Лабораторная работа №5
Строки

1 Цель работы

Изучить строки и научиться применять полученные знания на практике.

2 Краткая теория

2.1 Строки

Строка считывается со стандартного ввода функцией input(). Напомним, что для двух строк определена операция сложения (конкатенации), также определена операция умножения строки на число.

Строка состоит из последовательности символов. Узнать количество символов (длину строки) можно при помощи функции len.

Любой другой объект в Питоне можно перевести к строке, которая ему соответствует. Для этого нужно вызвать функцию str(), передав ей в качестве параметра объект, переводимый в строку.

На самом деле каждая строка, с точки зрения Питона, – это объект класса str. Чтобы получить по объекту другой объект другого класса, как-то ему соответствующий, можно использовать функцию приведения. Имя этой функции совпадает с именем класса, к которому мы приводим объект. (Для знатоков: эта функция – это конструктор объектов данного класса.) Пример: int – класс для целых чисел. Перевод строки в число осуществляется функцией int().

```
s = input()
print(len(s))
t = input()
number = int(t)
u = str(number)
print(s * 3)
print(s + ' ' + u)
```

2.2 Срезы (slices)

Срез (slice) – извлечение из данной строки одного символа или некоторого фрагмента подстроки или подпоследовательности.

Есть три формы срезов. Самая простая форма среза: взятие одного символа строки, а именно, S[i] – это срез, состоящий из одного символа, который имеет номер i. При этом считается, что нумерация начинается с числа 0. То есть если S = 'Hello', то S[0] == 'H', S[1] == 'e', S[2] == 'l', S[3] == 'l', S[4] == 'o'.

Заметим, что в Питоне нет отдельного типа для символов строки. Каждый объект, который получается в результате среза S[i] – это тоже строка типа str.

Номера символов в строке (а также в других структурах данных: списках, кортежах) называются индексом.

Если указать отрицательное значение индекса, то номер будет отсчитываться с конца, начиная с номера -1. То есть S[-1] == 'o', S[-2] == 'l', S[-3] == 'l', S[-4] == 'e', S[-5] == 'H'.

Или в виде таблицы:

Строка S	H	e	l	l	o
Индекс	S[0]	S[1]	S[2]	S[3]	S[4]
Индекс	S[-5]	S[-4]	S[-3]	S[-2]	S[-1]

Если же номер символа в срезе строки S больше либо равен len(S), или меньше, чем -len(S), то при обращении к этому символу строки произойдет ошибка IndexError: string index out of range.

Срез с двумя параметрами: S[a:b] возвращает подстроку из b - a символов, начиная с символа с индексом a, то есть до символа с индексом b, не включая его. Например, S[1:4] == 'ell', то же самое получится если написать S[-4:-1]. Можно использовать как положительные, так и отрицательные индексы в одном срезе, например, S[1:-1] – это строка без первого и последнего символа (срез начинается с символа с индексом 1 и заканчивается индексом -1, не включая его).

При использовании такой формы среза ошибки IndexError никогда не возникает. Например, срез S[1:5] вернет строку 'ello', таким же будет результат, если сделать второй индекс очень большим, например, S[1:100] (если в строке не более 100 символов).

Если опустить второй параметр (но поставить двоеточие), то срез берется до конца строки. Например, чтобы удалить из строки первый символ (его индекс равен 0), можно взять срез S[1:]. Аналогично если опустить первый параметр, то можно взять срез от начала строки. То есть удалить из строки последний символ можно при помощи среза S[:-1]. Срез S[:] совпадает с самой строкой S.

Любые операции среза со строкой создают новые строки и никогда не меняют исходную строку. В Питоне строки вообще являются неизменяемыми, их невозможно изменить. Можно лишь в старую переменную присвоить новую строку.

На самом деле в питоне нет и переменных. Есть лишь имена, которые связаны с какими-нибудь объектами. Можно сначала связать имя с одним объектом, а потом – с другим. Можно несколько имён связать с одним и тем же объектом.

Если задать срез с тремя параметрами S[a:b:d], то третий параметр задает шаг, как в случае с функцией range, то есть будут взяты символы с индексами a, a + d, a + 2 * d и т. д. При задании значения третьего параметра, равному 2, в срез попадет каждый второй символ, а если взять значение среза, равное -1, то символы будут идти в обратном порядке. Например, можно перевернуть строку срезом S[::-1].

```
s = 'abcdefg'
print(s[1])
print(s[-1])
print(s[1:3])
print(s[1:-1])
print(s[:3])
```

```
print(s[2:])
print(s[:-1])
print(s[::2])
print(s[1::2])
print(s[::-1])
```

Обратите внимание на то, как похож третий параметр среза на третий параметр функции range():

```
s = 'abcdefghijklm'
print(s[0:10:2])
for i in range(0, 10, 2):
    print(i, s[i])
```

2.3 Методы

Метод – это функция, применяемая к объекту, в данном случае – к строке. Метод вызывается в виде Имя_объекта.Имя_метода(параметры). Например, S.find("e") – это применение к строке S метода find с одним параметром "e".

```
S = 'Hello'
print(S.find('e'))
# вернёт 1
print(S.find('ll'))
# вернёт 2
print(S.find('L'))
# вернёт -1
```

2.3.1 Методы find и rfind

Метод find находит в данной строке (к которой применяется метод) данную подстроку (которая передается в качестве параметра). Функция возвращает индекс первого вхождения искомой подстроки. Если же подстрока не найдена, то метод возвращает значение -1.

Аналогично, метод rfind возвращает индекс последнего вхождения данной строки («поиск справа»).

```
S = 'Hello'
print(S.find('l'))
# вернёт 2
print(S.rfind('l'))
# вернёт 3
```

Если вызвать метод find с тремя параметрами S.find(T, a, b), то поиск будет осуществляться в срезе S[a:b]. Если указать только два параметра S.find(T, a), то поиск будет осуществляться в срезе S[a:], то есть начиная с символа с индексом a и до конца строки. Метод S.find(T, a, b) возвращает индекс в строке S, а не индекс относительно среза.

2.3.2 Метод replace

Метод replace заменяет все вхождения одной строки на другую. Формат: S.replace(old, new) – заменить в строке S все вхождения подстроки old на подстроку new. Пример:

```
print('Hello'.replace('l', 'L'))
# вернёт 'HeLLo'
```

Если методу replace задать еще один параметр: S.replace(old, new, count), то заменены будут не все вхождения, а только не больше, чем первые count из них.

```
print('Abrakadabra'.replace('a', 'A', 2))
# вернёт 'AbrAkAdabra'
```

2.3.3 Метод count

Подсчитывает количество вхождений одной строки в другую строку. Простейшая форма вызова S.count(T) возвращает число вхождений строки T внутри строки S. При этом подсчитываются только непересекающиеся вхождения, например:

```
print('Abracadabra'.count('a'))
# вернёт 4
print(('a' * 10).count('aa'))
# вернёт 5
```

При указании трех параметров S.count(T, a, b), будет выполнен подсчет числа вхождений строки T в срезе S[a:b].

3 Порядок выполнения работы

Получить задание для выполнения лабораторной работы (раздел 4) согласно своему варианту (по журналу). Разработать программу.

4 Задания для выполнения работы

Задание 1. Делаем срезы.

Дана строка.

Сначала выведите третий символ этой строки.

Во второй строке выведите предпоследний символ этой строки.

В третьей строке выведите первые пять символов этой строки.

В четвертой строке выведите всю строку, кроме последних двух символов.

В пятой строке выведите все символы с четными индексами (считая, что индексация начинается с 0, поэтому символы выводятся начиная с первого).

В шестой строке выведите все символы с нечетными индексами, то есть начиная со второго символа строки.

В седьмой строке выведите все символы в обратном порядке.

В восьмой строке выведите все символы строки через один в обратном порядке, начиная с последнего.

В девятой строке выведите длину данной строки.

Задание 2. Количество слов.

Дана строка, состоящая из слов, разделенных пробелами. Определите, сколько в ней слов. Используйте для решения задачи метод count.

Задание 3. Две половинки.

Дана строка. Разрежьте ее на две равные части (если длина строки – четная, а если длина строки нечетная, то длина первой части должна быть на один символ больше). Переставьте эти две части местами, результат запишите в новую строку и выведите на экран.

При решении этой задачи не стоит пользоваться инструкцией if.

Задание 4. Переставить два слова.

Дана строка, состоящая ровно из двух слов, разделенных пробелом. Переставьте эти слова местами. Результат запишите в строку и выведите получившуюся строку.

При решении этой задачи не стоит пользоваться циклами и инструкцией if.

Задание 5. Первое и последнее вхождения.

Дана строка. Если в этой строке буква f встречается только один раз, выведите её индекс. Если она встречается два и более раз, выведите индекс её первого и последнего появления. Если буква f в данной строке не встречается, ничего не выводите.

При решении этой задачи не стоит использовать циклы.

Задание 6. Второе вхождение.

Дана строка. Найдите в этой строке второе вхождение буквы f, и выведите индекс этого вхождения. Если буква f в данной строке встречается только один раз, выведите число -1, а если не встречается ни разу, выведите число -2.

Задание 7. Удаление фрагмента.

Дана строка, в которой буква h встречается минимум два раза. Удалите из этой строки первое и последнее вхождение буквы h, а также все символы, находящиеся между ними.

Задание 8. Обращение фрагмента.

Дана строка, в которой буква h встречается как минимум два раза. Разверните последовательность символов, заключенную между первым и последним появлением буквы h, в противоположном порядке.

Задание 9. Замена подстроки.

Дана строка. Замените в этой строке все цифры 1 на слово one.

Задание 10. Удаление символа.

Дана строка. Удалите из этой строки все символы @.

Задание 11. Замена внутри фрагмента.

Дана строка. Замените в этой строке все появления буквы h на букву H, кроме первого и последнего вхождения.

Задание 12. Удалить каждый третий символ.

Дана строка. Удалите из нее все символы, чьи индексы делятся на 3.