

Индексы и срезы

Сегодня мы поговорим об операциях взятия индекса и среза. Номера символов в строке (а также в других структурах данных: списках, кортежах) называются индексом.

Каждому символу в строке соответствует индексный номер, начиная с 0.

Например:

```
a = [1, 3, 8, 7]
print(a[0])    #выведет 1
print(a[3])    #выведет 7
```

В Python также поддерживаются отрицательные индексы, при этом нумерация идёт с конца, например:

```
a = [1, 3, 8, 7]
```

```
print(a[-1])
```

#выведет 7

```
print(a[-4])
```

#выведет 1

Прямой доступ к строке

При использовании прямого доступа к строке, тогда нумерация начинается с нуля.

Пример:

```
word = 'привет'  
print(word[1])
```

#выведет р

В качестве примера прямого доступа к элементам строки напишем программу доступа к случайному символу строки. Листинг:

```
import random

word = input("Введите слово: ")
max = len(word)
min = -len(word)

for i in range(10):
    position = random.randrange(min, max)
    print("word[" + position + "] = " + word[position])
```

Позиции с положительными и отрицательными номерами

Напомним, что позиция, то есть индекс строки, может быть как положительный, так и отрицательный.

Как уже было отмечено, нумерация начинается с 0, поэтому у первого символа строки будет индекс 0, у второго -1 и т.д.

Рассмотрим пример с положительными и отрицательными индексами: Пользователь вводит слово. Далее мы вычисляем минимум и максимум. Максимум равен длине введенного слова, минимум – отрицательной длине.

```
word = input("Введите слово: ")
max = len(word)
min = -len(word)
for i in range(min, max, 1):
    print("word[" + i + "] = " + word[i])
input()
```

Неизменяемость строк

Все последовательности делятся на изменяемые и неизменяемые. Элементы неизменяемой последовательности можно модифицировать.

Элементы неизменяемой последовательности изменить нельзя. Строки в Python – это пример неизменяемой последовательности. Рассмотрим пример:

```
message = "Привет"  
print(message) # выведет: Привет  
message = "Пока"  
print(message) # выведет: Пока
```

На первый взгляд кажется, что мы изменили строку, но на самом деле Python создал строку “Пока” и присвоил это значение переменной `message`.

Неизменяемость строк в Python означает одну неприятную вещь: вы не можете присвоить новое значение символу строки, обращаясь к нему по индексу, как это можно сделать в других языках программирования. Пример:

```
message = "Hello"  
print(message[1]) # Выведет e  
message[1] = "a" #Выведет ошибку
```

В данном примере мы создали строку `message`. Мы можем обратиться к `message[1]` и прочитать это значение. Но нельзя присвоить `message[1]` новое значение. Интерпретатор сообщает что тип `str` не поддерживает поэлементное назначение.

Срезы

Срез (slice) — это 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 считая, что нумерация начинается с числа 0.

Исходя из вышесказанного если $S='Hello'$, то $S[0]== 'H'$, $S[1]== 'e'$, $S[2]== 'l'$, $S[3]== 'l'$, $S[4]== 'o'$.

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. При использовании такой формы среза ошибки `IndexError` никогда не возникает.

Если задать срез с тремя параметрами $S[a:b:d]$, то третий параметр задает шаг, 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.

Также с помощью срезов можно не только извлекать элементы, но и добавлять и удалять элементы:

```
a = [1, 3, 8, 7]
a[1:3] = [0, 0, 0]
print(a)      # [1, 0, 0, 0, 7]

del a[:-3]
print(a)      # [0, 0, 7]
```