**Цикл while**

**Цикл while**

Цикл **while** («пока») — это так называемый цикл с условием или с предусловием.В цикле **for** мы явно указываем, какие значения будет принимать переменная i, например, 1 <= i <= 10. В цикле **while** можно задавать более сложные условия, например, i ^ 2 <= 10, кроме того можно использовать логические операции «И», «ИЛИ» и т.д.

Цикл **while** позволяет выполнить одну и ту же последовательность действий, пока проверяемое условие истинно. Условие записывается до тела цикла и проверяется до выполнения тела цикла. Как правило, цикл **while** используется, когда невозможно определить точное значение количества проходов исполнения цикла.

Синтаксис цикла **while** в простейшем случае выглядит так:

**while** условие:

 блок инструкций

При выполнении цикла **while** сначала проверяется условие. Если оно ложно, то выполнение цикла прекращается, и управление передаётся на следующую инструкцию после тела цикла **while** . Если условие истинно, то выполняются все инструкции из блока, после чего условие проверяется снова, и снова выполняется блок инструкций. Так продолжается до тех пор, пока условие будет истинно. Как только условие становится ложным, работа цикла завершается, и управление передаётся следующей инструкции после цикла.

**Пример:**

Следующий фрагмент программы напечатает на экран все целые числа, не превосходящие nn, и их сумму:

s = 0

i = 1

**while** i <= n:

 **print**(i)

 s += i

 i += 1

**print**(s)

**Пример:**

C предыдущей задачей однако мог бы справиться и цикл **for** . Рассмотрим более сложную задачу, с которой **for** уже не справится: вывести на экран все степени двойки, не превосходящие 100100. Её также удобно решать с помощью цикла **while**.

i = 1

**while** i <= 100:

 **print**(i)

 i \*= 2

После окончания цикла значение i будет уже больше, чем 100100, иначе цикл бы продолжился. Последнее напечатанное значение будет соответственно равно 64.64.

**Пример:**

Немного изменим задачу: найдём максимальную степень двойки, не превосходящую 100100.

p = 1

**while** p \* 2 <= 100:

 p \*= 2

**print**(p)

После окончания цикла значение переменной p будет максимальной степенью двойки, не превосходящей 100100, потому что на следующем шаге условие уже не будет выполняться.

**Пример:**

Вот еще один пример использования цикла **while** для определения количества цифр натурального числа nn и их суммы:

s = 0

n = int(input())

count = 0

**while** n > 0:

 count += 1

 s += n % 10

 n //= 10

**print**(count)

**print**(s)

В этом цикле мы отбрасываем по одной цифре числа, начиная с конца, что эквивалентно целочисленному делению на 1010 (n //= 10), при этом считаем в переменной count, сколько раз это было сделано, а в переменной s — сумму отброшенных цифр.

**Обработка последовательностей неизвестной длины**

**Обработка последовательностей неизвестной длины**

С помощью цикла **for** мы могли решать задачи, обрабатывающие последовательности числовых данных, например, посчитать сумму или произведение элементов некоторой последовательности с известным числом элементов.

Однако бывают случаи, когда заранее неизвестно число элементов последовательности, а ввод ограничен тем или иным образом. С использованием цикла **while** можно решать задачи для этого случая. Рассмотрим пример такой задачи.

**Пример:**

Дана последовательность натуральных чисел, заканчивающаяся нулем. Требуется найти произведение и количество этих чисел (ноль не должен участвовать в нахождении произведения).

Решение может выглядеть так:

count = 0

prod = 1

elem = int(input())

**while** elem != 0:

 count += 1

 prod \*= elem

 elem = int(input())

**print**(prod)

**print**(count)

Рассмотрим более сложную задачу.

**Пример:**

Дана последовательность неизвестной длины, требуется вывести первое число, которое встречается два раза подряд. Гарантируется, что такое число существует.

Для решения задачи потребуется хранить не только текущее считанное значение, но и предыдущее:

prev = int(input())

elem = int(input())

**while** elem != prev:

 prev = elem

 elem = int(input())

**print**(elem)