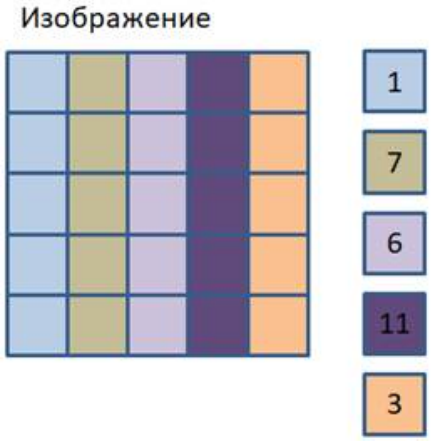
Вложенные списки, многомерные списки

Это заключительное занятие по спискам языка Python. Сегодня мы с вами узнаем, как формировать вложенные списки и работать с ними. Но сначала, что это такое и зачем они нужны.

Давайте представим, что нам в программе нужно хранить изображение. Для примера я нарисовал его небольшим, всего 5 на 5 пикселей. Каждый цвет представляется своим уникальным числом. Я, условно, обозначил их 1, 7, 6, 11 и 3. Значит, для представления этих данных нам нужен двумерный список 5x5 с соответствующими числовыми значениями. Мы уже знаем, как задавать одномерный список:

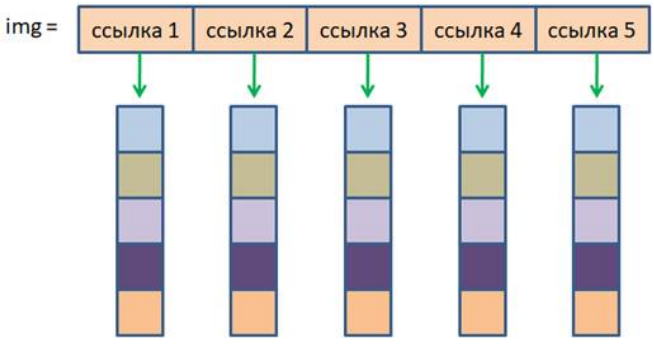



```
line = [1, 7, 6, 11, 3]
```

Но так он описывает всего лишь одну строку. А нам нужно хранить пять таких строк. Учитывая, что элементом списка может быть другой список, то данное изображение можно задать так:

```
img = [[1, 7, 6, 11, 3], [1, 7, 6
```

Мы здесь внутри первого списка определили пять вложенных и в результате получили двумерный список. Кстати, его можно было бы сформировать и проще, учитывая, что все вложенные списки одинаковы, на основе списка line, следующим образом:




```
img = [line[:], line[:], line[:],
```

В итоге получим такой же список с независимыми строками:

```
[[1, 7, 6, 11, 3], [1, 7, 6, 11,
```

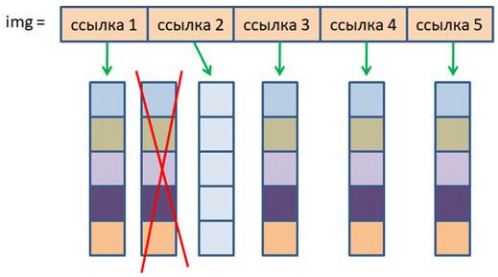
Вот эта последняя запись нам показывает структуру представления многомерных данных на уровне списков. Первый главный список хранит ссылки на вложенные списки. А вложенные списки уже хранят ссылки на соответствующие числа, представляющие тот или иной цвет. Поэтому, если взять первый элемент главного списка: `img[0]`

то мы получим список, представляющий первую строку (или, первый столбец в зависимости от интерпретации программистом этих данных). Главное, что мы получаем доступ к первому вложенному списку. А раз это так, то можно записать еще одни квадратные скобки и из этого вложенного списка взять, допустим, второй элемент: `img[0][1]`

Также, можно заменить, например, вторую строку на новую, допустим, такую: `img[1] = [0, 0, 0, 0, 0]`

или, то же самое, в более краткой форме: `img[1] = [0] * 5`

Что в итоге здесь произошло? Мы сформировали новый объект –список из нулей, связали с ним вторую ссылку главного списка, а прежний список был автоматически удален сборщиком мусора.



Если бы мы хотели изменить значения уже существующего вложенного списка, то следовало бы обратиться к его элементам, например, через механизм срезов:

```
img[1][:] = [0] * 5
```

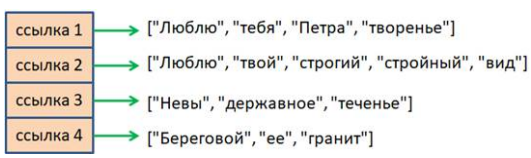
и присвоить его элементам новые числовые значения. Вот так это работает в деталях.

В качестве второго примера мы представим вложенными списками строки известного стихотворения на уровне отдельных слов:

Люблю тебя, Петра творенье,
Люблю твой строгий, стройный вид,
Невы державное теченье,
Береговой ее гранит,

Здесь в каждой строке разное число слов, но для вложенных списков – это не проблема. Они могут иметь разное число элементов:

```
t = [["Люблю", "тебя", "Петра", "
     ["Люблю", "твой", "строгий",
     ["Невы", "державное", "течен
     ["Береговой", "ее", "гранит"
     ]
```



В результате получаем следующую структуру наших данных. Здесь также для доступа к первой строке достаточно указать первый индекс: `t[0]`

а к отдельному слову этой строки, второй индекс:

```
t[0][2]
```

Если же мы хотим изменить какое-либо слово, то это делается, следующим образом:

```
t[0][2] = "Питон"
```

В итоге, первая строка принимает вид:

['Люблю', 'тебя', 'Питон', 'творенье']

Мало того, мы можем добавить новую строку, используя известный метод:

```
t.append(["Твоих", "оград", "узор
```

Удалять список: `del t[1]`

И так далее, то есть, делать с вложенными списками все те же операции, что и с обычными данными.

В заключение покажу пример многомерного списка с разными уровнями глубины:

```
A = [[[True, False], [1, 2, 3]],
```

Смотрите, здесь `A[0]`

это двумерный список, а `A[1]`

одномерный вложенный список. Соответственно, для третьего уровня вложенности можем использовать три индекса для доступа к отдельному элементу, например: `A[0][1][0]`

Здесь мы берем первый элемент, затем второй вложенный список и из него выбираем первый элемент