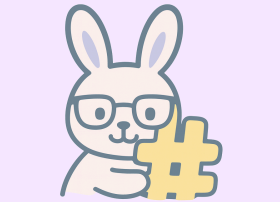
Python For-Loop with Range



ALGORITHM
WITH CHELSEA

Why do we need a loop?

Sometimes we want to do the **same thing many times**.

Examples:

- Print "Hello!" 5 times.
- Count from 1 to 10.
- Add up numbers from 5 to 10.

A **loop** helps us repeat code without copying and pasting.

What is a `for` loop?

A `for` loop repeats code for each value in a sequence.

Example:

```
for i in range(5):
    print(i)
```

- `i` is the **loop variable**.
- `i` changes each time the loop runs.
- The code inside must be **indented** (4 spaces).
- This prints numbers from 0 to 4.

`range(stop)`

- `range()` creates a sequence of numbers.
- The simplest form is: `range(stop)`.
- `range(5)` means `0, 1, 2, 3, 4`.
- Important: Count from `0`, **not** `1`.
- Important: `stop` is **not included**.

Q1 🖨️ `range(stop)`

What is the output?

```
for i in range(3):
    print(i)
```

Q1 🖨️ `range(stop)`

What is the output?

```
for i in range(3):
    print(i)
```

Output:

```
0
1
2
```

- Count from 0, **not** 1.
- `stop` (which is 3) is **not included**.

`range(start, stop)`

- We can choose where to start by using `range(start, stop)`.
- `range(2, 6)` means `2, 3, 4, 5`.
- Important: `start` (which is 2) is **included**.
- Important: `stop` (which is 6) is **not included**.

Q2 📋 `range(start, stop)`

What is the output?

```
for i in range(5, 8):
    print(i)
```

Q2 📋 `range(start, stop)`

What is the output?

```
for i in range(5, 8):
    print(i)
```

Output:

```
5
6
7
```

- `start` (which is 5) is included.
- `stop` (which is 8) is **not** included.

`range(start, stop, step)`

We can also choose how much to jump each time.

Example: Count by 2.

```
for i in range(0, 10, 2):
    print(i)
```

`range(start, stop, step)`

We can also choose how much to jump each time.

Example: Count by 2.

```
for i in range(0, 10, 2):
    print(i)
```

Output:

```
0
2
4
6
8
```

0 is included; 10 is **not** included.

Q3 ⏳ `range(start, stop, step)`

What is the output?

```
for i in range(1, 13, 3):
    print(i)
```

Q3 ⏳ `range(start, stop, step)`

What is the output?

```
for i in range(1, 13, 3):
    print(i)
```

Output:

```
1
4
7
10
```

1 is included; 13 is **not** included.

Count Down with a Negative Step

`step` can also be **negative**.

Example: countdown

```
for i in range(5, 0, -1):
    print(i)
```

- We start at 5.
- We stop before 0 (not included).
- We step by -1.
- This prints: 5, 4, 3, 2, 1 (each on its own line).

Common Mistakes with `range`

- **Stop is not included**
  - `range(1, 5)` gives 1 to 4.
  - If you want 1 to 5, use `range(1, 6)`.
- **Wrong step direction**
  - `range(0, 10, -1)` includes nothing.
  - You cannot go up with a negative step.
- **Step cannot be 0**
  - `range(1, 10, 0)` is an error.

Accumulator

Example:

```
total = 0
total = total + 5

print(total)
```

- First, compute 0 + 5 = 5 (the right side).
- Then, assign the resulting 5 to `total`.

Accumulator

Example:

```
total = 0
total = total + 5

print(total)
```

Output:

```
5
```

Q4 🔢 Accumulator

What is the output?

```
total = 3

total = total + 6  # total becomes 9
total = total + 7  # total becomes 16

print(total)
```

Q4 🔢 Accumulator

What is the output?

```
total = 3

total = total + 6  # total becomes 9
total = total + 7  # total becomes 16

print(total)
```

Output:

```
16
```

Sum of a Range

To compute a sum, we usually use an **accumulator**.

Example: sum from 1 to 5

```
total = 0

for i in range(1, 6):
    total = total + i

print(total)
```

- 1 + 2 + 3 + 4 + 5 = 15.
- The output is 15.

Sum of Even Numbers

Compute 2 + 4 + 6 + … + 20.

Sum of Even Numbers

Compute 2 + 4 + 6 + … + 20.

How?

- Start from 2
- Step is 2
- Stop is 21 (20 is included, but 21 is not included)

Sum of Even Numbers

Compute 2 + 4 + 6 + ... + 20.

How?

- Start from 2
- Step is 2
- Stop is 21 (20 is included, but 21 is not included)

Python code:

```
total = 0

for i in range(2, 21, 2):
    total = total + i

print(total)
```

Q5 ✍️ Print Odd Numbers

Make this code print: 5, 7, 9, 11.

```
for x in range(____, ____, ____):
    print(x)
```

Q5 ✍️ Print Odd Numbers

Make this code print: 5, 7, 9, 11.

```
for x in range(____, ____, ____):
    print(x)
```

Hints:

- Start at 5.
- Jump by 2 each time.
- Stop should be the first number that is **not included**.

Q5 ✍️ Print Odd Numbers

Make this code print: 5, 7, 9, 11.

```
for x in range(5, 12, 2):
    print(x)
```

Hints:

- Start at 5.
- Jump by 2 each time.
- Stop should be the first number that is **not included**.

Q6 🧮 Compute a Sum

Fill in the blanks to compute the sum from 1 to 100.

```
total = 0

for i in range(____, ____):
    total = total + i

print(total)
```

Hint: to include 100, the stop must be 101.

Q6 🧮 Compute a Sum

Fill in the blanks to compute the sum from 1 to 100.

```
total = 0

for i in range(1, 101):
    total = total + i

print(total)
```

Hint: to include 100, the stop must be 101.

Summary

- A `for` loop repeats code for each value.
- `range(stop)` gives 0 up to stop-1.
- `range(start, stop)` includes start but not stop.
- `range(start, stop, step)` can jump by step (step can be negative).
- To compute a sum, use an accumulator like `total = total + i`.