Python: Логические операторы

- И `and`
- ИЛИ `or`

Мы уже умеем писать функции, которые проверяют одиночные условия. А в этом уроке научимся строить составные условия.

Предположим, что сайт при регистрации требует, чтобы пароль был длиннее восьми символов и короче двадцати. В математике бы написали `8 < x < 20`, но во многих языках программирования так сделать нельзя. Попробуем написать два отдельных логических выражения и соединим их специальным оператором «И»:

> Пароль длиннее 8 символов **И** пароль короче 20 символов

Вот функция, которая принимает пароль и говорит, соответствует ли он условиям (`True`) или не соответствует (`False`):

```
def is_correct_password(password):
    length = len(password)
    return length > 8 and length < 20
```

```
print(is_correct_password('qwerty'))                    # =>
print(is_correct_password('qwerty1234'))                # =>
print(is_correct_password('zxcvbnmasdfghjkqwertyui'))   # =>
```

`and` — означает «И». В математической логике это называют конъюнкцией. Все выражение считается истинным, если истинен каждый **операнд** — каждое из составных выражений. Иными словами, `and` означает «и то, и другое». Приоритет этого оператора ниже, чем приоритет операторов сравнения. Поэтому выражение `length > 8 and length < 20` правильно отрабатывает без скобок.

Кроме `and` часто используется оператор `or` — «ИЛИ» (дизъюнкция). Он означает «или то, или другое, или оба». Выражение `a or b` считается истинным, если хотя бы один из операндов или одновременно все — истинные. В другом случае выражение ложное.

Операторы можно комбинировать в любом количестве и любой последовательности. Если в коде одновременно встречаются `and` и `or`, то приоритет задают скобками. Ниже пример расширенной функции, которая определяет корректность пароля:

```
def has_special_chars(str):
    # Проверяет содержание специальных символов в строке


def is_strong_password(password):
```

```
    length = len(password)
    # Скобки задают приоритет. Понятно, что к чему относитс
    return (length > 8 and length < 20) and has_special_cha
```

Теперь представим, что мы хотим купить квартиру, которая удовлетворяет таким условиям: площадь от 100 квадратных метров и больше на любой улице **ИЛИ** площадь от 80 квадратных метров и больше, но на центральной улице `Main Street`.

Напишем функцию, которая проверит квартиру. Она принимает два аргумента: площадь — число и название улицы — строку:

```
def is_good_apartment(area, street):
    return area >= 100 or (area >= 80 and street == 'Main S

print(is_good_apartment(91, 'Queens Street'))  # => False
print(is_good_apartment(78, 'Queens Street'))  # => False
print(is_good_apartment(70, 'Main Street'))    # => False

print(is_good_apartment(120, 'Queens Street'))  # => True
print(is_good_apartment(120, 'Main Street'))    # => True
print(is_good_apartment(80, 'Main Street'))     # => True
```

https://replit.com/@hexlet/python-basics-logic-logical-operators

Область математики, в которой изучаются логические операторы, называется булевой алгеброй. Ниже увидите **таблицы истинности** — по ним можно

определить, каким будет результат, если применить оператора:

И `and`

A	B	A and B
True	True	**True**
True	False	False
False	True	False
False	False	False

ИЛИ `or`

A	B	A or B
True	True	**True**
True	False	**True**
False	True	**True**
False	False	False

Задание

Реализуйте метод `is_leap_year()`, который определяет является ли год високосным или нет. Год будет високосным, если он кратен (то есть делится без остатка) 400 или он одновременно кратен 4 и не кратен 100. Как видите, в определении уже заложена вся необходимая логика, осталось только переложить её на код:

```
is_leap_year(2018) # false
is_leap_year(2017) # false
is_leap_year(2016) # true
```

Кратность можно проверять так:

```
# % - возвращает остаток от деления левого операнда на прав
# Проверяем что number кратен 10
number % 10 == 0

# Проверяем что number не кратен 10
number % 10 != 0
```

▶ Упражнение не проходит проверку — что делать? 😶

▶ В моей среде код работает, а здесь нет 😐

▶ Мой код отличается от решения учителя 🤔

▶ Прочитал урок — ничего не понятно 🙄

Полезное

- Булева алгебра
- Логическое «И»
- Логическое «ИЛИ»

Определения

- Логические операторы — операторы «И» (`and`), ИЛИ (`or`), позволяющие создавать составные

логические условия.