Урок №8. Знакомство со списками

План урока

1. Списки.
2. Метод append.
3. Обращение по индексу и другие операции со списками.
4. Вложенные списки.

Аннотация

В уроке рассматриваются списки (list), обращение к элементам списка по индексу (аналогично с строкам, но элементы списков можно менять), метод append. Рассматриваются сначала списки чисел и строк, а затем списки списков.

Мы написали уже много программ, работающих с данными, количество которых неизвестно на момент написания программы. Это количество может меняться от запуска к запуску благодаря циклам. Было бы здорово ещё и единовременно хранить в памяти неизвестное на момент написания программы количество данных. В этом нам помогут **списки**. Список — это составной тип данных, представляющий собой несколько значений (элементов списка) под одним именем. Этот тип называется `list` — никогда не создавайте переменные с таким именем! Чтобы задать список, нужно в квадратных скобках перечислить его элементы (в частности, конструкция [] обозначает пустой список). Здесь создаётся список из пяти элементов (целых чисел), который помещается в переменную `primes` («простые числа»):

```
primes = [2, 3, 5, 7, 11]
print(primes)  # выводим на экран список целиком
```

Чтобы получить отдельный элемент списка, нужно записать после него (или имени переменой, связанной с данным списком) в квадратных скобках номер (индекс) нужного элемента. Индекс отсчитывается с нуля — как в строках. Также, как и в строках, для нумерации с конца разрешены отрицательные индексы.

Таким образом, мы умеем использовать **квадратные скобки** в Питоне уже для двух вещей: задания нового списка (тогда внутри этих скобок перечисляются его элементы) и получения элемента списка или строки по индексу.

```
print('Сумма первых двух простых чисел:', primes[0] + primes[1])
print('Последнее из простых чисел в нашем списке:', primes[-1])
```

Добавление элемента в конец списка делается при помощи append:

```
primes.append(13)
primes.append(15)  # ой, ошиблись
```

append — это что-то вроде функции, «приклеенной» к конкретному списку. Такие «приклеенные функции» называются **методами**. В данном случае append - это метод списка, а конкретно — метод объекта primes. Заметьте, что два вызова метода append в следующем примере добавляют элемент к двум разным спискам.

```
odd_numbers = [1, 3, 5, 7, 9, 11, 13, 15, 17]
primes.append(19)
odd_numbers.append(19)
```

В отличие от отдельных символов в строках, элемент списка можно поместить слева от "=" в операторе присваивания и тем самым изменить этот элемент.

```
primes[6] = 17  # Исправляем ошибку:
#  седьмое (нумерация элементов списка - с нуля!)
#  простое число - не 15, а 17.
```

Тем не менее, многие вещи, которые можно делать со строками, можно делать и со списками:

```
print(len(primes))  # выводим длину списка
primes += [23, 29]  # списки можно складывать, как и строки
print(primes)  # выведет [2, 3, 5, 7, 11, 13, 17, 19, 23, 29]
if 1 in primes:  # можно проверять, содержится ли в списке элемент
    print('Мы считаем единицу простым числом.')
else:
    print('Мы, как и всё остальное человечество, не считаем 1 простым числом.')
```

На момент выполнения программы количество элементов списка известно. Поэтому для того, чтобы сделать что-то со всеми элементами списка, мы можем использовать цикл for. Заметьте, что индексы перебираются в цикле очень удачно: от 0 включительно до длины списка не включительно.

```
for i in range(len(primes)):
    print(primes[i]) # выведем по очереди все элементы списка
```

Более того, тот же цикл нередко используется и для формирования списка, если мы заранее знаем, сколько элементов в нём должно быть:

```
n = 10
a = []
print('Введите', n, 'значений:')
for i in range(n):
    a.append(input())
print('Получился список строк:', a)
```

Задачи:

- **«Список покупок» (https://lms.yandexlyceum.ru/task/view/940),**
- **«Подробный список покупок» (https://lms.yandexlyceum.ru/task/view/941),**
- **«Пра-пра-пра-Яндекс» (https://lms.yandexlyceum.ru/task/view/942),**
- **«Буква каждого слова» (https://lms.yandexlyceum.ru/task/view/943),**
- **«Автоматическое объявление» (https://lms.yandexlyceum.ru/task/view/944),**
- **«Не бином Ньютона» (https://lms.yandexlyceum.ru/task/view/945)**

Необязательные задачи:

- **«Пра-пра-пра-Яндекс — 2» (https://lms.yandexlyceum.ru/task/view/950),**
- **«A272727» (https://lms.yandexlyceum.ru/task/view/951),**
- **«Напёрстки» (https://lms.yandexlyceum.ru/task/view/952).**

Элементами списка могут быть значения любого типа: целые и действительные числа, строки и даже другие списки.

```
a = [[3, 4, 5], [10, 20, 30]]
print(a[0])  # [3, 4, 5]
print(a[0][1])  # 4
print(a[1][0])  # 10
```

Во многих языках программирования (да и в самом Питоне — в недрах стандартной библиотеки) имеется другой тип данных с похожими свойствами — **массив**. Поэтому списки иногда называют массивами, хоть это и не совсем правильно.