Алгоритм. Циклический алгоритм. While,for. Range. Random

Повторение

- Что такое алгоритм?
- Что такое условный алгоритм?
- Перечислите операторы сравнения
- Приведите пример условной конструкции python.
- Что такое блок? Можно своими словами

Циклический алгоритм. Цикл For.

- Если нужно пять раз напечатать слово «привет», вы можете сделать следующее:

```
>>> print("привет")
привет
>>> print("привет")
привет
>>> print("привет")
привет
>>> print("привет")
привет
>>> print("привет")
привет
```

- Однако это не красиво, да и не удобно, что если нам пришлось бы писать не 5 ра: а 100 … Для решения такого рода задач принято использовать циклы.

Циклический алгоритм

- **Циклический алгоритм** – описание действий, которые должны повторяться указанное число раз или пока не выполнено заданное условие. Перечень повторяющихся действий называют **телом цикла**.
- Циклические алгоритмы бывают двух типов:
  Циклы со счетчиком, в которых какие-то действия выполняются определенное число раз – цикл for ;
  Циклы с условием, в которых тело цикла выполняется, в зависимости от какого-либо условия – цикл while.

Циклический алгоритм

- Слегка изменим нашу предыдущую программу, и воспользуемся циклом for, т.к. нам известно что нужно вывести 5 раз. Для удобства воспользуемся функцией range(), которая возвращает нам последовательность чисел в указанном диапазон, которую мы можем использовать в качестве счетчика цикла.

```
>>> for x in range(0, 5):
        print('привет')
```

```
привет
привет
привет
привет
привет
```

Функция range()

- Дополнительные возможности:

- `range(end)`: создается набор чисел от 0 до числа end
- `range(start, end)`: создается набор чисел от числа start до числа end
- `range(start, end, step)`: создается набор чисел от числа start до числа end с шагом step

Цикл For.

- В цикле for мы поочередно берем каждое значение из этого списка и сохраняем его в переменную X чтобы могли воспользоваться ею внутри тела цикла. Цикл for работает с последовательностями, ему нужно проходиться, по чему-то, что-то перебирать, как листать странички, одна за другой.
- Более наглядно:

```
>>> for x in range(0, 5):
        print('привет %s' % x)
привет 0
привет 1
привет 2
привет 3
привет 4
```

Блоки

- Python ожидает, что у всех строк в блоке будут одинаковые отступы. И неважно, из скольких пробелов состоит отступ, главное, чтобы каждая новая строка блока начиналась с такого же отступа, как и предыдущая (такой код легче для человеческого восприятия).

```
>>> hugehairypants = ['огромные', 'волосатые', 'штаны']
>>> for i in hugehairypants:
        print(i)
        for j in hugehairypants:
            print(j)
```

Практика

- Как вы считаете, что делает эта программа?

```
for x in range(0, 20):
    print('привет %s' % x)
    if x < 9:
        break
```

- Программа выводит на экран квадраты всех целых чисел от 1 до 10.

```
for x in range(1, 11):
    print(x*x)
```

Цикл While.

- Цикл while также используется для повторения частей кода, но вместо зацикливания на n количество раз, он выполняет работу до тех пор, пока не достигнет определенного условия. Давайте взглянем на простой пример:

```
i = 0
while i < 10:
    print(i)
    i = i +1
```

Break

- Цикл while является своего рода условным оператором. Вот что значит этот код: пока переменная i меньше единицы, её нужно выводить на экран. Далее, в конце, мы увеличиваем её значение на единицу. Если вы запустите этот код, он выдаст от 0 до 9, каждая цифра будет в отдельной строке, после чего задача будет выполнена. Если вы удалите ту часть, в которой мы увеличиваем значение i, то мы получим бесконечный цикл. Как правило – это плохо. Бесконечные циклы известны как логические ошибки, и их нужно избегать. Существует другой способ вырваться из цикла, для этого нужно использовать встроенную функцию break. Давайте посмотрим, как это работает.

```
i = 0
while i < 10:
    print(i)
    if i == 5:
        break
    i += 1
```

Continue

- Существует еще один, под названием continue, который в основном используется для пропуска итерации, или перейти к следующей итерации. Вот один из способов его применения:

```
i = 0
while i < 10:
    if i == 3:
        i += 1
        continue
    print(i)
    if i == 5:
        break
    i += 1
```

Итого

- Итак, цикл while выполняет следующие действия:

1. Проверяет условие.
2. Выполняет код в блоке.
3. Повторяет все сначала.

- Основное отличие цикла while от for в том, чтобы он не зациклился, необходимо как-то изменять наше условие. В цикле for же у нас заранее известно количество итераций.

Random

- Модуль random позволяет генерировать случайные числа. Прежде чем использовать модуль, необходимо подключить его с помощью инстр
```
import random
```
- Наиболее популярные функции:
  - random.random() - возвращает псевдослучайное число от 0.0 до 1.0
  - random.randint(<Начало>, <Конец>) - возвращает псевдослучайное целое число в диапазоне от <Начало> до <Конец>.
  - random.randrange(<Начало>, <Конец>, <Шаг>) - возвращает случайно выбранное число из последовательности.

Практика

- Программа выводит на экран квадраты всех целых чисел от 1 до 10. При помощи цикла while

```
i = 1
while i <=10:
    print(i*i)
    i+=1
```

- Посчитать четные и нечетные цифры числа. Нужно определить сколько в числе четных цифр, а сколько нечетных. Число вводится с кл

```
a = input()
a = int(a)
even = 0
odd = 0

while a > 0:
    if a%2==0:
        even += 1
    else:
        odd += 1
    a = a//10
print("Even: %d, odd: %d" %(even, odd))
```

ДЗ

- Бесконечный калькулятор. Программа, которая выполняет над двумя числами одну из четырех операций (сложение, вычитание, умножение, деление). Программа должна завершаться только по желанию пользователя.
- Посчитать сумму и произведение цифр числа. При помощи цикла while
- Посчитать сумму и произведение цифр числа. При помощи цикла for, не преобразовывая изначально введенную строку в число.