AND Gate

2 min

Our next gate is the *AND gate*. This gate receives two inputs and only returns current if the inputs are both *on*.

As in previous exercises, you'll only be able to use gates you've previously built: NAND_gate(), and NOT_gate().

Here's the truth table:

a	***b***	**output**
0	0	0
0	1	0
1	0	0
1	1	1

Instructions

1. Checkpoint 1 Passed

1.

Define AND_gate() which has two parameters a and b.

AND_gate() should return 1 if both a and b are 1. It should return 0 otherwise.

Hint

The first gate we built, NAND_gate() is short for NOT AND gate.

```
NAND_gate(1, 1)
# 0
NAND_gate(0, 1)
# 1

# meanwhile...

AND_gate(1, 1)
# 1
AND_gate(0, 1)
# 0
```

to Clipboard

In other words, AND is the **negated** version of NAND. We can "negate" an output with our previous gate:

```
NOT_gate(NAND_gate(1, 1)) == AND_gate(1, 1)
# True
```

script.py

```
from nand import NAND_gate
from not_gate import NOT_gate


def AND_gate(a, b):
  if a and b:
    return 1
  else:
    return 0



# TEST CASES


print("A: 0, B: 0 | Output: {0}".format(AND_gate(0, 0)))
print("A: 0, B: 1 | Output: {0}".format(AND_gate(0, 1)))
print("A: 1, B: 0 | Output: {0}".format(AND_gate(1, 0)))
print("A: 1, B: 1 | Output: {0}".format(AND_gate(1, 1)))
```