ИНДЕКСЫ

1. **Индекс** (в контексте строк) – целочисленный номер позиции символа в строке.

Индексы считаются с нуля!

- первый символ в строке имеет индекс, равный **0**

- последний символ в строке имеет индекс, равный **Длина строки - 1**

***my\_course = "PYTHON"***

1. Получение первого символа:

***first\_letter = my\_course[0]  
  
# first\_letter: "P"***

2. Получение последнего символа:

***last\_letter = my\_course[len(my\_name) – 1]***

***# last\_letter: "N"***

3. Так как Python поддерживает **отрицательные индексы,** то получение последнего символа строки можно записать как:

***last\_letter = my\_course[-1]****(предпочтительная форма записи)*

Наглядный пример:

# прямое индексирование

**[0] [1] [2] [3] [4] [5]**

**P Y T H O N**

**[-6][-5][-4][-3][-2][-1]**

# обратное индексирование

4. При попытке получить символ по индексу, которого нет - получаем ошибку

***my\_course[6]  
my\_course[-7]  
  
# Ошибка: IndexError: string index out of range***

СРЕЗЫ (Slices)

1. Назначение (в контексте строк): получение подстроки из исходной строки, через указание верхней и нижней границ исходной строки.

Общая форма среза:

**string\_object[START:STOP:STEP]**

**- START - нижняя граница среза**

**- STOP - верхняя граница среза**

**- STEP - шаг (по умолчанию равен 1)**

**ВАЖНО!**

- Нижняя граница среза входит в результат

- Верхняя граница среза НЕ входит в результат!

***first\_two\_letters = my\_course[0:2]***

# ***first\_two\_letters:* "PY"**  
(эквивалентно выражению ***my\_course[0] + my\_course[1])***

***first\_three\_letters = my\_course[0:3]***

# ***first\_three\_letters:* "PYT"**  
(эквивалентно выражению ***my\_course[0] + my\_course[1] + my\_course[2])***

2. Нижнюю и Верхнюю границу можно НЕ указывать явно:

***my\_course[0:2]*** эквивалентно ***my\_course[:2]***

Получение подстроки со 2-го символа и до конца исходной строки (все слово без самой первой буквы)

***without\_first\_letter = my\_course[1:]***   
# ***without\_first\_letter:* "YTHON"**

Если не указать ОБЕ границы среза - получим копию исходной строки:

***word\_copy = my\_course[:]***

***# word\_copy: "PYTHON"****(Так как строки в Python являются НЕИЗМЕНЯЕМЫМИ объектами, получение копии строки, попросту, не имеет смысла. Язык позволяет это сделать, так как синтаксис срезов одинаков как для строк, так и для списков (списки – ИЗМЕНЯЕМЫЕ Объекты))*

3. Срезы также поддерживают обратное индексирование (через отрицательные индексы)

4. **ШАГ среза**

Шаг среза по умолчанию равен 1. То есть, при получении среза с шагом, равным 1, шаг можно не указывать вообще (в большинстве реальных случаев использования нужен именно шаг, равный 1)

При получении среза с шагом, равным 1, из исходной строки возьмутся **все символы подряд** в заданном диапазоне границ   
(это наглядно видно на предыдущих примерах, где мы не указывали шаг)

При получении среза с шагом, равным 2, из исходной строки символы будут браться **через 1,** в заданном диапазоне границ.

***my\_string = "aBaBaBaB"***

***result = my\_string1[::2]***

*(нижнюю и верхнюю границы срезов не указываем явно, указываем только шаг)*

***# result:* "aaaa"**

При получении среза с шагом, равным 3, из исходной строки символы будут браться **через 2,** в заданном диапазоне границ.  ***my\_string = "aBCaBCaBCaBC"***

***result = my\_string2[::3]***

***# result:* "aaaa"**

Если нам нужно получить символы исходной строки в ОБРАТНОМ порядке, нужно использовать шаг, равный **-1**

5. Срез строки можно "склеивать" с другим срезом строки, используя оператор **+ (конкатенация строк)**, в результате получаем новую строку:

***result = my\_course[:3] + my\_course[3:]***

# ***result:"PYTHON"***

ДОМАШНЕЕ ЗАДАНИЕ

**1**. **Дана строка:** ***"aaaaaBccBddBeeBffBggB"***

**Получить новую строку, состоящую из символов исходной строки. начиная с 5-го Индекса и по последний Cимвол исходной строки, с шагом, равным 3.**  
  
(в результате должно остаться только 6 символов **B**)  
  
**2. Дана строка: "AAAABBBBCCCCDDDDFFFF"  
  
- разбить строку на равные части по 4 символа с помощью срезов;  
- получить точную копию исходной строки через конкатенацию полученных срезов.**

(**Конкатенация** – “склеивание” строк через оператор **+**)

**\*3**. **Дана строка:** ***"PYTHON"***

**Получить новую строку, которая будет содержать все символы исходной строки, но в обратном порядке (справа налево) с помощью срезов.**

Для вывода результатов всех 3-х заданий использовать функцию ***print()***