Самое важное

Вложенные циклы
Важно понять, в какой последовательности Python выполняет код.

Если мы укажем:
```
for i in range(10):
    <тело цикла> — тело цикла повторится 10 раз.
```

При этом Python’у на самом деле не так важно, что будет телом цикла.

Это может быть одна операция print(i):
```
for i in range(10):
    print(i)
```

Но это может быть и вложенный цикл:
```
for i in range(10):
    for i in range(10):
        print(i)
```

Для лучшего понимания последовательности выполнения кода используйте дополнительные принты!

```
for i in range(2):
    print(‘start’)
    for j in range(5):
        print(i, j)
    print(‘end’)
```

Такие принты позволят явно увидеть, в каком порядке выполняются операции.

Пример

Итерация первого внешнего цикла

start
пары чисел — тут начинается вложенный цикл (его первый запуск)
0 0
0 1
0 2
0 3
0 4
end — вложенный цикл закончился, и после него выполнился print(‘end’)

Итерация второго внешнего цикла

start
пары чисел — тут начинается вложенный цикл (его первый запуск)
1 0 (обратите внимание, что первое число теперь равно 1, так как это вторая итерация внешнего цикла)
1 1
1 2
1 3
1 4
end — вложенный цикл закончился, и после него выполнился print(‘end’)

Таким образом, нам видно, что вложенный цикл отработал полностью два раза (по одному разу на каждую итерацию внешнего цикла).

Условные блоки во вложенных циклах
Выше указано, что вложенный цикл полностью отрабатывает на одной итерации внешнего цикла. Это так же значит, что всё это время, пока работает вложенный цикл, переменная внешнего цикла не изменяется.

Это позволяет выполнять во вложенном цикле разные действия в зависимости от переменной внешнего цикла.

```
for i in range(4):
    print(‘start’)
    for j in range(2):
        if i < 2:
            print(‘hello’)
        else:
            print(‘bye’)
    print(‘end’)
```

На первых двух итерациях вложенный цикл будет два раза повторять hello, на последующих — по два раза повторять bye.

start
hello
hello
end
start
hello
hello
end
start
bye
bye
end

start
bye
bye
end

Изменение количества итераций во вложенном цикле
Работая со вложенными циклами, программист получает доступ к нескольким переменным цикла. Важно при этом понимать, чему равна каждая из них на протяжении всего времени работы циклов.

Особенно это важно, если нужно связать переменные циклов между собой. Например, можно влиять на количество итераций вложенного цикла в зависимости от переменной цикла внешнего цикла.

```
for i in range(3):
  print('start')
  for j in range(i):
     print('hello')
  print('end')
```

Мы увидим:

start — внешняя переменная цикла равна 0, значит, вложенный цикл просто не будет запущен (0 итераций)
end
start — тут внешняя переменная уже равна 1, значит, одна итерация вложенного цикла будет выполнена
hello
end
start
hello — теперь переменная внешнего цикла равна 2, и у нас появилось две надписи
hello
hello
end

Оператор else для цикла FOR
Оператор else может использоваться не только для условного блока if/else. Он также может быть использован для циклов, в этом случае код внутри блока else сработает, если цикл был завершен самостоятельно, то есть без применения break.

Синтаксис

```
for <переменная цикла> in <последовательность>:
  <тело цикла>
else:
  <код>
```

Обращайте внимание на отступы — именно они позволят Python понять, что else вы применяете к циклу for, так как они находятся на одном уровне отступа.

Не допускай следующих ошибок!

Не забывайте, что вложенный цикл — это просто блок кода, который выполняется, как и весь остальной код, внутри цикла. Помните о том, что код выполняется сверху вниз (строчка за строчкой).

Это значит, что на каждую итерацию внешнего цикла будет полностью выполнен код внутри, в том числе полностью отработает вложенный цикл от начала до конца!