

```
>>>
```

Результатом применения логического оператора `not` (НЕ) произойдет отрицание операнда, т.е. если операнд истинный, то `not` вернет – ложь, если ложный, то – истину.

Логический оператор `and` (И) вернет `True` (истину) или `False` (ложь)²⁴, если его операндами являются логические высказывания.

```
>>> 2 > 4 and 45 > 3 # комбинация False and True вернет False
False
>>>
```

Если операндами оператора `and` являются объекты, то в результате Python вернет объект:

```
>>> '' and 2 # False and True
''
>>>
```

Для вычисления оператора `and` Python вычисляет операнды слева направо и возвращает первый объект, имеющий ложное значение.

Посмотрим на столбец `and` таблицы истинности. Какая закономерность? Если среди операндов (X, Y) есть ложный, то получим ложное значение, но вместо ложного значения для операндов-объектов Python возвращает первый ложный операнд, встретившийся в выражении, и дальше вычисления НЕ производит. Это называется вычислением по короткой схеме.

```
>>> 0 and 3 # вернет первый ложный объект-операнд
0
>>> 5 and 4 # вернет крайний правый объект-операнд
4
>>>
```

Если Python не удастся найти ложный объект-операнд, то он возвращает крайний правый операнд.

Логический оператор `or` действует похожим образом, но для объектов-операндов Python возвращает первый объект, имеющий истинное значение. Python прекратит дальнейшие вычисления, как только будет найден первый объект, имеющий истинное значение.

```
>>> 2 or 3 # вернет первый истинный объект-операнд
2
>>> None or 5 # вернет второй объект-операнд, т.к. первый всегда ложный
5
>>> None or 0 # вернет оставшийся объект-операнд
0
```

X Y	and	or
0 0	0	0
0 1	0	1
1 0	0	1
1 1	1	1

Таким образом, конечный результат становится известен еще до вычисления остальной части выражения.

²⁴ Исходя из таблицы истинности