Язык программирования Python

08 Конструкция ветвления

Последовательность выполнения

До сих пор все написанные нами программы выполнялись последовательно, оператор за оператором.

```
print('one') # Оператор 1
print('two') ') # Оператор 2
print('three') ') # Оператор 3
```

Выполнение этой программы выдаст нам следующий результат:

```
one
two
three
```

Сначала выполняется первый оператор, затем второй и так далее.

Конструкция ветвления

Программа, состоящая только из последовательно идущих операторов, годится для очень простых программ. С повышением сложности программ какие-то фрагменты последовательности приходится разветвлять, какие-то повторять несколько раз подряд и еще много чего.

Ветвление — это первая конструкция, с которой мы познакомимся, позволяющая влиять на последовательность выполнения.

Условие

При выполнении ветвления сначала проверяется условие.

Условие может быть истинным или ложным.

Ветвление может быть полным и неполным.

При полном ветвлении конструкция содержит операторы на выполнение, как для истинного условия, так и для ложного.

При неполном ветвлении есть только оператор для истинного условия.

Оператор if

В языке Python ветвление задается с помощью оператора if (если).

```
a = int(input())
if a == 1:
    print(a, 'равно 1')
```

В данной программе происходит следующее:

Мы вводим значение для переменной a (предположим, что ввели ‘1’). Дальше в заголовке оператора if, после ключевого слова if находится условие, представляющее собой логическое выражение. Результатом вычисления логического выражения является не число, а логическое значение (истина или ложь). a равно 1? Да, верно. Тогда выполняется тело условия, стоящее после двоеточия и отдаленное от заголовка оператора if на 4 пробела.

Логический тип данных

Логический тип данных (bool) — это тип данных, множество значений которого состоит лишь из 2 значений: True (истина) и False (ложь).

Если в условии стоит выражение, значение которого не типа bool, то оно приводится к нему. Например, если в условии будет стоять целое число, то любое значение, кроме 0, преобразуется в True.

0 — это False, пустая строка — тоже преобразуется в False, пустой список — False.

Логические операции: not (логическое НЕ), and (логическое И), or (логическое ИЛИ).

Логические операции часто используются для объединения результатов условных выражений.

Логические выражения

Логические выражения — это выражения, возвращающие логическое значение.
Логические выражения содержат логические операции и операции сравнения.
Операции сравнения:

==	равенство	True, если оба операнда равны, иначе False
!=	неравенство	True, если оба операнда не равны, иначе False
>	больше	True, если первый операнд больше второго, иначе False
>=	больше или равно	True, если первый операнд больше или равен второму, иначе False
<	меньше	True, если первый операнд меньше второго, иначе False
<=	меньше или равно	True, если первый операнд меньше или равен второму, иначе False

Полное ветвление

```
a = int(input())
if a == 1:
    print(a, ‘равно 1’)
else:
    print(a, ‘не равно 1’)
```

При а, не равном 1, будет выполняться блок операторов, стоящий после ключевого слова else (иначе). После слова else также стоит двоеточие.

Множественное ветвление

Множественное ветвление задается посредством if-elif-else.

```
a = int(input())
if a == 1:
    print(a, ‘равно 1’)
elif a > 1:
    print(a, ‘больше 1’)
else:
    print(a, ‘меньше 1’)
```

Здесь добавляется ключевое слово elif (else if — иначе если).

Сравнение списков

```
numbers1 = [1, 2, 3, 4, 5, 6, 7, 8, 9]
numbers2 = list(range(1,10))
if numbers1 == numbers2:
    print("numbers1 equal to numbers2")
else:
    print("numbers1 is not equal to numbers2")
```

Тернарная условная операция

- Тернарная условная операция во многих языках представлена следующим образом:

```
result = condition ? a : b
```

result равно a, если значение condition истинно, иначе result равно b.

- В Python это дело выглядит немного иначе:

```
result = a if condition else b
```

- Еще способ:

```
result = (b, a)[condition]
```

- Аналог с помощью if-else выглядит так:

```
if condition:
    result = a
else:
    result = b
```

Итоги

Мы познакомились с такими понятиями, как последовательность выполнения, конструкция ветвления, условие, неполное ветвление, полное ветвление, множественное ветвление, логический тип данных, логические операции, условные операции, оператор if, ключевое слово else, ключевое слово elif, тернарная условная операция,

а также попрактиковались в работе с интерпретатором в файловом режиме.