11.1 Введение в списки

Шаг 1

Тема урока: введение в списки

1. Создание списков
2. Пустые списки
3. Встроенная функция `list()`
4. Вывод списков

Аннотация. Списки как сохранение последовательностей и аналог массивов.

Списки

В предыдущих уроках мы работали с последовательностями чисел, символов, строк, но не сохраняли всю последовательность в памяти компьютера, а обрабатывали ее поэлементно, считывая раз за разом новый элемент. Однако во многих задачах требуется **сохранять всю последовательность**. Например, классическая задача сортировки (https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B8) (упорядочения) некоторой последовательности требует сохранения всех данных в памяти компьютера. Увы, не сохранив, их невозможно отсортировать. И тут на помощь приходит структура данных, которая в большинстве языков программирования называется массивом (https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%81%D1%81%D0%B8%D0%B2_(%D1%82%D0%B8%D0%BF_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85)). В Python она называется **списком**.

Структура данных (data structure) — программная единица, позволяющая **хранить и обрабатывать** множество однотипных и/или логически связанных данных.

Список представляет собой последовательность элементов, пронумерованных от 0, как символы в строке.

Создание списка

Чтобы создать список, нужно перечислить его элементы через запятую в квадратных скобках:

```
numbers = [2, 4, 6, 8, 10]
languages = ['Python', 'C#', 'C++', 'Java']
```

Список `numbers` состоит из 5 элементов, и каждый из них — целое число.

- `numbers[0] == 2;`
- `numbers[1] == 4;`
- `numbers[2] == 6;`
- `numbers[3] == 8;`
- `numbers[4] == 10.`

Список `languages` состоит из 4 элементов, каждый из которых — **строка.**

- `languages[0] == 'Python';`
- `languages[1] == 'C#';`
- `languages[2] == 'C++';`
- `languages[3] == 'Java'.`

Значения, заключенные в квадратные скобки и отделенные запятыми, называются **элементами списка**.

Список может содержать значения **разных типов данных**:

```
info = ['Timur', 1992, 61.5]
```

Список `info` содержит строковое значение, целое число и число с плавающей точкой.

- `info[0] == 'Timur';`
- `info[1] == 1992;`
- `info[2] == 61.5.`

Обычно элементы списка содержат данные одного типа, и на практике редко приходится создавать списки, содержащие элементы разных типов данных.

Пустой список

Создать пустой список можно двумя способами:

1. Использовать пустые квадратные скобки `[]`;
2. Использовать встроенную функцию, которая называется `list`.

Следующие две строки кода создают пустой список:

```
mylist = []  # пустой список
mylist = list()  # тоже пустой список
```

Вывод списка

Для вывода всего списка можно применить функцию `print()`:

```
numbers = [2, 4, 6, 8, 10]
languages = ['Python', 'C#', 'C++', 'Java']
print(numbers)
print(languages)
```

Функция `print()` выводит на экран элементы списка, в квадратных скобках, разделенные запятыми:

```
[2, 4, 6, 8, 10]
['Python', 'C#', 'C++', 'Java']
```

Обратите внимание, что вывод списка содержит квадратные скобки. Позже мы научимся выводить элементы списка в более удобном виде с помощью циклов или с помощью распаковки.

Встроенная функция list

Python имеет встроенную функцию `list()`, которая помимо создания пустого списка может преобразовывать некоторые типы объектов в списки.

Например, мы знаем, что функция `range()` создает последовательность целых чисел в заданном диапазоне. Для преобразования этой последовательности в список, мы пишем следующий код:

```
numbers = list(range(5))
```

Во время исполнения этого кода происходит следующее:

1. Вызывается функция `range()`, в которую в качестве аргумента передается число 5;
2. Эта функция возвращает последовательность чисел `0, 1, 2, 3, 4`;
3. Последовательность чисел `0, 1, 2, 3, 4` передается в качестве аргумента в функцию `list()`;
4. Функция `list()` возвращает список `[0, 1, 2, 3, 4]`;
5. Список `[0, 1, 2, 3, 4]` присваивается переменной `numbers`.

Вот еще один пример:

```
even_numbers = list(range(0, 10, 2))  # список содержит четные числа 0, 2, 4, 6, 8
odd_numbers = list(range(1, 10, 2))  # список содержит нечетные числа 1, 3, 5, 7, 9
```

Точно так же с помощью функции `list()` мы можем создать список из символов строки. Для преобразования строки в список мы пишем следующий код:

```
s = 'abcde'
chars = list(s)  # список содержит символы 'a', 'b', 'c', 'd', 'e'
```

Во время исполнения этого кода происходит следующее:

1. Вызывается функция `list()`, в которую в качестве аргумента передается строка `'abcde'`;
2. Функция `list()` возвращает список `['a', 'b', 'c', 'd', 'e']`;
3. Список `['a', 'b', 'c', 'd', 'e']` присваивается переменной `chars`.

Примечания

Примечание 1. Как уже было сказано, списки в Python аналогичны массивам в других языках программирования. Однако разница между списками и массивами все же существует. Элементы массива всегда имеют одинаковый тип данных и располагаются в памяти компьютера непрерывным блоком, а элементы списка могут быть разбросаны по памяти как угодно и могут иметь разный тип данных.

Примечание 2. Обратите внимание, при выводе содержимого списка с помощью функции `print()`, все строковые элементы списка обрамляются **одинарными кавычками**. Если требуется осуществить вывод в двойных кавычках, нужно самостоятельно писать код вывода.

❤️ Happy Pythoning!

Шаг 2

Значения в списках, заключённые в квадратные скобки и отделённые запятыми, называются

Чтобы решить это задание откройте https://stepik.org/lesson/324750/step/2

Шаг 3

Из скольких элементов состоит список `numbers`?

```
numbers = [3, 5, 7, 9]
```

Чтобы решить это задание откройте https://stepik.org/lesson/324750/step/3

Шаг 4

Какой индекс у числа 17 в списке `numbers`?

```
numbers = [1, 100, 7, 20, 17, 37, 22]
```

Чтобы решить это задание откройте https://stepik.org/lesson/324750/step/4

Шаг 5

Может ли список в Python содержать значения разных типов данных?

Чтобы решить это задание откройте https://stepik.org/lesson/324750/step/5

Шаг 6

Что покажет приведённый ниже код?

```
numbers = [0, 1, 3, 14, 2, 7, 9, 8, 10]
print(numbers)
```

Чтобы решить это задание откройте https://stepik.org/lesson/324750/step/6

Шаг 7

Что покажет приведённый ниже код?

```
names = ['Michael', 'John', 'Freddie']
print(names)
```

Чтобы решить это задание откройте https://stepik.org/lesson/324750/step/7

Шаг 8

Список чисел

На вход программе подаётся одно число n. Напишите программу, которая выводит список `[1, 2, 3, ..., n]`.

Формат входных данных

На вход программе подаётся одно натуральное число.

Формат выходных данных

Программа должна вывести текст в соответствии с условием задачи.

▶ Тестовые данные

Sample Input 1:

```
1
```

Sample Output 1:

```
[1]
```

Sample Input 2:

```
5
```

Sample Output 2:

```
[1, 2, 3, 4, 5]
```

Sample Input 3:

```
10
```

Sample Output 3:

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

Чтобы решить это задание откройте https://stepik.org/lesson/324750/step/8

Шаг 9

Список букв

На вход программе подаётся натуральное число n. Напишите программу, которая выводит список, состоящий из n букв английского алфавита `['a', 'b', 'c', ...]` в нижнем регистре.

Формат входных данных

На вход программе подаётся натуральное число n $(1 \leq n \leq 26)$.

Формат выходных данных

Программа должна вывести текст в соответствии с условием задачи.

▶ Тестовые данные

Sample Input 1:

```
1
```

Sample Output 1:

```
['a']
```

Sample Input 2:

```
5
```

Sample Output 2:

```
['a', 'b', 'c', 'd', 'e']
```

Sample Input 3:

```
10
```

Sample Output 3:

```
['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
```

Чтобы решить это задание откройте https://stepik.org/lesson/324750/step/9