# For loops with range.

## Why is this important?

Sometimes you won't want to just loop over a list, but you'll want to loop over a range of numbers. This is where the `range` function comes in.

## What are we going to do?

We're going to create a short program that will calculate the squares of a range of numbers.

## Steps

### 1. Let's create a new file called `calculate_squares.py`

### 2. In this file let's create a for loop that loops over a range of numbers.

- Let's start with a range of 5 numbers.
- Let's take a look what this code looks like

```
print("Calculating squares for the following range")

for number in range(5):
    print(f"iteration: {number}")
```

- you can see in the above code that we're not looping over a list of items but we're looping over a `range`
    - when using `range(5)` you see that a range is a sequence of numbers that starts at `0` and goes up to but not including the number that you pass in which here is `5`
- if you take a look at the output

```
$ python calculate_squares.py
Calculating squares for the following range
iteration: 0
iteration: 1
iteration: 2
iteration: 3
iteration: 4
```

- you can see that we're looping over the number 0 to 4.

### 3. Let's calculate the squares on each iteration.

- Let's change the code so that it calculates the square of each number in the range.

```python
print("Calculating squares for the following range")

for number in range(5):
    print(f"iteration: {number} has a square of {number ** 2}")
```

- so you can see here that you can use the `**` operator to calculate the square of a number.
- Let's take a look at the output.

```
$ python calculate_squares.py
Calculating squares for the following range
iteration: 0 has a square of 0
iteration: 1 has a square of 1
iteration: 2 has a square of 4
iteration: 3 has a square of 9
iteration: 4 has a square of 16
```

- You can see here that our calculate squares file is working as expected.

### 4. Let's change the range to start at a different point rather than 0.

- Let's say that we wanted to change the squares of each number from 2-6. We can do that by changing `range(2,6)` to have two values.
    - Note these are called arguments, and we're going to talk about them more shortly, but essentially what they do is that they change the behavior of the function that you're calling.

```python
print("Calculating squares for the following range")

for number in range(2,6):
    print(f"iteration: {number} has a square of {number ** 2}")
```

- Let's take a look at the output

```
$ python calculate_squares.py
Calculating squares for the following range
iteration: 2 has a square of 4
iteration: 3 has a square of 9
iteration: 4 has a square of 16
iteration: 5 has a square of 25
```

- you can see that we're now starting at 2 and going up to but not including 6.

### Let's increment by 2 instead of 1 and see what happens.

- Right now we're calling `range(2,6)` which is the same as calling `range(2,6,1)` because the third argument is the step.
  - the `1` is the default value for the step, and is optional if you're going to keep the default value.
  - if you want to change the step you can pass in a different value we're going to change this to `2` and see what happens
- Let's take a look at the code.

```
print("Calculating squares for the following range")

for number in range(2,6,2): # added the step
    print(f"iteration: {number} has a square of {number ** 2}")
```

- Let's take a look at the output

```
$ python calculate_squares.py
Calculating squares for the following range
iteration: 2 has a square of 4
iteration: 4 has a square of 16
```

- Why are we only seeing two numbers?
  - Well, we're starting at 2 and going up to but not including 6, but we're incrementing by 2 each time.
  - So we're starting at 2, then we're adding 2 to 2 which is 4, and then we're adding 2 to 4 which is 6, but we're not including 6 so we're only seeing 2 and 4.
- Note: if you're ever curious about what's in the range you can always convert it to a list and print it out.

```
# For example
list(range(2,6,2))
```

## Conclusion

You can see here how we can loop through some numbers using the `range` function. This is useful when you want to loop over a range of numbers rather than a list of items.