**43-45 Символы Строки Срезы**

**Символы**

Любой текст состоит из символов.

Вместо символов в компьютере хранятся их номера.

Стандарт, указывающий, какие числовые коды соответствуют каким основным символам, называется ASCII. В таблицу ASCII включены символы с кодами от 0 до 127, то есть ASCII - это семибитный код.

В настоящее время для кодирования символов используется стандарт Unicode версия 6.0 которого (октябрь, 2010) включает свыше 109000 различных символов.

Язык программирования Python — современный язык, поэтому он работает исключительно с Unicode-символами.

Код символа можно определить при помощи функции **ord**. Эта функция получает на вход строку, которая должна состоять ровно из одного символа. Функция возвращает код этого символа.

Например, ord('A') вернет число 65.

Обратная функция получения по числовому коду его номера называется **chr**.

**Сравнение символов**

Поскольку для символов заданы их числовые коды, то их можно сравнивать при помощи операций сравнения. Поскольку символы алфавита идут подряд, то результат их сравнения будет соответствовать лексикографическому порядку, но можно сравнивать между собой не только буквы алфавита, но и два произвольных символа.

Также в питоне определены и операции сравнения строк, которые также сравниваются в лексикографическом порядке.

**Строки**

**Символьной строкой** называется набор нумерованных символов, которые хранятся в оперативной памяти компьютера под одним именем и располагаются последовательно друг за другом. Индексы символов строки, как и индексы элементов списка, начинаются с нуля.

Главное отличие символьной строки от списка состоит в том, что **символьная строка – это неизменяемая величина**. На практике это означает, что к символу строки можно обратиться, как и к элементу списка, указав его индекс, однако в отличие от элемента списка, символ строки таким образом изменять нельзя.

**Сложение или конкатенация строк**. Она записывается знаком арифметического сложения между строками.

**Умножение или дублирование строки -** символом умнжения (\*).

**Срезы (slices)**

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

Есть три формы срезов.

**1** Взятие одного символа строки, а именно, S[i] — это срез, состоящий из одного символа, который имеет номер i. 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. 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.

**2** Срез с двумя параметрами: S[a:b] возвращает подстроку из b - a символов, начиная с символа c индексом a, до символа с индексом b, не включая его. 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.**

**3** 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 третье значение среза, равное -1, то символы будут идти в обратном порядке. Например, можно перевернуть строку срезом S[::-1].

**Методы обработки строк**

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".

Рассмотрим несколько полезных методов для обработки строк.

**1,2** Методы **upper** и **lower** изменяют регистр всех символов строки, соответственно, на верхний и нижний.

**str = "это пример 1 строки....wow!!!"**

**print ("str.upper : ",str.upper())**

**str = "ЭТО ПРИМЕР 2 СТРОКИ....WOW!!!"**

**print ("str.lower : ",str.lower())**

**3** Логический метод **isdigit** проверяет, только ли цифры содержит строка. Он возвращает значение **«истина», если строка состоит только из цифр**, в противном случае будет возвращено значение «ложь». Для отрицательных чисел и для чисел с плавающей точкой этот метод будет возвращать значение «ложь».

**4** Часто бывает полезен метод **find**. Он возвращает **индекс элемента**, с которого начинается первое вхождение заданной подстроки в строку, или же -1, если заданной подстроки в строке не обнаружено. Для того, чтобы найти последнее вхождение подстроки в строку, можно использовать метод **rfind**.

|  |  |
| --- | --- |
| **find** | **rfind** |
| **S = 'Hello'**  **print(S.find('e')) # вернёт 1**  **print(S.find('ll')) # вернёт 2**  **print(S.find('L')) # вернёт -1** | **S = 'Hello'**  **print(S.find('l')) # вернёт 2**  **print(S.rfind('l')) # вернёт 3** |

Данные любого из основных типов можно преобразовать в символьную строку, для этого используется функция **str**. Для того, чтобы преобразовать строку в число, можно воспользоваться одной из уже известных нам функций преобразования – **int** или **float**. Однако, если использовать эту функцию для строки, содержимое которой не является числом, то исполнение программы будет завершено ошибкой несоответствия типов.

**5,6** Методы **split** и **join**, первый из них делит исходную строку на подстроки по заданной строке-разделителю. Результат этого метода – список подстрок исходной строки. Второй метод, напротив, соединяет список строк в одну строку, вставляя между ними заданную строку-разделитель.

**7** Метод **s.replace(а,b)**, который **заменяет** в исходной строке (s) все вхождения одной подстроки (a) на другую подстроку(b).

Если методу replace задать еще один параметр: **s.replace(а,b,x)**, то заменены будут не все вхождения, а только не больше, чем первые x из них.

|  |  |
| --- | --- |
| **replace(а,b)** | **replace(а,b,x)** |
| **print('Hello'.replace('l', 'L')) # вернёт HeLLo** | **print('abrakadabra'.replace('a', 'A', 2))**  **# вернёт 'AbrAkadabra'** |

**8** Метод **count.** Подсчитывает количество вхождений одной строки в другую строку. 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 5** | **print(('a' \* 10).count('aa')) # вернёт 5** |

**Задание 1: код ASCII-символа**

Считывайте со стандартного ввода символ и выводите его код до тех пор, пока не будет введен **q**.

**Задание 2: символ с данным ASCII-кодом**

Считывайте со стандартного ввода целое число и выведите ASCII-символ с таким кодом до тех пор, пока не будет введен **-1**. Решите эту задачу с использованием только одной переменной типа int.

**Задание 3 s = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'**

|  |  |  |  |
| --- | --- | --- | --- |
| **print(s[1])** | **print(s[:3])** | **print(s[::2])** | **print(s[::-1])** |
| **print(s[-1])** | **print(s[2:])** | **print(s[1::2])** |  |
| **print(s[1:3])** | **print(s[:-1])** | **print(s[1:-1])** |  |

**Задание 4** Дана строка с клавиатуры.

1. Сначала выведите третий символ этой строки.
2. Во второй строке выведите предпоследний символ этой строки.
3. В третьей строке выведите первые пять символов этой строки.
4. В четвертой строке выведите всю строку, кроме последних двух символов.
5. В пятой строке выведите все символы с четными индексами (считая, что индексация начинается с 0, поэтому символы выводятся начиная с первого).
6. В шестой строке выведите все символы с нечетными индексами, то есть начиная со второго символа строки.
7. В седьмой строке выведите все символы в обратном порядке.
8. В восьмой строке выведите все символы строки через один в обратном порядке, начиная с последнего.
9. В девятой строке выведите длину данной строки.

**Задание 1-1 Напишем модуль по формированию новой строки (по описанию алгоритма).**

В строке через запятую перечислены 4 слова. Сформировать из этих слов новую строку, в которой они будут перечислены через пробел и отсортированы по убыванию без учета регистра.

1. Вначале с помощью инструкции **print** выведем на экран запрос на ввод исходной строки. Далее с помощью функции **input** считаем исходную строку в переменную **s**.
2. Так как сортировать слова нужно без учета регистра, приведем строку **s** к нижнему регистру, используя метод **lower**.
3. Теперь сформируем 4 строковые переменные (s1,s2,s3,s4), состоящие из слов строки **s**. Для этого к строке **s** применим метод **split**, в котором в качестве разделителя укажем подстроку, состоящую из запятой.
4. Отсортируем переменные по убыванию и запишем их в этом порядке в другие переменные (с1=max1, с2=max2, c3=max3, c4=max4), воспользуемся операцией сравнения.
5. Сформируем из них результирующую строку, которую обозначим **r**. Для этого используем метод **join**, в котором в качестве строки-разделителя укажем символ «пробел».
6. С помощью инструкции **print** с соответствующим поясняющим сообщением выведем на экран результирующую строку.

**собака,кот,почтальон,гусь собака почтальон кот гусь**

**Задание 2**

|  |  |
| --- | --- |
| Что будет выведено на экран при исполнении инструкций? | **s = 'АРАРАТ'**  **s.replace ('АРА', 'ФА')**  **print (s[0:4])** |

**Задание 3**

|  |  |
| --- | --- |
| Что будет выведено на экран при исполнении инструкций? | **s, t = 'Владикавказ', 'Владивосток'**  **if t > s:**  **print (s)**  **else:**  **print (t)** |

**Задание 4** Сопоставьте методы символьных строк, с результатом, который они вернут при **s = 'Самолёт'**?

|  |  |  |  |
| --- | --- | --- | --- |
| **1** | **'САМОЛЁТ'** |  | **s.split ('о')** |
| **2** | **-1** |  | **s.upper ()** |
| **3** | **4** |  | **s.rfind ('Лёт')** |
| **4** | **'Сам', 'лёт'** |  | **s.isdigit ()** |
| **5** | **False** |  | **s.find ('лёт')** |

**Задание 5**

|  |  |
| --- | --- |
| В какой из инструкций допущена ошибка, если s = 'мама мыла раму'? | * **t = s[0:4]** * **t = s[3]** * **s[3] = ' '** * **s = s[0:4]** |

**Задание 6**

|  |  |
| --- | --- |
| Что будет выведено на экран при исполнении инструкций? | **s = '-5'**  **print (s.isdigit ())** |

**Задание 7**

|  |  |
| --- | --- |
| Что будет выведено на экран при исполнении инструкций? | **a = 'flimsy'**  **b = 'miserable'**  **c = b[0:1] + a[2:]**  **print(c)** |

**Задание 8**

Дана строка **s** и целые числа **a, b, c, d.** Вывести строку, полученную из **s**, начиная с символа с индексом **a** до **b**, и с **c** до **d** включительно (между “кусками” - пробел).

**Sample Input: abcdefghijklmnopqrstuvwxyz 2 5 8 15**

**Sample Output: cdef ijklmnop**

**Задание 9**

Есть строка **s = "abcdefghifdgffdgfgsddasdsfgjk"**.

Запишите строки (без кавычек), полученные в результате следующих операций:

**s[3:7] s[:9] s[3:] s[::-1]**

**s[-4:] s[:-9] s[-1:-11:-2]**

**Задание 10** **«Число десятков» (PT)**

Дано натуральное число. Найдите число десятков в его десятичной записи.