

# Условные операторы и тип Bool

Умение работать с условными операторами пригодится разработчику, когда нужно создать программу, в которой в зависимости от условий ход решения должен варьироваться, и у разработчика при этом должна быть возможность контролировать ход выполнения этой программы. **Оператор if**

Условный оператор **if** (в пер. с англ. «если») задаёт определённое поведение программы в зависимости от заданных разработчиком условий: «если условие истинно — выполни код, следующий за условием».

Синтаксис оператора:

1 Объявление ветвления: ключевое слово **if** и условие.

2 После условия — двоеточие.

3 Тело условного оператора отбивается четырьмя пробелами от начала строки с объявлением ветвления.

```
rating = 4.9

# Объявление ветвления и условие.
if rating > 4.7: # Обязательное двоеточие.
    # Тело условного оператора
    print('Фильм крут')

# Этот код не вложен в блок if,
# поэтому выполнится в любом случае
print('Проверка окончена')
```

## Конструкция if...else

Позволяет обработать ситуацию, когда результат вычисления условия в **if** ложный. Для этого используется ключевое слово **else** (в пер. с англ. «иначе»). Код, записанный после этого ключевого слова, сработает в том случае, если условие в **if** ложно.

```
rating = 4.9

if rating > 4.7:
    print('Фильм крут')
else:
    # Код выполнится, если rating <= 4.7
    # Отступы важны!
    print('Так себе киношечка')

print('Проверка окончена')
```

Короткий вариант конструкции if...else:

```
result = 'Фильм крут' if rating > 4.7 else 'Так себе киношечка'
print(result)

print('Проверка окончена')
```

## Конструкция if...else

Когда условий больше, чем одно, логику программы можно описать так:

- если **if** ...
- если **elif** ..., то ...
- иначе **else** ... Особенности:
- после ключевого слова **elif** записывается условие;
- после условия ставится двоеточие, точно как в **if**;
- код в теле блока сработает, если условие в **elif** выполнится;
- код в теле **elif** отбивается четырьмя отступами.

```
rating = 3.0

if rating > 4.7:
    print('Фильм крут')
elif rating > 3.5:
    print('Смотреть можно')
else:
    print('Так себе киношечка')
print('Проверка окончена')
```

В конструкцию **if...elif...else** может быть включено сколько угодно блоков **elif**.



Важен порядок, в котором установлены условия. Программа читает код сверху вниз, и как только одно из условий сработает — программа выполнит код в теле этого условия, а в следующие блоки (**elif** и **else**) не будет заглядывать.

## Операторы сравнения

Знак в Python	Описание
<=	меньше или равно
>=	больше или равно
<	меньше
>	больше
==	равно
!=	не равно

## Логический тип bool

Выражения с операторами сравнения возвращают значение типа **bool**. Это логический тип данных, у которого два значения: **True** (истина) и **False** (ложь)

Тип **bool** — это подкласс типа **int**, и значения **True** и **False** можно конвертировать в целые числа:

- **int(True)** приведёт значение к числу 1
- **int(False)** приведёт значение к числу 0

Любое число, не равное 0 или непустой объект конвертируются в **True**. Число 0 пустые объекты и значение **None** конвертируются в **False**. Сравнение с **None** выполняется по ключевому слову **is**:

```
rating = None

if rating is None:
    print('У фильма нет оценки')
else:
    print('Оценка поставлена')
```

## Логические операторы

### Оператор AND

Логический оператор **and** возвращает **True**, если оба логических выражения вернут **True**; если хоть одно из выражений вернёт **False** — то оператор **and** вернёт **False**.

```
rating = 4.9

if rating and rating >= 4.7:
    # При rating = 4.9 это условие вернёт True
    print('Отличный фильм, бегом за билетами!')
else:
    # Если оценки нет или она меньше 4.7
    print('У фильма нет оценки')

print('Проверка окончена')
```

### Оператор OR

Логический оператор **or** (в пер. с англ. «или»): возвращает **True**, если хотя бы одно из логических выражений возвращает **True**.

### Оператор NOT

Оператор **not** инвертирует булевы значения: **not True** вернёт **False**; **not False** вернёт **True**.