NOT Gate

3 min

Now that we've completed our NAND_gate(), we can use that function to build other gates.

This time you're starting from scratch, so lean on that truth table. Fortunately, our next gate, *NOT* only has one input.

Here's the truth table:

a	**output**
0	1
1	0

Instructions

1. Checkpoint 1 Passed

1.

Define NOT_gate() which takes one input a and returns the outputs specified in the truth table.

You can do this using if/else statements, like with the last gate. But, if you want to challenge yourself, you can **use NAND_gate() inside your NOT_gate()**.

Hint

NOT_gate() returns 1 if the input is 0 and 0 if the input is 1.

While you *can* do this without NAND_gate(), push yourself to use that function!

Remember how NAND_gate() operates:

```
# our NOT_gate() will receive one input...
input_1 = 1
# or
input_0 = 0

NAND_gate(input_1, input_1)
# 0
NAND_gate(input_0, input_0)
# 1
```

script.py

```
from nand import NAND_gate


def NOT_gate(a):
  if a:
    return 0
  else:
    return 1
```

```
# TEST CASE

print("A: 0 | Output: {0}".format(NOT_gate(0)))

print("A: 1 | Output: {0}".format(NOT_gate(1)))
```