Урок №3. Знакомство с циклом while

Аннотация

План урока

1. Повторение.
2. Обращение по индексу.
3. Цикл `while`.
4. Краткие операторы присваивания.

Аннотация

Сначала рассматривается обращение по индексу к символам строки (включая отрицательные индексы). Затем — цикл `while`. Акцентируется внимание на том, что в одной и той же строчке программы на разных итерациях цикла переменные могут иметь разное значение. Вводятся краткие операторы типа `+=`.

Повторение

Задача «Вспомнить всё: if» (https://lms.yandexlyceum.ru/task/view/703).

Задача «password123» (https://lms.yandexlyceum.ru/task/view/704).

Необязательная задача «password123456» (https://lms.yandexlyceum.ru/task/view/719).

Немного новой информации о строках. Пока мы умеем делать строки четырьмя способами: задавать напрямую, считывать с клавиатуры функцией `input()`, преобразовывать число функцией `str` и склеивать из двух других строк операцией `+`.

```
fixed_word = 'опять'
print(fixed_word)
word = input()
print(word)
number = 25
number_string = str(number)
print(number_string)
word_plus_number = fixed_word + number_string
print(word_plus_number)
```

А если нам нужно взять только часть строки — например, одну букву? Для этого нужно после самой строки написать в квадратных скобках номер этой буквы (он же «**индекс**»). Нумерация при этом начинается с нуля: первая буква имеет номер 0, следующая — 1, и так далее. «Буква» в данном случае не какой-то новый тип данных, а обычная строка длиной 1.

```
word = 'привет'
initial_letter = word[0]
print(initial_letter)  # сделает то же, что print('п')
other_letter = word[3]
print(other_letter)  # сделает то же, что print('в')
```

Естественно, в этом примере `word` с тем же успехом можно было считать с клавиатуры через `input()`. Тогда мы не могли бы сразу сказать, чему равны переменные `initial_letter` и `other_letter`.

Поскольку нумерация начинается с нуля, номер последней буквы в строке на единицу меньше длины этой строки. Если попытаться получить букву, номер которой слишком велик, Питон выдаст ошибку.

Задача «Пятая буква» (https://lms.yandexlyceum.ru/task/view/705).

Задача «Скажите а» (https://lms.yandexlyceum.ru/task/view/706).

Необязательная задача «Скажите а (заглавное)» (https://lms.yandexlyceum.ru/task/view/720).

Конечно, номер в квадратных скобках – не обязательно фиксированное число, которое прописано в самой программе. Его тоже можно считать с клавиатуры или получить в результате арифметического действия.

```
word = 'привет'
number_of_letter = int(input())  # предположим, пользователь ввёл 3
print(word[number_of_letter])  # тогда это выведет 'в'
```

Задача «Какая-то там буква» (https://lms.yandexlyceum.ru/task/view/707).

Необязательная задача «Именно та буква» (https://lms.yandexlyceum.ru/task/view/721).

Задача «Последняя буква» (https://lms.yandexlyceum.ru/task/view/708).

Разрешены и отрицательные индексы: `word[-1]` означает последний символ строки `word`, `word[-2]` — предпоследний, и т. д.

Задача «Последняя буква 2» (https://lms.yandexlyceum.ru/task/view/709). +

Задача «Игра в города: один раунд» (https://lms.yandexlyceum.ru/task/view/710).

Необязательная задача «Игра в города: мягкий знак» (https://lms.yandexlyceum.ru/task/view/722).

Важная конструкция в Питоне — **оператор цикла** `while`. Он выполняет блок кода, **пока** истинно какое-то условие.

Напомним, условный оператор `if` проверяет условие и, в зависимости от того, истинно оно или ложно, выполняет либо не выполняет следующий записанный с отступом блок. После этого программа в любом случае выполняется дальше (там ещё может быть elif или else, но сути это не меняет). Оператор `while` («пока») тоже проверяет условие и тоже в случае его истинности выполняет следующий блок кода («**тело цикла**»). Однако после выполнения этого блока кода выполняется не то, что идёт после него, а снова проверяется условие, записанное после `while`. Ведь при выполнении тела цикла значения каких-то переменных могли измениться — в результате условие цикла может уже не быть истинным. Если условие всё ещё истинно, тело цикла выполняется снова. Как только условие цикла перестало выполняться (в том числе если оно с самого начала не было выполнено), программа идёт дальше — выполняются команды, записанные после тела цикла.

Один шаг цикла (выполнение тела цикла) ещё называют **итерацией**.

Используйте цикл **while** всегда, когда какая-то часть кода должна выполниться несколько раз — причём невозможно заранее сказать, сколько именно.

```
number = float(input())
while number > 0:
    print('Вы ввели положительное число! Вводите дальше.')
    number = float(input())
    print('Так-так, что тут у нас...')
print('Вы ввели отрицательное число или ноль. Всё.')
```

Сначала выполняется первая строчка: `number = float(input())` — пользователь вводит число.

(Мы предполагаем, что пользователь действительно ввёл число, и программа не вылетела с ошибкой.)

Предположим, он ввёл число 10. Оно записано в переменной `number`.

Выполняется вторая строчка: `while number > 0:` — «пока number > 0» — здесь проверяется, выполнено ли условие number > 0. Поскольку мы предположили, что `number` в этот момент равно 10, то да, условие выполнено, поэтому дальше выполняется блок, записанный с отступом — тело цикла.

Третья строчка программы выводит на экран строку, тут всё понятно.

Четвёртая строчка вновь считывает с клавиатуры число и сохраняет его в переменную number. Пусть пользователь ввёл 2.

Когда выполнение программы доходит до конца тела цикла, происходит возврат к заголовку цикла (второй строчке программы) и повторная проверка условия. Поскольку 2>0, снова выполняется тело цикла.

Третья строчка снова выводит на экран сообщение, четвёртая строчка снова считывает число (пусть это будет число 3), пятая строчка снова выводит на экран сообщение...

Закончив тело цикла, опять проверяем условие в заголовке. number равно 3, 3>0, поэтому продолжаем.

Третья строчка опять выводит на экран сообщение, четвёртая строчка опять считывает число. Пусть теперь это будет -1 — обратите внимание, переменная number на каждой итерации цикла приобретает новое значение! Пятая строчка опять выводит на экран сообщение...

Вновь вернувшись на вторую строчку, получаем, что -1>0 — ложно. Поэтому цикл завершается, тело цикла больше не выполняется, прыгаем сразу на следующую после цикла строчку программы — шестую. Она выводит последнее сообщение.

Всё.

Задачи:

«Продолжайте говорить А» (https://lms.yandexlyceum.ru/task/view/711) +

«Числа до ноля» (https://lms.yandexlyceum.ru/task/view/712) +

«Строки до пустой» (https://lms.yandexlyceum.ru/task/view/713). +

Необязательные задачи:

«Круглые» (https://lms.yandexlyceum.ru/task/view/723) +

«password» (https://lms.yandexlyceum.ru/task/view/724) +

Напомним, что в операторе присваивания одно и то же имя переменной может стоять и справа (в составе какого-то выражения), и слева. В этом случае сначала вычисляется правая часть со старым значением переменной, после чего результат становится новым значением этой переменной. Ни в коем случае не воспринимайте такой оператор присваивания как уравнение!

```
number = int(input())  # например, 5
number = number + 1  # тогда здесь number становится равным 6
print(number)
```

Это особенно важно для циклов. Рассмотрим такую задачу: пользователь вводит числа. Пусть это будут цены на товары, купленные пользователем, а наша программа — часть программного обеспечения кассового аппарата. Ввод "-1" — сигнал остановки. Нужно сосчитать сумму всех введённых чисел (сумму чека). Поскольку требуется повторить нечто (ввод очередной цены) неизвестное количество раз, потребуется цикл `while`. Нам понадобится как минимум две переменные: price для цены очередного товара и total для общей суммы.
Если бы мы знали точно, что пользователю надо купить ровно три товара, то цикл (и ввод -1 как условие его прерывания) был бы не нужен. Тогда программа могла бы выглядеть так:

```
total = 0
price = float(input())
total = total + price
price = float(input())
total = total + price
price = float(input())
total = total + price
print('Сумма введённых чисел равна', total)
```

Однако количество товаров неизвестно, поэтому понадобится цикл. С учётом сказанного выше программа будет выглядеть так:

```
total = 0
print('Вводите цены; для остановки введите -1.')
price = float(input())
while price > 0:
    total = total + price
    price = float(input())
print('Общая стоимость равна', total)
```

Обратите внимание: мы назвали переменные осмысленно. Это очень облегчит жизнь программисту, который будет читать наш код позже — даже если это будете вы сами неделю спустя. Однако интерпретатор Питона к этому факту совершенно равнодушен. Чтобы значения переменных соответствовали названиям и тому смыслу, который мы в них закладываем, нужно поддерживать переменные в актуальном состоянии. И только вы, программист, можете это сделать. С переменной price всё более или менее понятно: её значение обновляется при считывании с клавиатуры на каждой итерации цикла, как это делалось во многих других задачах. total сначала равно нулю: до начала ввода цен их сумма, конечно, ноль. Однако значение переменной total устаревает каждый раз, когда пользователь вводит цену очередного товара. Поэтому нам нужно прибавить к значению total только что введённую цену, чтобы эта переменная по-прежнему обозначала сумму цен всех купленных товаров.

Отметим, что для конструкций вида `total = total + price` есть краткая запись `total += price`. Аналогично оператор `x = x + y` можно записать как `x += y`, оператор `x = x * y` — как `x *= y`, и так для любого из семи арифметических действий.

Задача «Скидки» (https://lms.yandexlyceum.ru/task/view/714).

Задача «Игра в города» (https://lms.yandexlyceum.ru/task/view/715).

Необязательная задача «Лабиринт с правом на ошибку» (https://lms.yandexlyceum.ru/task/view/725) , продолжающая задачу «Лабиринт» (https://lms.yandexlyceum.ru/task/view/691) (если это задание не было сделано раньше, нужно его сделать).

Задача «Учитель». (https://lms.yandexlyceum.ru/task/view/716)

Необязательные задачи:

«Сиракузская последовательность» (https://lms.yandexlyceum.ru/task/view/726).

«Бурсацкое развлечение» (https://lms.yandexlyceum.ru/task/view/727).

Поподробнее почитать про цикл while можно здесь (http://pythontutor.ru/lessons/while/) (там же есть много другой информации о Питоне).

Также можно порешать задачи на сайте pythontutor.ru (http://pythontutor.ru/) (но там программа курса сильно отличается от нашей).

Домашнее задачи:

«Сколько строк?» (https://lms.yandexlyceum.ru/task/view/717)

«Среднее» (https://lms.yandexlyceum.ru/task/view/718)

«1024 и все-все-все» (необязательная) (https://lms.yandexlyceum.ru/task/view/728).