**Циклы в Python**

В Python циклы позволяют повторять набор инструкций до выполнения определённых условий. Одним из самых популярных циклов является цикл for, который позволяет проходить по элементам заданного диапазона или коллекции.

**Цикл for**

Цикл for в Python используется для итерации по последовательностям, таким как списки, строки, диапазоны или другие итерируемые объекты.

**Пример цикла по списку**:

python

Копировать код

for элемент in [1, 2, 3, 4]:

print(элемент)

**Функция range()**

Функция range() создаёт диапазон чисел и часто используется в циклах for. Она может принимать один, два или три параметра:

1. **range(конец)** — создаёт последовательность от 0 до конец-1.
2. **range(начало, конец)** — создаёт последовательность от начало до конец-1.
3. **range(начало, конец, шаг)** — создаёт последовательность с шагом шаг.

**Примеры**:

python

Копировать код

for i in range(5):

print(i)

Выведет числа от 0 до 4.

python

Копировать код

for i in range(1, 10, 2):

print(i)

Выведет 1, 3, 5, 7, 9 (числа от 1 до 9 с шагом 2).

**Обратный цикл**

Чтобы организовать цикл с убывающим индексом, используется range() с отрицательным шагом.

**Пример**:

python

Копировать код

for i in range(10, 0, -1):

print(i)

Это выведет числа от 10 до 1.

**Настройка функции print()**

Функция print() в Python по умолчанию добавляет перевод строки после каждого вывода. Это можно изменить с помощью параметра end, который позволяет указать, что будет выводиться в конце каждой команды print.

**Пример**:

python

Копировать код

for i in range(5):

print(i, end=' ')

Вывод будет: 0 1 2 3 4 — все числа на одной строке, разделённые пробелом.

**Пример задачи с циклом**

**Задача 1**: Вывести числа от 1 до 10.

python

Копировать код

for i in range(1, 11):

print(i)

**Задача 2**: Вывести числа от 10 до 1 в обратном порядке.

python

Копировать код

for i in range(10, 0, -1):

print(i)

**Практические рекомендации**

* **Используйте range()**: Для создания последовательности чисел всегда удобно использовать функцию range() с нужным количеством параметров.
* **Контролируйте отступы**: Отступы важны для структуры цикла и условий, не забывайте соблюдать одинаковый отступ для команд внутри цикла.
* **Настройка вывода**: Используйте параметр end в функции print(), если нужно изменить поведение вывода, например, чтобы избежать переноса строки.