# Урок 13.02. Срезы списков.

При работе со списками, а на самом деле не только с ними, приходится создавать их копии, а ещё чаще – нужный для работы кусок из них. И сегодня мы рассмотрим какие инструменты есть у программиста для такой работы.

Рассмотрим задачу «Клиент-сервер»:

Мы тестируем клиент-серверную систему у себя на компьютере. Сервер генерирует числа, которые делятся на 10, начинаю с 1 и заканчивая 100. Нам нельзя напрямую менять данные, которые генерирует сервер, но нам нужно занулить 3 элемент, и вывести элементы списка с индекса 2 до индекса 7. Давайте посмотрим код такой программы:

nums = [x for x in range(1, 101) if x % 10 == 0]

new\_nums = nums

new\_nums[3] = 0

for i\_elem in range(2, 8):

print(new\_nums[i\_elem], end=' ')

Вроде всё в порядке, и нет никаких проблем. Но давайте для сравнения выведем на экран все элементы первого списка.

nums = [x for x in range(1, 101) if x % 10 == 0]

new\_nums = nums

new\_nums[3] = 0

for i\_elem in range(2, 8):

print(new\_nums[i\_elem], end=' ')

print()

print(nums)

Так! И что же мы видим?! Кажется что-то пошло не так, наш изначальный список изменился. Почему же так получается? Если провести дебаггинг (от английского слова debug – отлаживать программу), то при выполнении строчки с обнулением элемента с индексом 3 сразу у обоих списков обнуляется этот элемент. И тут стоит приоткрыть вас еще одну тайну работы со списками: на самом деле у нас не два списка (nums и new\_nums), а один список, точнее его отображение в памяти компьютера. И в строчке где мы приравниваем один список другому (new\_nums = nums) мы на самом деле создаем еще одну переменную с ссылкой на один и тот же список. И теперь получается у одного списка два названия, и копию мы в прямом смысле этого не делаем. А вот для создания копии нам нужно всего лишь сделать вот так (добавить скобки и двоеточие между ними):

nums = [x for x in range(1, 101) if x % 10 == 0]

new\_nums = nums[:]

new\_nums[3] = 0

for i\_elem in range(2, 8):

print(new\_nums[i\_elem], end=' ')

print()

print(nums)

Запустим – всё работает. На самом деле это двоеточие в внутри скобок называется срезом. И тут название раскрывает еще один удобный способ работы со списками: это двоеточие используется, как указание, какой кусочек списка нам нужен. Давайте этим воспользуемся. Сотрем наш цикл и добавим следующую строчку:

nums = [x for x in range(1, 101) if x % 10 == 0]

new\_nums = nums[:]

new\_nums[3] = 0

print(new\_nums[2:8])

print(nums)

Запустим. Всё получилось. Обращаю ваше внимание на конечный индекс (он на 1 больше как у range). И тут есть еще одно сходство с функцией range, также есть [start:stop:step]. Так, чтобы вывести все элементы через один со второго можно сделать вот так:

nums = [x for x in range(1, 101) if x % 10 == 0]

print(nums[1::2])

Здесь мы видим, что если не указывать индекс то он берется по умолчанию. Для start – это первый элемент или индекс «0», для stop – последний элемент или по-другому ещё индекс «‑1», а для step – это число 1. Для переворачивания списка нужно лишь указать step «‑1» nums[::-1]. На самом деле срезы списков можно заменять на один элемент или несколько элементов. Например вот так:

nums = [x for x in range(1, 101) if x % 10 == 0]

nums[3:6] = [7]

print(nums)

Вот так.