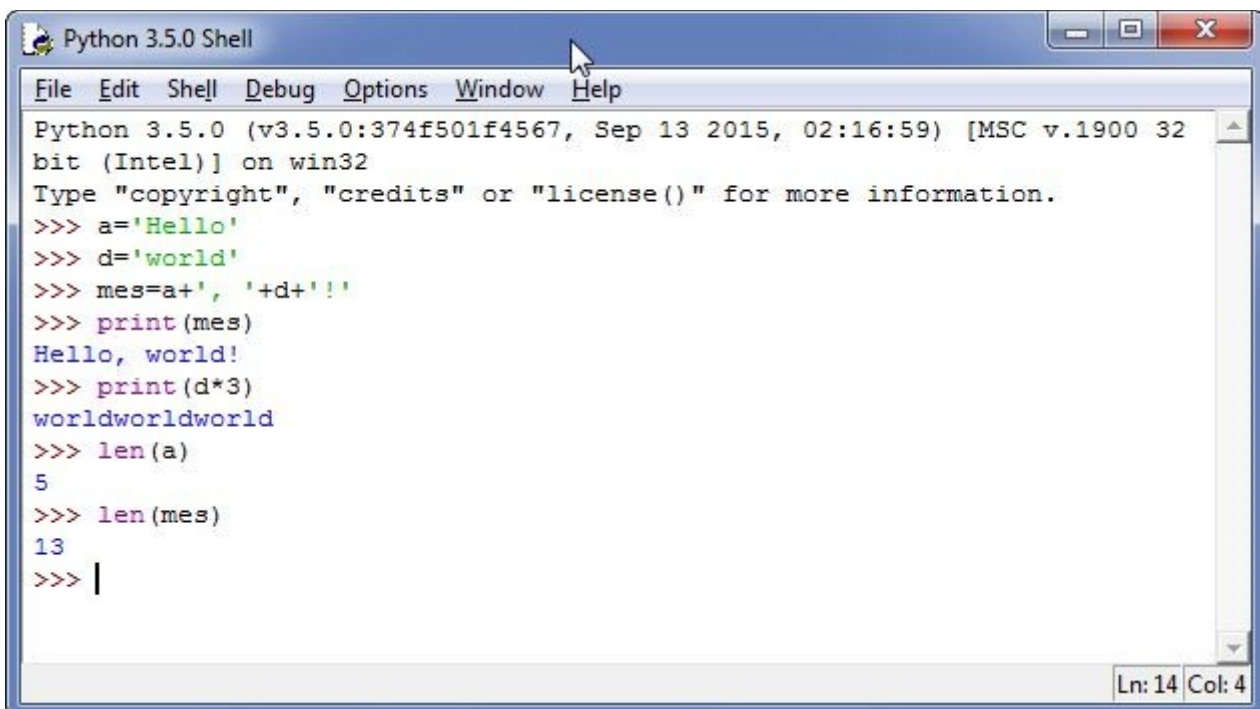


```
Python 3.5.0 Shell
File Edit Shell Debug Options Window Help
Python 3.5.0 (v3.5.0:374f501f4567, Sep 13 2015, 02:16:59) [MSC v.1900 32
bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> a='Hello'
>>> d='world'
>>> mes=a+', '+d+'!'
>>> print(mes)
Hello, world!
>>> print(d*3)
worldworldworld
>>> |
```

Ln: 10 Col: 4

Определение длины строки (кол-во символов)

Длина строки определяется с помощью встроенной функции len:



```
Python 3.5.0 Shell
File Edit Shell Debug Options Window Help
Python 3.5.0 (v3.5.0:374f501f4567, Sep 13 2015, 02:16:59) [MSC v.1900 32
bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> a='Hello'
>>> d='world'
>>> mes=a+', '+d+'!'
>>> print(mes)
Hello, world!
>>> print(d*3)
worldworldworld
>>> len(a)
5
>>> len(mes)
13
>>> |
```

Ln: 14 Col: 4

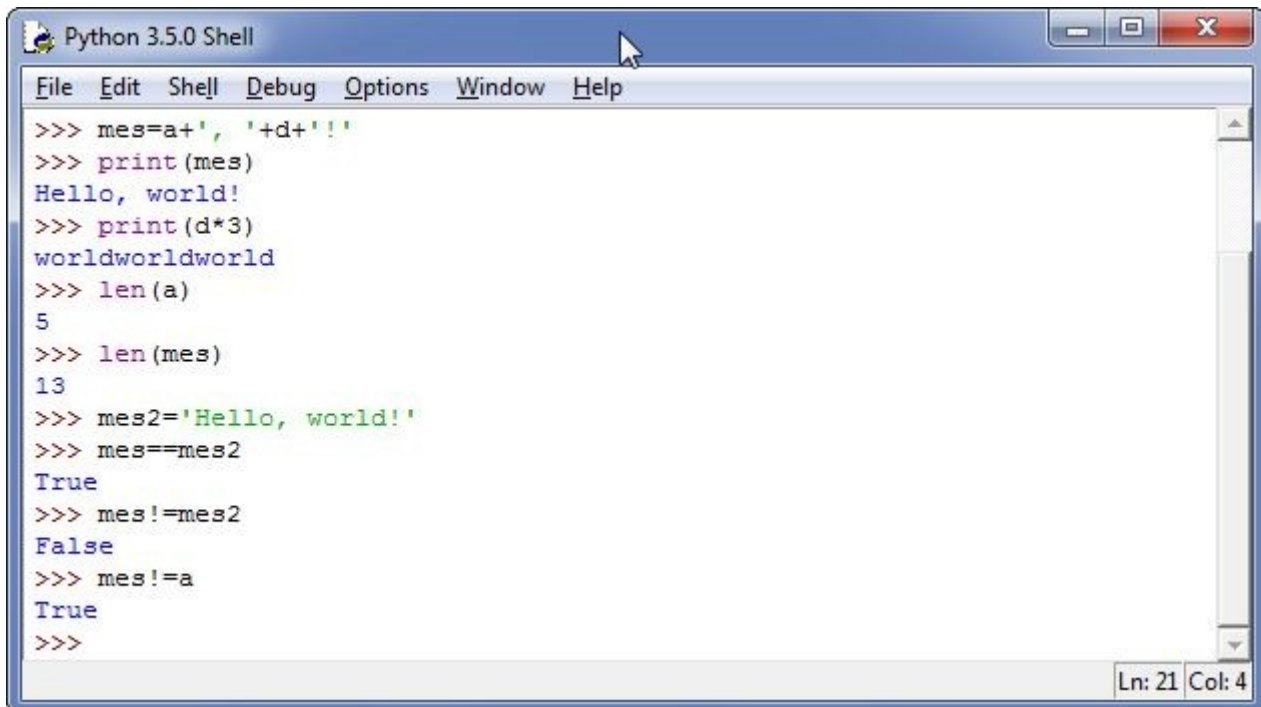
Символы и кодировка ASCII

ord('символ') – возвращает код ASCII указанного символа,

chr(код) – возвращает символ, соответствующий данному коду в таблице ASCII.

Сравнение строк

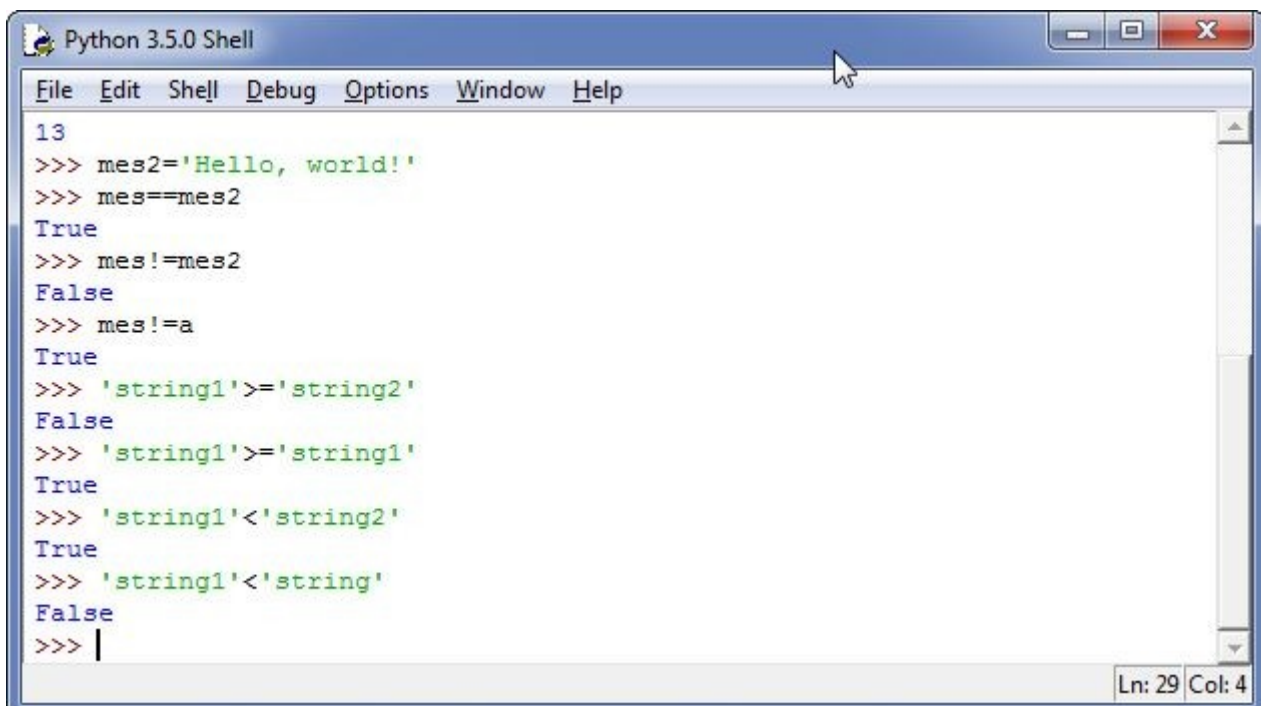
Строки можно использовать как операнды операторов сравнения. Можно проверить, что строки равны (или не равны), например:

A screenshot of a Python 3.5.0 Shell window. The window has a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Window', and 'Help'. The main text area contains the following code:

```
>>> mes=a+', '+d+'!'
>>> print(mes)
Hello, world!
>>> print(d*3)
worldworldworld
>>> len(a)
5
>>> len(mes)
13
>>> mes2='Hello, world!'
>>> mes==mes2
True
>>> mes!=mes2
False
>>> mes!=a
True
>>>
```

The status bar at the bottom right shows 'Ln: 21 Col: 4'.

Также можно использовать операторы строгого и нестрогого неравенства, например:

A screenshot of a Python 3.5.0 Shell window. The window has a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Window', and 'Help'. The main text area contains the following code:

```
13
>>> mes2='Hello, world!'
>>> mes==mes2
True
>>> mes!=mes2
False
>>> mes!=a
True
>>> 'string1'>='string2'
False
>>> 'string1'>='string1'
True
>>> 'string1'<'string2'
True
>>> 'string1'<'string'
False
>>> |
```

The status bar at the bottom right shows 'Ln: 29 Col: 4'.

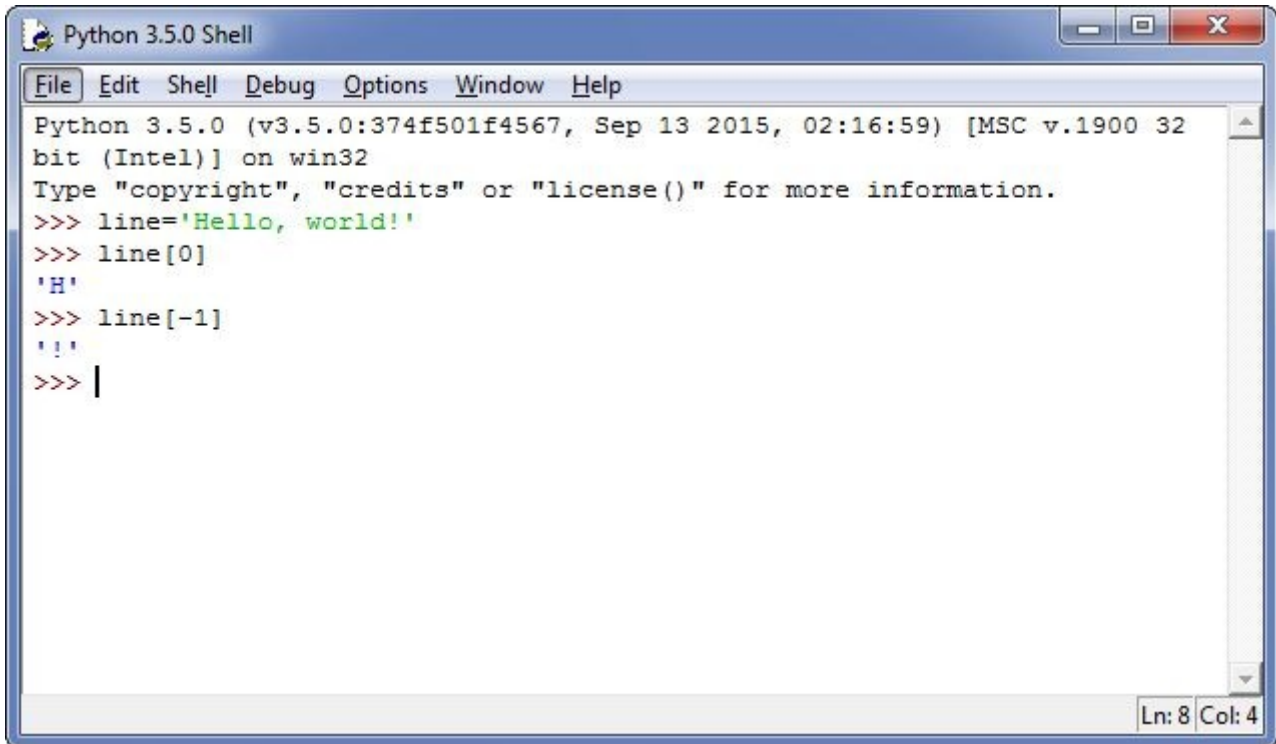
При сравнении строк используется лексикографический порядок: меньше считается та строка, которая в данном порядке идет раньше.

Специальные символы форматирования

При необходимости можно использовать символы: '\n' - перевод строки, '\t' - табуляция.

6.3 Индексы элементов. Слайсинг

Каждый элемент строки имеет свой номер. Нумерация начинается с 0. Чтобы обратиться к элементу строки, нужно ввести его индекс в квадратных скобках. Если индекс отрицательный, происходит отсчет с конца строки. Например:

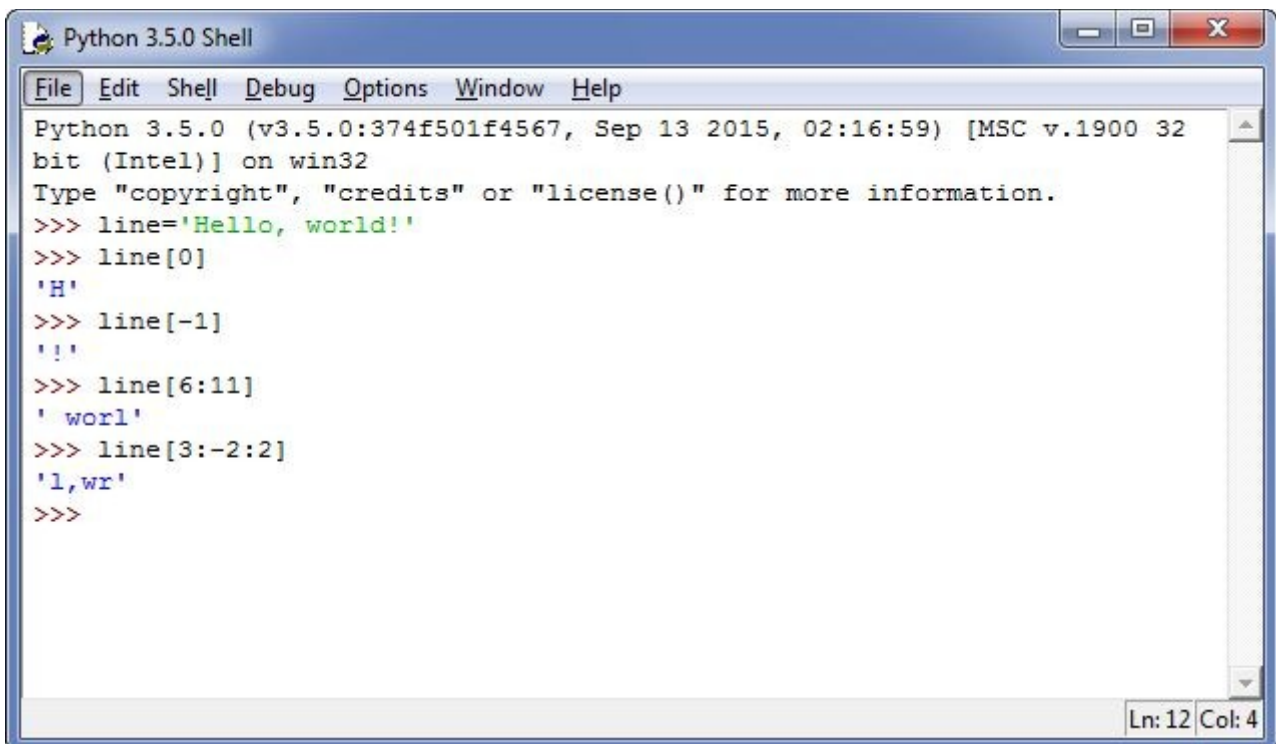


```
Python 3.5.0 Shell
File Edit Shell Debug Options Window Help
Python 3.5.0 (v3.5.0:374f501f4567, Sep 13 2015, 02:16:59) [MSC v.1900 32
bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> line='Hello, world!'
>>> line[0]
'H'
>>> line[-1]
'!'
>>> |
```

Ln: 8 Col: 4

Слайсинг (или срез) — это получение части строки по номерам её элементов.

Можно использовать два варианта слайсинга: без и с указанием шага. Например:



```
Python 3.5.0 Shell
File Edit Shell Debug Options Window Help
Python 3.5.0 (v3.5.0:374f501f4567, Sep 13 2015, 02:16:59) [MSC v.1900 32
bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> line='Hello, world!'
>>> line[0]
'H'
>>> line[-1]
'!'
>>> line[6:11]
' worl'
>>> line[3:-2:2]
'l,wr'
>>>
```

Ln: 12 Col: 4

Два числа обозначают начало и конец среза. Второе число не входит в срез. Три числа — начало, конец и шаг, с которым выполняется срез.

6.4 Методы работы со строками

Python поддерживает парадигму объектно-ориентированного программирования (ООП). Для ООП характерно понятие метода — функции, которая принадлежит определенным объектам. В ранних версиях Python часть функционала, реализованного, в настоящее время, через методы строк, была реализована с помощью отдельного модуля.

В общем виде применение метода к строке выглядит следующим образом:

`строка.метод([аргументы])`

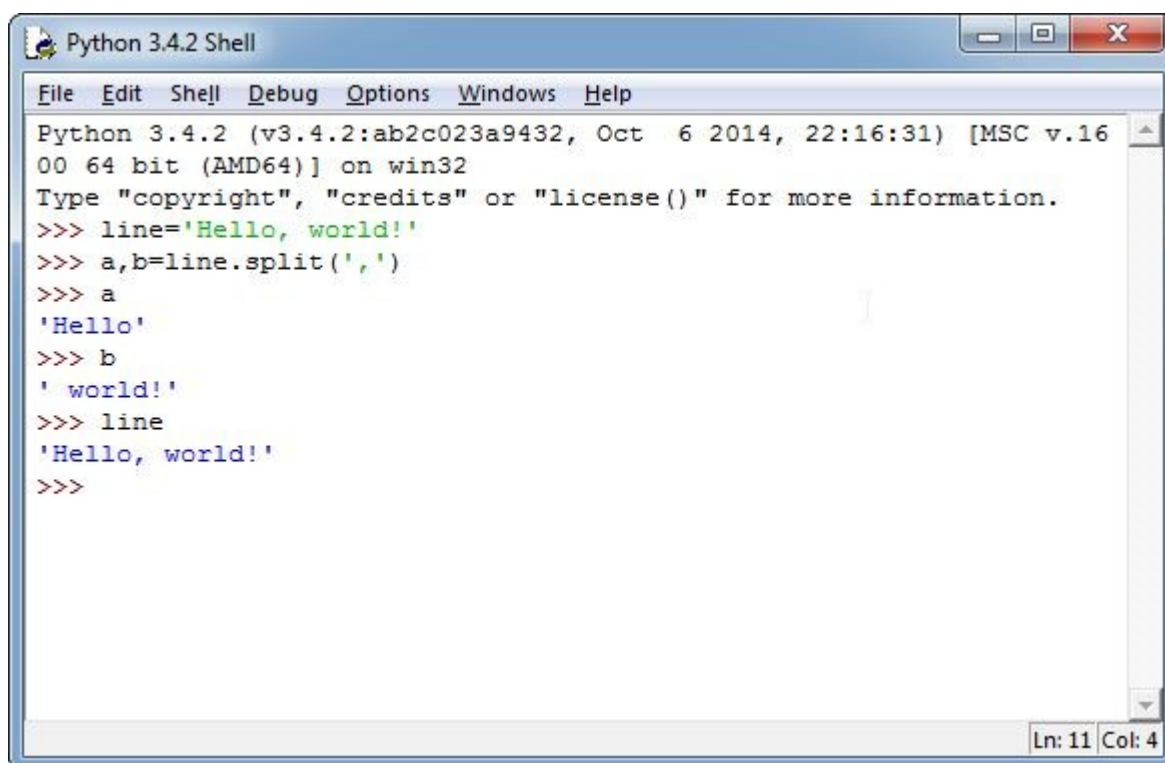
где:

строка — строка, над которой производится операция, следует помнить, что строка — это неизменяемый тип данных, поэтому для сохранения результатов необходимо использовать операцию присваивания,

метод — применяемый к строке метод,

аргументы — аргументы, могут требоваться или не требоваться — в зависимости от метода.

Рассмотрим применение методов на примере:

A screenshot of a Python 3.4.2 Shell window. The window has a title bar 'Python 3.4.2 Shell' and a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Windows', and 'Help'. The main text area shows the following code and output:

```
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct 6 2014, 22:16:31) [MSC v.1600 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> line='Hello, world!'
>>> a,b=line.split(',')
>>> a
'Hello'
>>> b
' world!'
>>> line
'Hello, world!'
>>>
```

The status bar at the bottom right shows 'Ln: 11 Col: 4'.

В данном случае мы при помощи метода `split` разделили строку по разделителю «`,`» (указан как аргумент). Значение переменной `line` осталось неизменным.

Рассмотрим некоторые методы, которые могут пригодиться вам в дальнейшей работе:

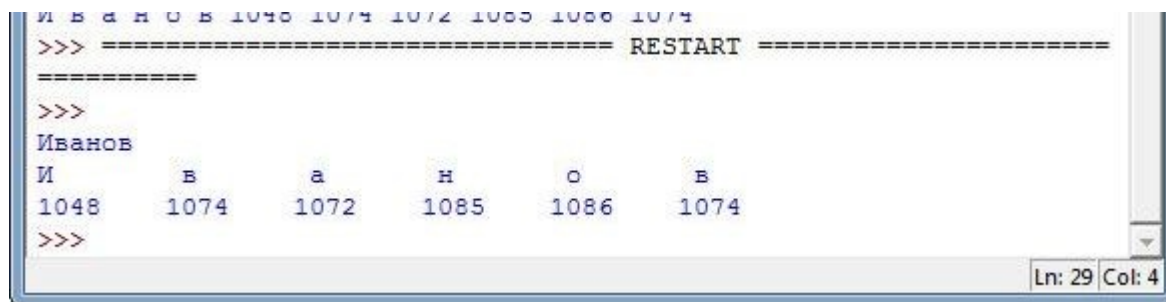
- `строка.split('разделитель')` — разбивает строку по разделителю,
- `разделитель.join(список)` — собирает строку из списка с разделителем,
- `строка.count('подстрока')` — подсчитывает количество вхождений подстроки в строку, считаются неперекрывающиеся вхождения,
- `строка.upper()` — преобразовывает все буквы в заглавные,
- `строка.lower()` — преобразовывает все буквы в строчные,

- `строка.find('подстрока')` - позиция первого вхождения подстроки в строку, возвращает -1, если вхождение нет,
- `строка.replace('подстрока','подстрока')` — замена,
- `строка.isalpha()` - проверяет, состоит ли строка только из букв,
- `строка.isdigit()` - проверяет, состоит ли строка только из цифр,
- `строка.startswith('подстрока')` — проверяет, начинается ли строка с подстроки,
- `строка.endswith('подстрока')` — проверяет, заканчивается ли строка подстрокой.

Это лишь некоторые методы строк. Для более подробного знакомства с методами рекомендуем вам читать документацию или информационные материалы в Интернете.

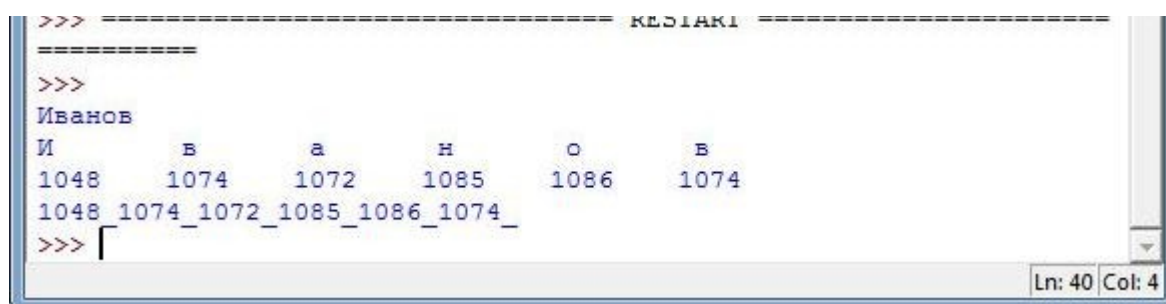
6.5 Самостоятельная работа

1. Напишите программу, которая преобразует буквы вашей фамилии в соответствующие коды ASCII и выведет их на экран. Желательно вывести код символа строго под соответствующей буквой. Например:



```
И В А Н О В 1048 1074 1072 1085 1086 1074
>>> ===== RESTART =====
>>>
Иванов
И      В      а      н      о      в
1048   1074   1072   1085   1086   1074
>>>
```

2. Доработайте программу (п.1) так, чтобы она выводила строку, которая состоит из кодов ASCII, разделенных знаком подчеркивания. Например:



```
>>> ===== RESTART =====
>>>
Иванов
И      В      а      н      о      в
1048   1074   1072   1085   1086   1074
1048_1074_1072_1085_1086_1074_
>>> |
```

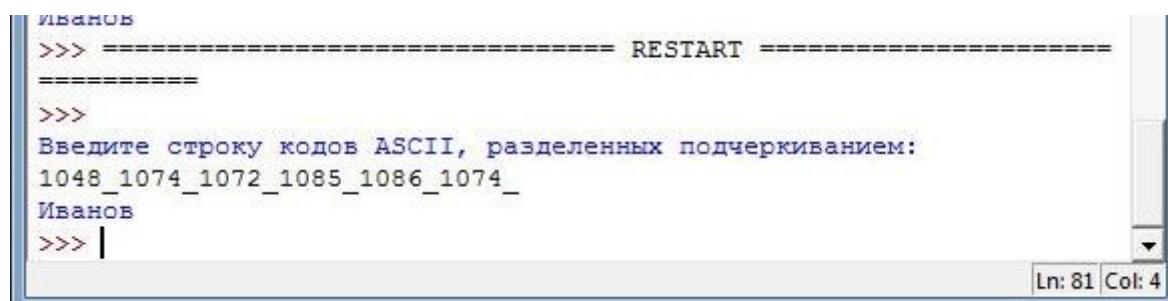
Скопируйте полученную строку (в примере «1048_1074_1072_1085_1086_1074_») и вставьте её в отчет.

3. Напишите программу, которая из строки, состоящей из кодов ASCII, разделенных символом подчеркивания, формирует строку символов. Проверьте работу программы, используя результат, полученный в программе 2.

Для быстрого доступа к элементам строки можно использовать цикл вида:

```
for i in line.split('_'):
    ...
    ...
```

Например:



```
Иванов
>>> ===== RESTART =====
>>>
Введите строку кодов ASCII, разделенных подчеркиванием:
1048_1074_1072_1085_1086_1074_
Иванов
>>> |
```

Шифр Цезаря

Шифр Цезаря, также известный как шифр сдвига, код Цезаря или сдвиг Цезаря — один из самых простых и наиболее широко известных методов шифрования.

Шифр Цезаря — это вид шифра подстановки, в котором каждый символ в открытом тексте заменяется символом, находящимся на некотором постоянном числе позиций левее или правее него в алфавите. Например, в шифре со сдвигом вправо на 3, А была бы заменена на Г, Б станет Д, и так далее.

Например:

Ключ шифра = 3

Исходный алфавит: АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ

Шифрованный: ГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯАБВ

5. Написать программу, которая на основании исходного алфавита (строка) и ключа (натуральное число) будет формировать шифрованный алфавит шифра Цезаря.

Например:

```
>>> ===== RESTART =====
>>>
Введите исходный алфавит: АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ
Введите ключ шифра 3
ГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯАБВ
>>> ===== RESTART =====
>>>
Введите исходный алфавит: АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ
Введите ключ шифра 10
ЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯАБВГДЕЁЖЗИ
>>> |
```

Примечание: претендующим на оценку «отлично» решить задачу минимум двумя способами.

6. Написать программу, на вход которой поступают:

- исходный алфавит,
- ключ,
- сообщение.

На выходе программы — зашифрованное шифром Цезаря сообщение.

Например:

```
nameerror: name 'line' is not defined
>>> ===== RESTART =====
>>>
Введите исходный алфавит: АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ
Введите ключ шифра 3
Введите сообщение: СООБЩЕНИЕ
ФССДЪЗРЛЗ
>>>
```

Зашифровать свою фамилию. Использовать алфавит из примера, буквы — большие.