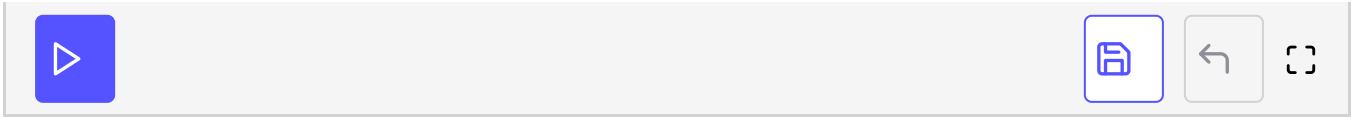
The while Loop

Another type of loop in python is while loop. Let's learn how a while loop works in python

The while loop is also used to repeat sections of code, but instead of looping n number of times, it will only loop until a specific condition is met. Let's look at a very simple example:

```
i = 0
while i < 10:
    print(i)
    i = i + 1
```



The while loop is kind of like a conditional statement. Here's what this code means: while the variable **i** is less than ten, print it out. Then at the end, we increase i's value by one. If you run this code, it should print out 0-9, each on its own line and then stop. If you remove the piece where we increment i's value, then you'll end up with an infinite loop. This is usually a bad thing. Infinite loops are to be avoided and are known as logic errors.

There is another way to break out of a loop. It is by using the **break** builtin. Let's see how that works:

```
i = 0
while i < 10:
    print(i)
    if i == 5:
        break
    i += 1

# 0
# 1
# 2
# 3
# 4
# 5
```

In this piece of code, we add a conditional to check if the variable **i** ever equals 5. If it does, then we break out of the loop. As you can see from the sample output, as soon as it reaches 5, the code stops even though we told the while loop to keep looping until it reached 10. You will also note that we changed how we increment the value by using +=. This is a handy shortcut that you can also use with other math operations, like subtraction (-=) and multiplication (*=).

The **break** builtin is known as a **flow control tool**. There is another one called **continue** that is used to basically skip an iteration or continue with the next iteration. Here's one way to use it:

```
i = 0

while i < 10:
    if i == 3:
        i += 1
        continue

    print(i)

    if i == 5:
        break
    i += 1
```

This is a little confusing, no? Basically we added a second conditional that checks if **i** equals 3. If it does, we increment the variable and continue with the next loop, which effectively skips printing the value 3 to the screen. As before, when we reach a value of 5, we break out of the loop.

There's one more topic we need to cover regarding loops and that's the **else** statement.