**Занятие 4. Цикл for**

1. Цикл for

Цикл for, также называемый циклом с параметром, в языке Питон богат возможностями. В цикле for указывается переменная и множество значений, по которому будет пробегать переменная. Множество значений может быть задано списком, кортежем, строкой или диапазоном.

Вот простейший пример использования цикла, где в качестве множества значений используется кортеж:

**i = 1**

**for color in 'red', 'orange', 'yellow', 'green', 'cyan', 'blue', 'violet':**

 **print('#', i, ' color of rainbow is ', color, sep = '')**

 **i += 1**

В этом примере переменная color последовательно принимает значения 'red', 'orange' и т.д. В теле цикла выводится сообщение, которое содержит название цвета, то есть значение переменной color, а также номер итерации цикла  число, которое сначала равно 1, а потом увеличивается на один (инструкцией i += 1 с каждым проходом цикла.

Инструкция i += 1 эквивалентна конструкции i = i + 1 (это просто сокращенная запись). Такую сокращенную запись можно использовать для всех арифметических операций: \*=, -=, /=, %=...

В списке значений могут быть выражения различных типов, например:

**for i in 1, 2, 3, 'one', 'two', 'three':**

 **print(i)**

При первых трех итерациях цикла переменная i будет принимать значение типа int, при последующих трех — типа str.

### 2. Функция range

Как правило, циклы for используются либо для повторения какой-либо последовательности действий заданное число раз, либо для изменения значения переменной в цикле от некоторого начального значения до некоторого конечного.

Для повторения цикла некоторое заданное число раз n можно использовать цикл for вместе с функцией range:

**for i in range(4): # равносильно инструкции for i in 0, 1, 2, 3:**

 **# здесь можно выполнять циклические действия**

 **print(i)**

 **print(i \*\* 2)**

**# цикл закончился, поскольку закончился блок с отступом**

**print('Конец цикла')**

В качестве n может использоваться числовая константа, переменная или произвольное арифметическое выражение (например, 2 \*\* 10). Если значение n равно нулю или отрицательное, то тело цикла не выполнится ни разу.

Функция range может также принимать не один, а два параметра. Вызов range(a, b) означает, что индексная переменная будеть принимать значения от a до b - 1, то есть первый параметр функции range, вызываемой с двумя параметрами, задает начальное значение индексной переменной, а второй параметр — первое значение, которое индексная переменная принимать **не будет**. Если же a≥b, то цикл не будет выполнен ни разу. Например, для того, чтобы просуммировать значения чисел от 1 до n можно воспользоваться следующей программой:

**sum = 0**

**n = 5**

**for i in range(1, n + 1):**

 **sum += i**

**print(sum)**

В этом примере переменная i принимает значения 1, 2, ..., n, и значение переменной sum последовательно увеличивается на указанные значения.

Наконец, чтобы организовать цикл, в котором индексная переменная будет уменьшаться, необходимо использовать функцию range с тремя параметрами. Первый параметр задает начальное значение индексной переменной, второй параметр — значение, до которого будет изменяться индексная переменная (не включая его!), а третий параметр — величину изменения индексной переменной. Например, сделать цикл по всем нечетным числам от 1 до 99 можно при помощи функции range(1, 100, 2), а сделать цикл по всем числам от 100 до 1 можно при помощи range(100, 0, -1).

Более формально, цикл for i in range(a, b, d) при d > 0 задает значения индексной переменной i = a, i = a + d, i = a + 2 \* d и так для всех значений, для которых i < b. Если же d < 0, то переменная цикла принимает все значения i > b.

3. Настройка функции print()

По умолчанию функция print() принимает несколько аргументов, выводит их через пробел, после чего ставит перевод строки. Это поведение можно изменить, используя именованные параметры sep (разделитель) и end (окончание).

**print(1, 2, 3)**

**print(4, 5, 6)**

**print(1, 2, 3, sep=', ', end='. ')**

**print(4, 5, 6, sep=', ', end='. ')**

**print()**

**print(1, 2, 3, sep='', end=' -- ')**

**print(4, 5, 6, sep=' \* ', end='.')**

**Задачи**

### Задача «Ряд - 1»

Даны два целых числа A и B (при этом A ≤ B). Выведите все числа от A до B включительно.

### Задача «Ряд - 2»

Даны два целых числа A и В. Выведите все числа от A до B включительно, в порядке возрастания, если A < B, или в порядке убывания в противном случае.

### Задача «Ряд - 3»

Даны два целых числа A и В, A>BA>B. Выведите все нечётные числа от A до B включительно, в порядке убывания. В этой задаче можно обойтись без инструкции if.

### Задача «Сумма десяти чисел»

Дано 10 целых чисел. Вычислите их сумму. Напишите программу, использующую наименьшее число переменных.

### Задача «Сумма N чисел»

Дано несколько чисел. Вычислите их сумму. Сначала вводите количество чисел N, затем вводится ровно N целых чисел. Какое наименьшее число переменных нужно для решения этой задачи?

### Задача «Сумма кубов»

### По данному натуральному n вычислите сумму 13+23+33+...+n3.

### Задача «Факториал»

Факториалом числа n называется произведение 1 × 2 × ... × n. Обозначение: n!.

По данному натуральному n вычислите значение n!. Пользоваться математической библиотекой math в этой задаче запрещено.

### Задача «Сумма факториалов»

### По данному натуральном nn вычислите сумму 1!+2!+3!+...+n!1!+2!+3!+...+n!. В решении этой задачи можно использовать только один цикл. Пользоваться математической библиотекой math в этой задаче запрещено.

### Задача «Количество нулей»

### Дано N чисел: сначала вводится число N, затем вводится ровно N целых чисел. Подсчитайте количество нулей среди введенных чисел и выведите это количество. Вам нужно подсчитать количество чисел, равных нулю, а не количество цифр.

### Задача «Лесенка»

### По данному натуральному n ≤ 9 выведите лесенку из n ступенек, i-я ступенька состоит из чисел от 1 до i без пробелов.

### Задача «Потерянная карточка»

Для настольной игры используются карточки с номерами от 1 до N. Одна карточка потерялась. Найдите ее, зная номера оставшихся карточек.

Дано число N, далее N − 1 номер оставшихся карточек (различные числа от 1 до N). Программа должна вывести номер потерянной карточки.

Для самых умных: массивами и аналогичными структурами данных пользоваться нельзя.