# Урок 04.02. Бесконечный цикл, логический типа, флаг

Ранее мы с вами уже сталкивались с такой штукой как «бесконечный цикл». Давайте вспомним задачку с отбрасыванием числа:

number = int(input('Введите число: '))

summ = 0

while number > 0:

 last\_number = number % 10

 summ += last\_number

 if last\_number == 5:

 print('Обнаружен разрыв!')

 break

print('Сумма:', summ)

Эта программа осуществляет выделение последней цифры, но не уменьшает наше изначальное число, поэтому данный цикл работает бесконечно. Т.е. логическая ошибка привела к тому, что наша программа не работает, а еще и «зависла». Кажется, что бесконечный цикл это плохо, однако это не всегда так.

Давайте рассмотрим другой пример: нам поручили постоянно тестировать сервера на предмет их зависания. Давайте для этого мы будем постоянно выводить число 5. Такая программа может выглядеть вот так:

number = 5

while number > 0:

 print(number)

Число 5 всегда больше 0, тогда эту программу можно сделать короче:

while 5 > 0:

 print('5')

Сравнение 5 > 0 всегда будет истинно, поэтому мы можем еще сократить нашу программу:

while True:

 print('5')

Такая программа будет самой короткой. И тут мы вспоминаем логические типы данных True – истина и False – ложь. Если посмотреть шире, то все программы, которые у нас запущены, являются бесконечными циклами. Вспомним нашу программу про суммирование чисел:

number = int(input('Введите число (0 – для выхода): '))

summ = 0

while number != 0:

 summ += number

 number = int(input('Введите число (0 – для выхода): '))

print(summ)

А если использовать бесконечный цикл:

summ = 0

while True:

 number = int(input('Введите число (0 – для выхода): '))

 summ += number

 if number == 0:

 break

print(summ)

## Программа с флагом

Эта такой типа построения цикла, где в качестве условия выхода используется переменная, которая может принимать два значения, либо True (истина) – цикл должен выполняться, либо False (ложь) – прекращение выполнения цикла.

Для этого перепишем предыдущую программу:

summ = 0

flag = True

while flag:

 number = int(input('Введите число (0 – для выхода): '))

 summ += number

 if number == 0:

 flag = False

print(summ)

В этой программе введена новая переменная flag, которая отвечает за выполнение цикла, пока эта переменная равна True – цикл выполняется, как только переменная изменила свое значение на False и производится проверка выполнения цикла, то цикл заканчивается.

На первый взгляд может показаться, что особой разницы между этими двумя исполнения программы нет, но иногда в программах необходимо использовать именно переменную флага.

## Цикл while со счетчиком

Помните, когда мы только проходили переменные, то делали задачу которая несколько раз выводила одно и тоже сообщение на экран. Давайте еще вспомним этот пример:

text = 'Python'

print(text)

print(text)

print(text)

print(text)

print(text)

Теперь мы уже стали гораздо лучше разбираться в программировании, и можем это сделать еще эффективнее. Тут у нас повторяющиеся строчки текста, а мы знаем, что для повторения одних и тех же действий есть циклы. Так давайте использовать цикл для вывода этого текста:

text = 'Python'

words\_count = 0

while words\_count < 5:

 print(text)

 words\_count += 1

Таким образом, мы с вами создали переменную-счетчик.

Давайте рассмотрим еще одну подобную задачу. Наши родственники сходили на рыбалку и наловили кучу куч рыбы. И теперь перетаскивают мешки в машину и взвешивают. Мы знаем, что они перетащили пять мешков. Давайте напишем программу, которая суммирует массу рыбы после каждого взвешивания:

bags = int(input('Сколько нужно перетащить мешков: '))

total\_weight = 0

bags\_count = 0

while bags\_count < bags:

 bags\_weight = int(input('Сколько весит', bags\_count,'мешок: '))

 total\_weight += bags\_weight

 bags\_count += 1

print('Все мешки вешают: ', total\_weight)

Рассмотрим еще один из элементов управления циклами. Мы уже один знаем это оператор break – он позволяет прервать выполнение цикла. А есть еще один оператор, который позволяет продолжить выполнение цикла. Пока это не совсем понятно. Давайте рассмотрим пример: в Китае считается неудачным число 4 (потому что звучит также как слово «смерть»), и иногда в лифте вы можете не найти этаж с таким номером. Давайте сделаем программу, которая выведет номера этажей без числа 4:

floors = 20

floor\_count = 1

while floor\_count <= floors:

 if floor\_count != 4:

 print(floor\_count, 'этаж')

 floor\_count += 1

Такое решение имеет место быть, но можно сделать эту же программу еще одним способом, а именно с использованием команды продолжить цикл и пишется она – continue, а программа:

floors = 20

floor\_count = 1

while floor\_count <= floors:

 if floor\_count == 4:

 floor\_count += 1

 continue

 print(floor\_count, 'этаж')

 floor\_count += 1

Такая программа работает так: печатаются все этажи, но когда переменная floor\_count равно 4 выполнение цикла начинается с начала.