Do Now

bigd103.link/student-loan-calculator

Conditionals

Making Decisions in Code

Why Do Programs Need to Make Decisions?

Think about real life decisions:

- "If the bill is over $50, I'll leave a tip"
- "If it's raining, I'll take an umbrella"
- "If I have enough money, I'll buy coffee"

Programs need to make similar decisions based on conditions!

The `if` Statement

The simplest way to make a decision in code:

```python
bill = 75.50
if bill > 50:
    bill *= 1.2  # Add 20% tip
print(f"Total: ${bill}")
```

```
Total: $90.6
```

Key parts:

- `if` keyword
- A condition that’s True or False
- Colon `:`
- Indented code that runs if True

How `if` Works

```python
temperature = 85
if temperature > 80:
    print("It's hot outside!")
    print("Turn on the AC!")

print("This always prints")
```

```
It's hot outside!
Turn on the AC!
This always prints
```

- Python checks: Is `temperature > 80`?
- If True → runs the indented code
- If False → skips the indented code
- Non-indented code always runs

Comparison Operators

Tools for creating conditions:

Operator	Meaning	Example	Result
`==`	Equal to	`5 == 5`	`True`
`!=`	Not equal to	`5 != 3`	`True`
`>`	Greater than	`5 > 3`	`True`
`<`	Less than	`5 < 3`	`False`
`>=`	Greater than or equal	`5 >= 5`	`True`
`<=`	Less than or equal	`5 <= 3`	`False`

True/False Values

`True` and `False` are values just like numbers and strings. We can either evaluate an expression directly in the `if`-statement or use a variable:

```python
if 5 > 3:
    print("5 is greater than 3")

five_is_greater = 5 > 3
if five_is_greater:
    print("5 is greater than 3")
```

```
5 is greater than 3
5 is greater than 3
```

Common Mistake: = vs ==

```python
# Assignment (giving a value)
age = 18

# Comparison (checking equality)
if age == 18:
    print("You just became an adult!")
```

Remember:

- One `=` assigns a value
- Two `==` compares values

The `else` Statement

What if the condition is False?

```python
balance = 25.00
pizza_cost = 30.00

if balance >= pizza_cost:
    print("You can afford the pizza!")
else:
    print("Not enough money for pizza :(")
    print(f"You need ${pizza_cost - balance:.2f} more")
```

```
Not enough money for pizza :(
You need $5.00 more
```

if / else Structure

Else is used to handle the case when the `if` condition is False:

```
if condition:
    # This runs if condition is True
    code_block_1
else:
    # This runs if condition is False
    code_block_2
```

Only ONE block runs - never both!

```
temperature = 70
if temperature > 75:
    print("It's hot!")
else:
    print("It's not hot")
```

```
It's not hot
```

The `elif` Statement

What about multiple options?

```python
grade = 85

if grade >= 90:
    print("A - Excellent!")
elif grade >= 80:
    print("B - Good job!")
elif grade >= 70:
    print("C - Satisfactory")
else:
    print("Needs improvement")
```

```
B - Good job!
```

Python checks top to bottom and stops at the first True condition

Order Matters with `elif`

Each statement is checked in order:

```
score = 95
if score >= 70:
    print("C")
elif score >= 90:
    print("A")
```

In order to get an "A", the highest conditions must be checked first:

```
# RIGHT - Check highest first
if score >= 90:
    print("A")
elif score >= 70:
    print("C")
```

```
PythonError: Traceback (most recent call last):
File "/lib/python312.zip/_pyodide/_base.py", line 597, in eval_code_async
await CodeRunner(
File "/lib/python312.zip/_pyodide/_base.py", line 411, in run_async
```

Combining Conditions with `and`

Both conditions must be True:

```python
age = 16
has_permit = True

if age >= 16 and has_permit:
    print("You can practice driving!")
else:
    print("Not ready to drive yet")
```

```
You can practice driving!
```

This is useful when you need multiple criteria to be met before taking action.

We can also use `and` with variables:

```python
age = 16
has_permit = True

can_drive = age >= 16 and has_permit

if can_drive:
    print("You can practice driving!")
else:
    print("Not ready to drive yet")
```

```
You can practice driving!
```

Sometimes it’s nice to break complex conditions into variables for clarity.

Combining Conditions with `or`

At least one condition must be True:

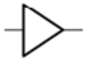
```python
day = "Saturday"
is_holiday = False

if day == "Saturday" or day == "Sunday" or is_holiday:
    print("No school today!")
else:
    print("Time for school")
```

Truth Tables

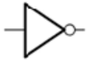
A	B	A and B	A or B
True	True	True	True
True	False	False	True
False	True	False	True
False	False	False	False

YES



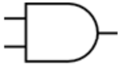
INPUT	OUTPUT
A	
0	0
1	1

NOT



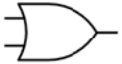
INPUT	OUTPUT
A	
0	1
1	0

AND



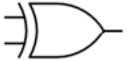
INPUT		OUTPUT
A	B	
0	0	0
1	0	0
0	1	0
1	1	1

OR



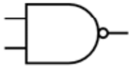
INPUT		OUTPUT
A	B	
0	0	0
1	0	1
0	1	1
1	1	1

XOR



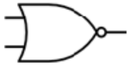
INPUT		OUTPUT
A	B	
0	0	0
1	0	1
0	1	1
1	1	0

NAND



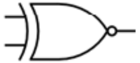
INPUT		OUTPUT
A	B	
0	0	1
1	0	1
0	1	1
1	1	0

NOR



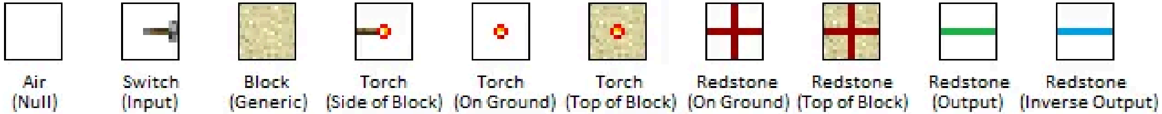
INPUT		OUTPUT
A	B	
0	0	1
1	0	0
0	1	0
1	1	0

XNOR

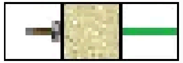


INPUT		OUTPUT
A	B	
0	0	1
1	0	0
0	1	0
1	1	1

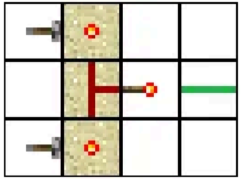
MineCraft Logic Gates



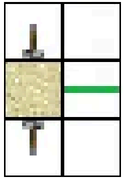
Input/Output Gate



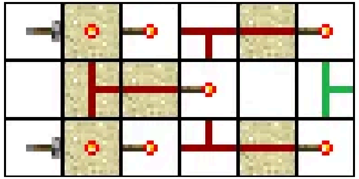
AND Gate



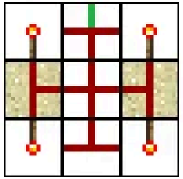
OR Gate



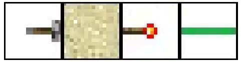
XOR Gate



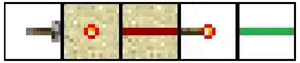
Rapid Pulser



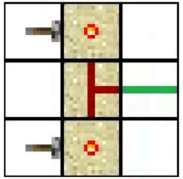
NOT Gate (Inverter)



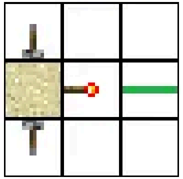
Double NOT (Repeater)



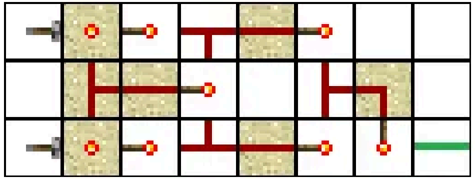
NAND Gate



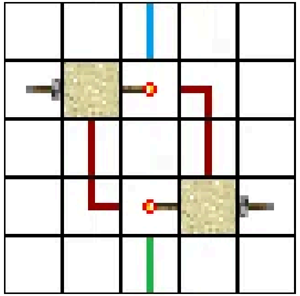
NOR Gate



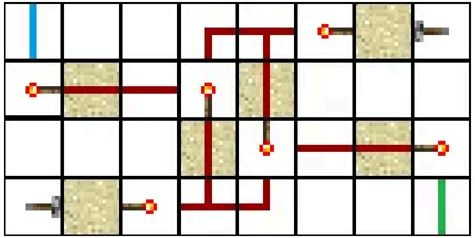
XNOR Gate



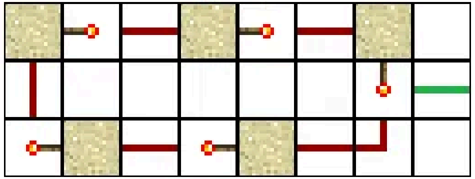
RS NOR Latch (Memory Cell)



RS NAND Latch (Memory Cell)



5-Clock (Pulser)



Nested Conditionals

We can put `if` statements inside `if` statements:

```python
has_money = True
amount = 50

if has_money:
    if amount >= 100:
        print("You're rich!")
    elif amount >= 20:
        print("You have some spending money")
    else:
        print("You're almost broke")
else:
    print("You have no money")
```

This is often cleaner to many `and` statements:

```python
if has_money and amount >= 100:
    print("You're rich!")
elif has_money and amount >= 20:
    print("You have some spending money")
elif has_money and amount < 20:
    print("You're almost broke")
else:
    print("You have no money")
```

Real Example: Smart Tip Calculator

```python
bill = 75.50
service = "excellent"

if service == "excellent":
    tip_percent = 25
elif service == "good":
    tip_percent = 20
elif service == "okay":
    tip_percent = 15
else:
    tip_percent = 10

tip = bill * (tip_percent / 100)
print(f"Tip: ${tip:.2f}")
```

```
Tip: $18.88
```

Exercise: Grade Calculator

bigd103.link/grade-calculator