Операторы ветвления в Python

условное и альтернативное выполнение, последовательность условий, вложенные условия

Что такое операторы ветвления

Операторы ветвления (условные операторы) – операторы, конструкции языка программирования, которые обеспечивают выполнение определенной команды (набора команд) только при условии истинности некоторого логического выражения, либо выполнение одной из нескольких команд (наборов команд) в зависимости от некоторого выражения

```
1   age = 25
2
3   if age < 14:
4       print('ребенок')
5   elif age < 18:
6       print('подросток')
7   elif age < 60:
8       print('взрослый')
9   else:
10      print('пожилой')
```

В чем истина, брат?

Что, если элемент, который мы проверяем, не является булевым? Что Python считает за True, а что за False?

Значение False не обязательно явно означает False. Например, к False приравниваются все следующие значения:

- Булева переменная False
- Значение None
- Целое число 0
- Число с плавающей точкой 0.0
- Пустая строка (‘’)
- Пустой список []
- Пустой кортеж ()
- Пустой словарь {}
- Пустое множество set()

Все остальные значение приравниваются к True

Вопрос-ответ

? Как правильно сравнить выражение с True или False?

✔ Для проверки на истинность достаточно написать:

- if some_condition:
- if not some_condition:
- Не сравнивайте логические типы с помощью «==» (PEP8)
- Совсем плохо использовать оператор «is» (PEP8)

```python
val = True

# правильно
if val:
    print('ok')

# правильно
if not val:
    print('not ok')

# неправильно
if val == True:
    print('oh no')


# совсем неправильно
if val is True:
    print('oh no')
```

Вопрос-ответ

? Как правильно сравнивать выражение с None?

✔ Для сравнения с None используются операторы is и is not. Так и только так (PEP8). При другом варианте сравнения можно словить баг, так как в некоторых классах может быть переопределен метод __bool__

```
>>> None is None
True
>>> True is None
False
>>> False is None
False
```

Тернарный оператор

Если необходимо выполнить простое короткое условие, то его можно записать в одну строку с помощью тернарного оператора. Таким образом можно в одну строку присвоить значение переменной или вернуть значение из функции по условию.

Не надо писать в тернарный оператор сложные условия, ибо по дзену «Читаемость имеет значение»

```
1  def some_func(some_val):
2      another_val = "it's true" if some_val else "it's false"
3      return '+' if another_val == "it's true" else '-'
4
5
6  print(some_func(True))
```

В Python нет switch

Java

```java
int season = 3;

String seasonString;
switch(season){
  case 1:
    seasonString = "зима";
    break;
  case 2:
    seasonString = "весна";
    break;
  case 3:
    seasonString = "лето";
    break;
  case 4:
    seasonString = "осень"
    break;
  default:
    seasonString = "значение не найдено";
    break;
}
```

Python

```python
season = 3

season_string = {
    1: 'зима',
    2: 'весна',
    3: 'лето',
    4: 'осень'
}.get(season, 'Значение не найдено')
```

Вложенные условия

Условия могут быть вложены друг в друга и будут выполняться последовательно

```python
some_val = 123

if isinstance(some_val, int):
    if some_val >= 0:
        result = 'positive'
    else:
        result = 'negative'
elif isinstance(some_val, str):
    if some_val.isdigit():
        result = 'digit'
    else:
        result = 'not digit'
```

Ваши вопросы

что необходимо прояснить в рамках данного раздела

Основы булевой алгебры

как работают условия, которые мы пишем

Булева алгебра

Булева алгебра – математический аппарат, с помощью которого записывают, вычисляют упрощают и преобразовывают логические выражения

- Создатель – математик Джордж Буль (в честь него названа)
- Оперирует двумя понятиями: высказывание истинно или высказывание ложно

Операции

- Логическое сложение (Дизъюнкция)
- Логическое умножение (Конъюнкция)
- Логическое отрицание (Инверсия)

Логическое сложение (дизъюнкция, «или»)

Результат дизъюнкции истинен тогда, когда истинно хотя бы одно из входящих в него простых высказываний

A	B	F=A∨B
0	0	0
0	1	1
1	0	1
1	1	1

Логическое умножение (конъюнкция, «и»)

Результат конъюнкции истинен тогда, когда истинны оба из входящих в него простых высказываний

- Результат конъюнкции определяется по таблице истинности - это таблица, устанавливающая соответствие между всеми возможными наборами логических переменных, входящих в логическую функцию, и значениями функции

A	B	F=A^B
0	0	0
0	1	0
1	0	0
1	1	1

Логическое отрицание (инверсия, «не»)

Инверсия – операция, которая делает истинное высказывание ложным, а ложное истинным

A	F=¬A
0	1
1	0

Основные законы

- Закон тождества. Всякое высказывание тождественно самому себе. A=A
- Закон непротиворечия. A∧(¬A)=0, A∨(¬A)=1
- Инволютивность отрицания ¬¬A=A

Основные законы

- Закон коммуникативности. В булевой алгебре можно менять местами логические переменные при операциях логического сложения и умножения. A∧B=B∧A, A∨B=B∨A
- Закон ассоциативности. Когда результат вычисления не зависит от порядка вычисления. A∧(B∧C)=(A∧B)∧C, A∨(B∨C)=(A∨B)∨C
- Закон дистрибутивности.
  - Конъюнкция относительно дизъюнкции : A∨(B∧C) =(A∨B)∧(A∨C)
  - Дизъюнкция относительно конъюнкции: A∧(B∨C) =(A∧B)∨(A∧C)

Полезные ссылки

- Булева алгебра: https://math.wikia.org/ru/wiki/Булева_алгебра