# Python

# Control Flow 2

in programming

## Loops

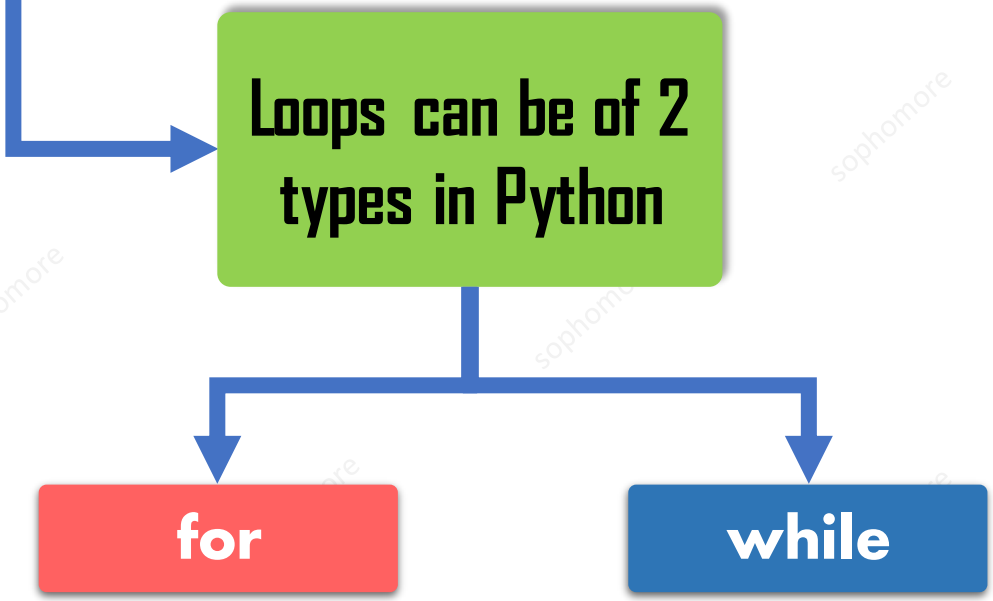
In python or in any other programming language as they help you to execute a block of code repeatedly.

**Why**

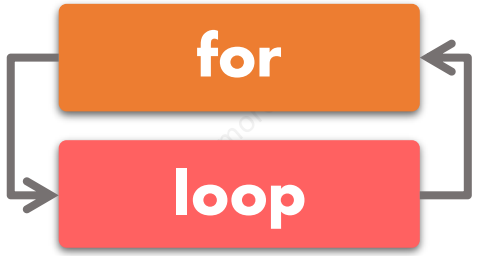
**To write less code, concise code for repeated process**

**When**

- **Run a code block repeatedly**
- **Process item in a sequence with same code**

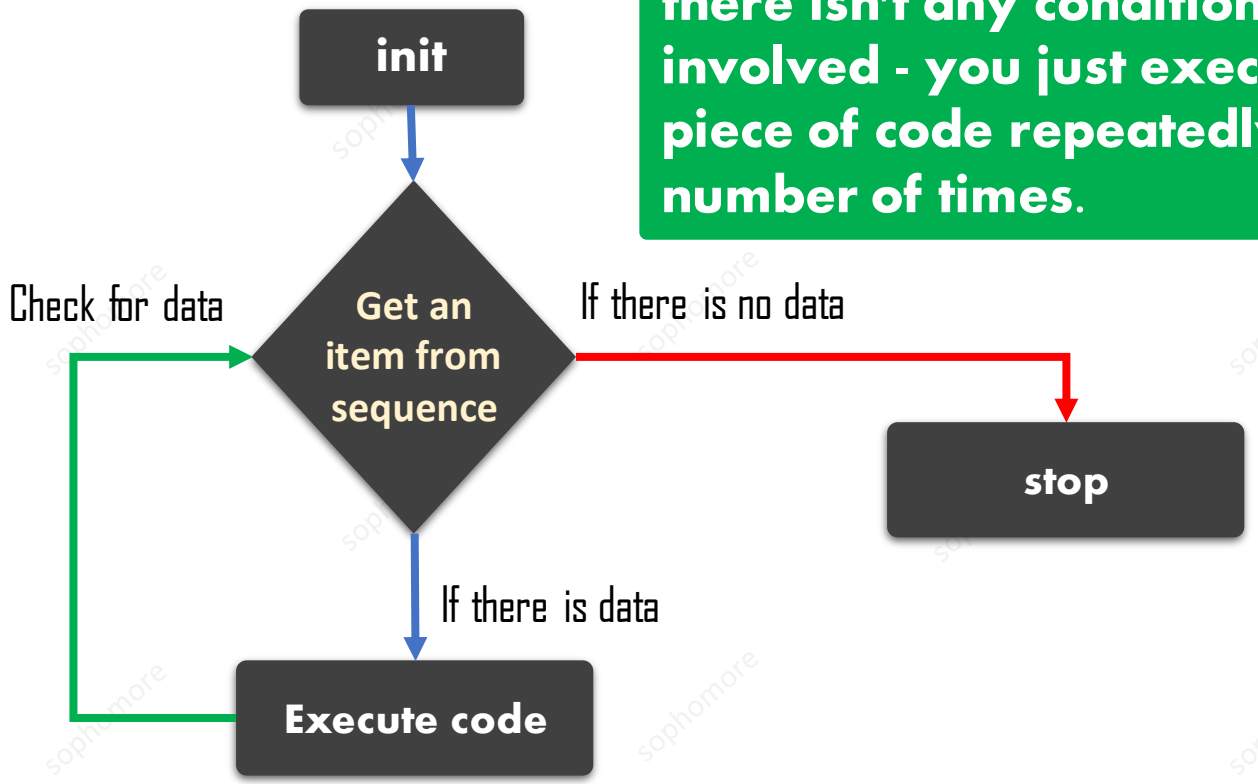


# For loop concept




the "for" keyword in "for loop" refers to something that you do for a certain number of times.

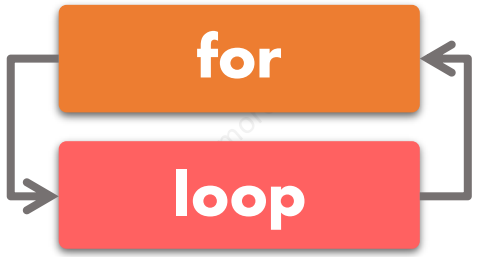
It executes a piece of code over and over again "for" a certain number of times, based on a sequence.

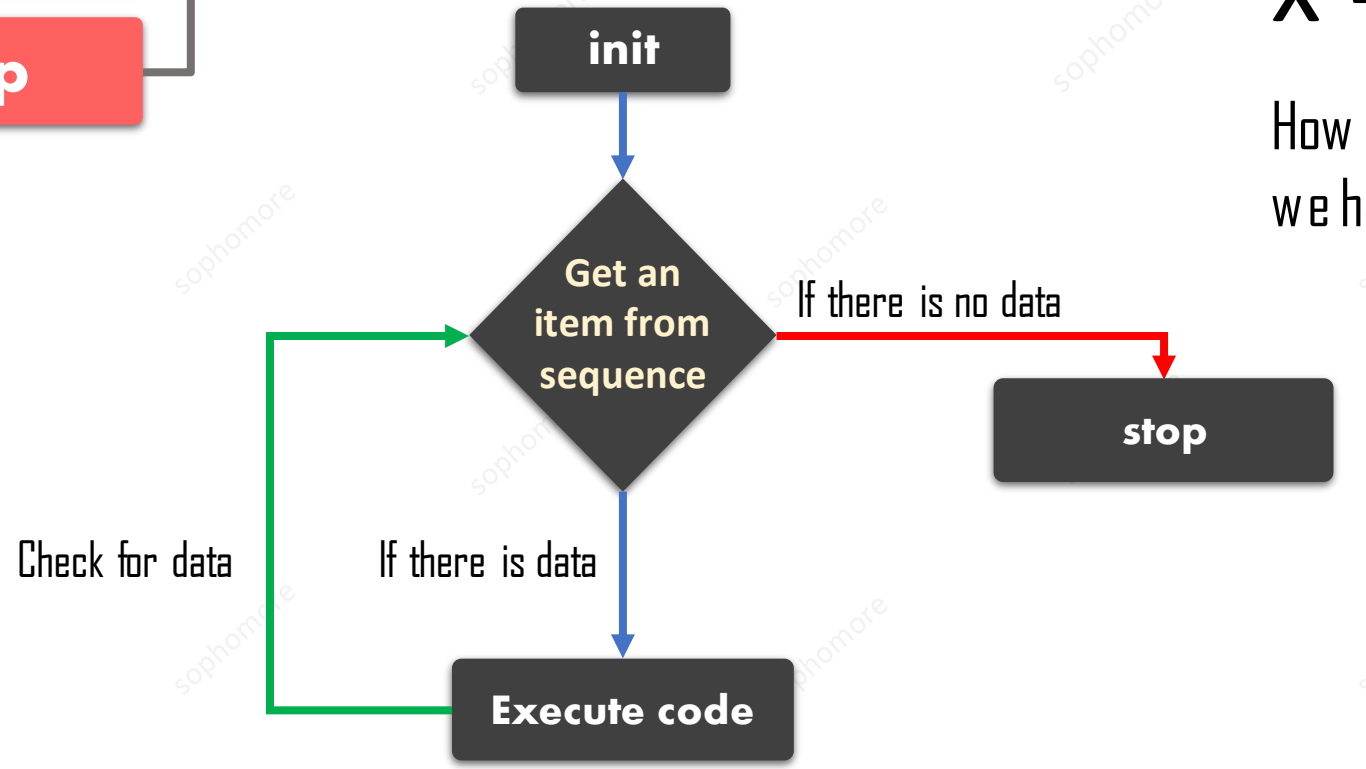



there isn't any condition actively involved - you just execute a piece of code repeatedly for a number of times.

Fact

# For loop



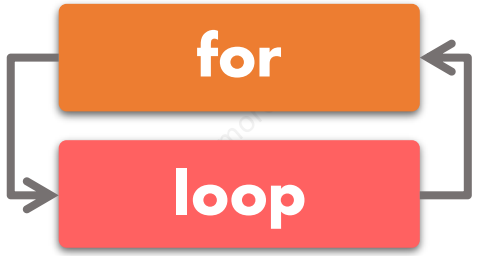

x = P Y T H O N

How many characters we have ?

**This is a sequence**

Because x contain multiple items so it can be taken as sequence.

# For loop




x = P Y T H O N

How many characters we have ?

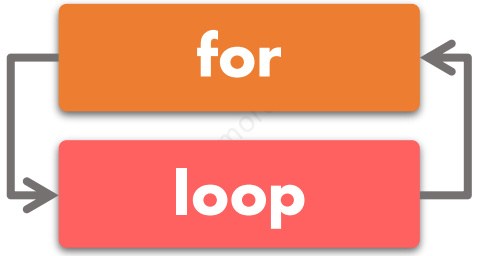
**This is a sequence**

Because x contain multiple items so it can be taken as sequence.

## for loop syntax

```
syntax

for <variable> in <sequence>:
    statement 1
    statement 2
    ...
    statement n
```

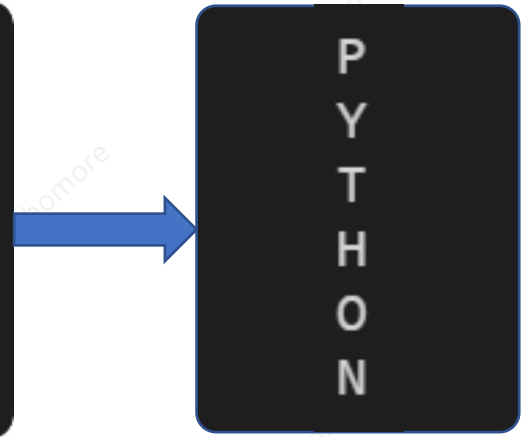
# For loop



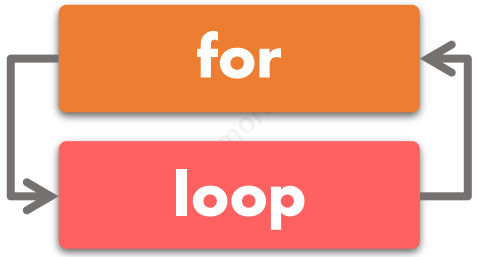

for loop syntax

```
syntax

for <variable> in <sequence>:
    statement 1
    statement 2
    ...
    statement n
```

```
x = "PYTHON"
for i in x:
    print(i)
```



# For loop




**for loop syntax**

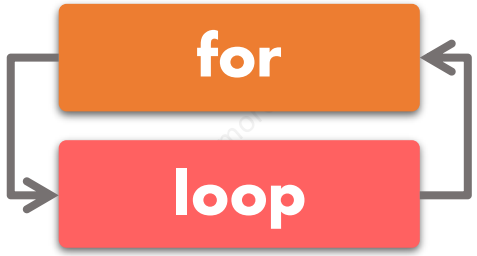
```
syntax

for <variable> in <sequence>:
    statement 1
    statement 2
    ...
    statement n
```

```
fruits = ['apple','banana','lemon']
for item in fruits:
    print(f'i have {item}')
```

output

```
i have apple
i have banana
i have lemon
```

# range() function




```
range()
```

We can use the python's range() function to specify how many time a loop will run

Depending on how many arguments we is passing to the function, we can decide where that series of numbers will begin and end as well as he gap between one number and the next.

```
range(10)
```

The range will be 10 item , from 0 to 9. 10 is the stopping value

The range will be from 1 to 19, 20 is the stopping value

```
range(1,20)
```

```
range(2,20,2)
```

The range will be from 2 to 18, and the gap between each number is 2, and 20 is stopping value

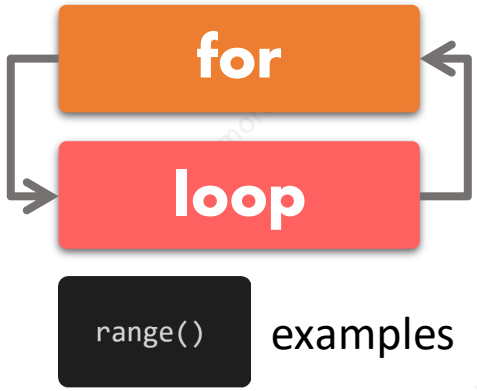
range() takes mainly three arguments

(optional) start: integer, which tells us the start value of the loop

stop: integer, that marks the end value of the range, for which the loop will run. It should be always n+1, where n is stop value

(optional) step: the gap between to 2 consecutive numbers in a range

# range() function




examples

```
for i in range(6):
    print('you got point',i)
```

```
you got point 0
you got point 1
you got point 2
you got point 3
you got point 4
you got point 5
```

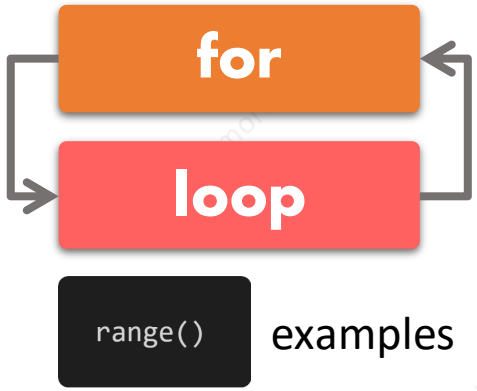
```
for num in range(2,11):
    print('sheep',num)
```

```
sheep 2
sheep 3
sheep 4
sheep 5
sheep 6
sheep 7
sheep 8
sheep 9
sheep 10
```

```
for num in range(1,10,3):
    print('level',num)
```

```
level 1
level 4
level 7
```

# range() function




examples

```
# reverse loop
for num in range(10,5,-1):
    print(num)
```

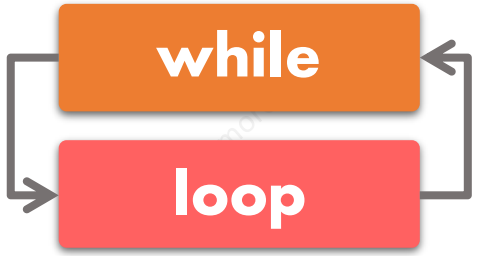
```
10
9
8
7
6
```

```
# pattern loop
for i in range(10):
    print(i * 'Na')
```

```
Na
NaNa
NaNaNa
NaNaNaNa
NaNaNaNaNa
NaNaNaNaNaNa
NaNaNaNaNaNaNa
NaNaNaNaNaNaNaNa
NaNaNaNaNaNaNaNaNa
```

# Le Code time

# While loop



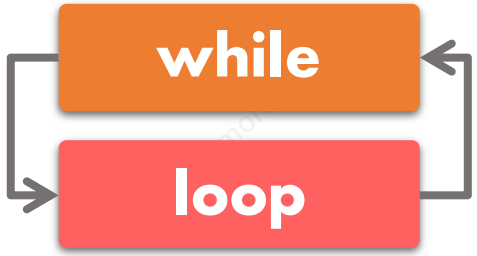

The while loop in Python is used to iterate over a block of code as long as the test expression (condition) is True.

We generally use this loop when we don't know the number of times to iterate beforehand.

**Syntax of while Loop in Python**

```
while expression:
    statement 1
    statement 2
    ...
    statement n
```

# While loop




```
x = 1
while x < 5:
    print('run')
    x += 1
print('stop')
```

```
run
run
run
run
stop
```

```
# a complex program
n = 10

# initialize sum and counter
sum = 0
i = 1

while i <= n:
    sum = sum + i
    i = i+1    # update counter

# print the sum
print("The sum is", sum)
```

```
The sum is 55
```

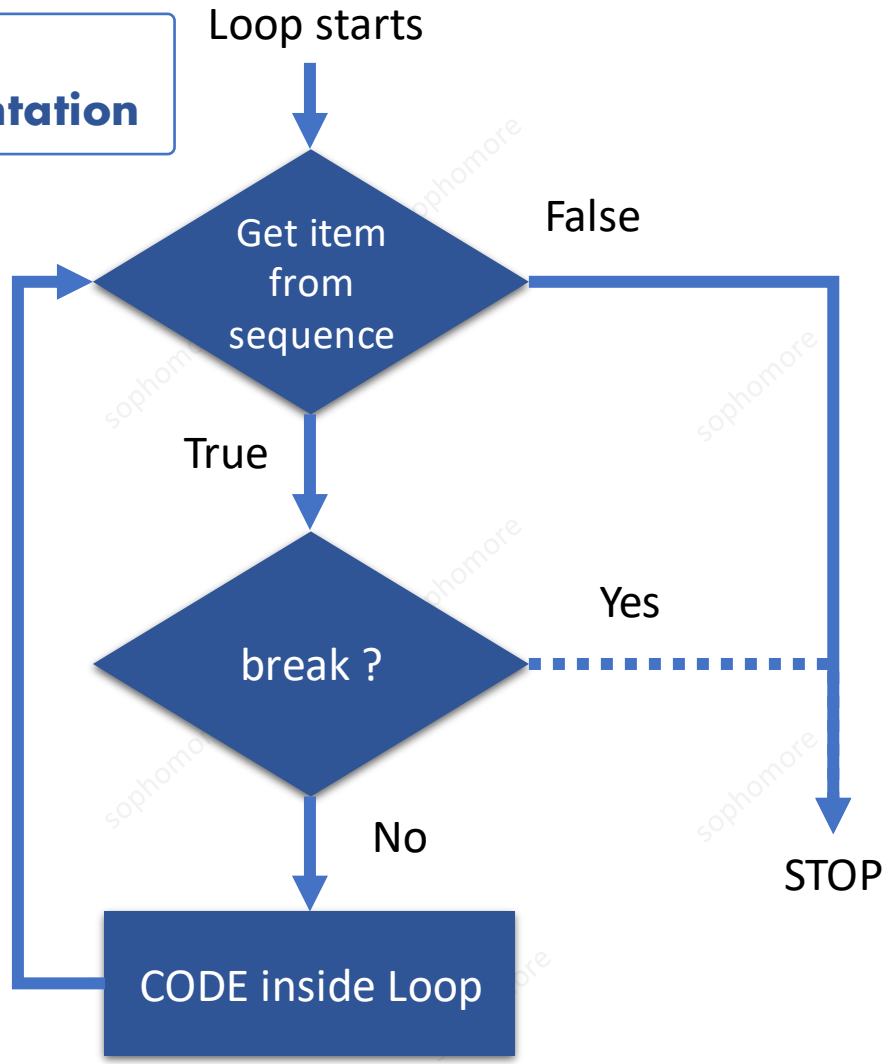
# Break and Continue

**In Python, break and continue statements can alter the flow of a normal loop.**

Loops iterate over a block of code until the test expression is false, but sometimes we wish to terminate the current iteration or even the whole loop without checking test expression.

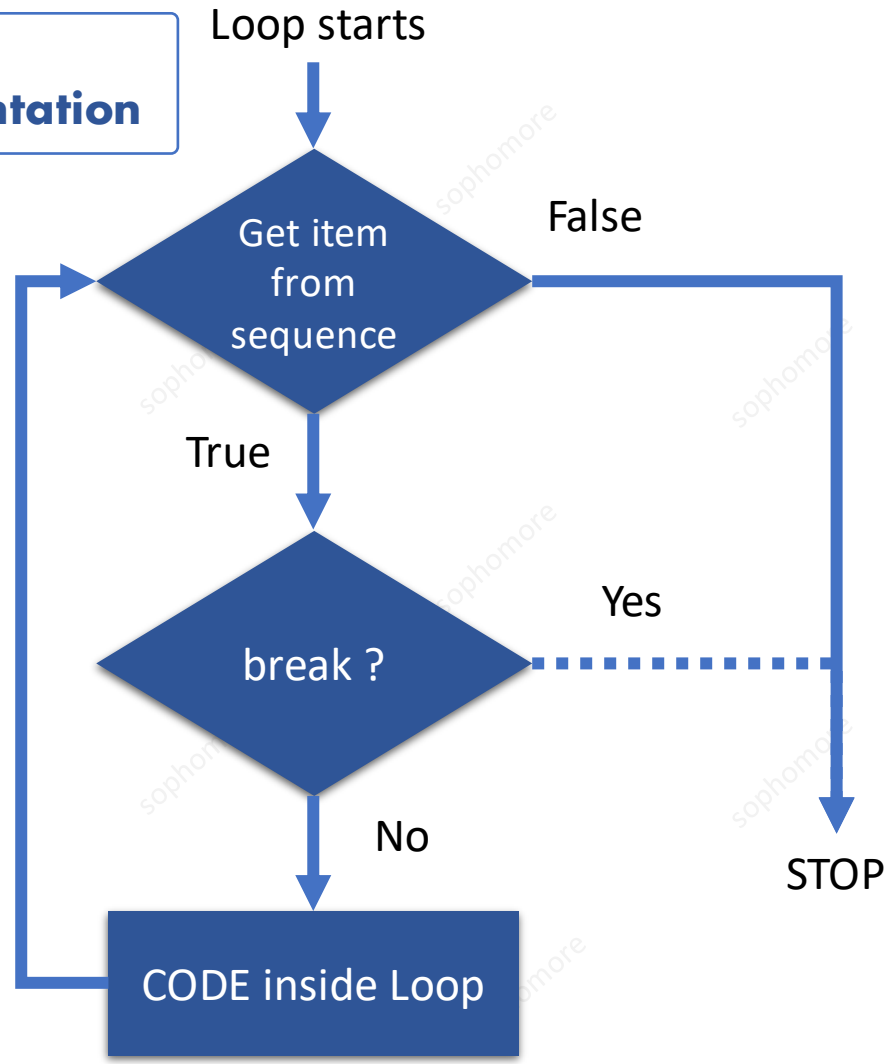
**The break and continue statements are used in these cases**

**Break representation**



# Break and Continue

**Break representation**




```
for var in sequence:
    # codes inside for loop
    if  condition:
        break
    # codes inside for loop

# codes outside for loop
```

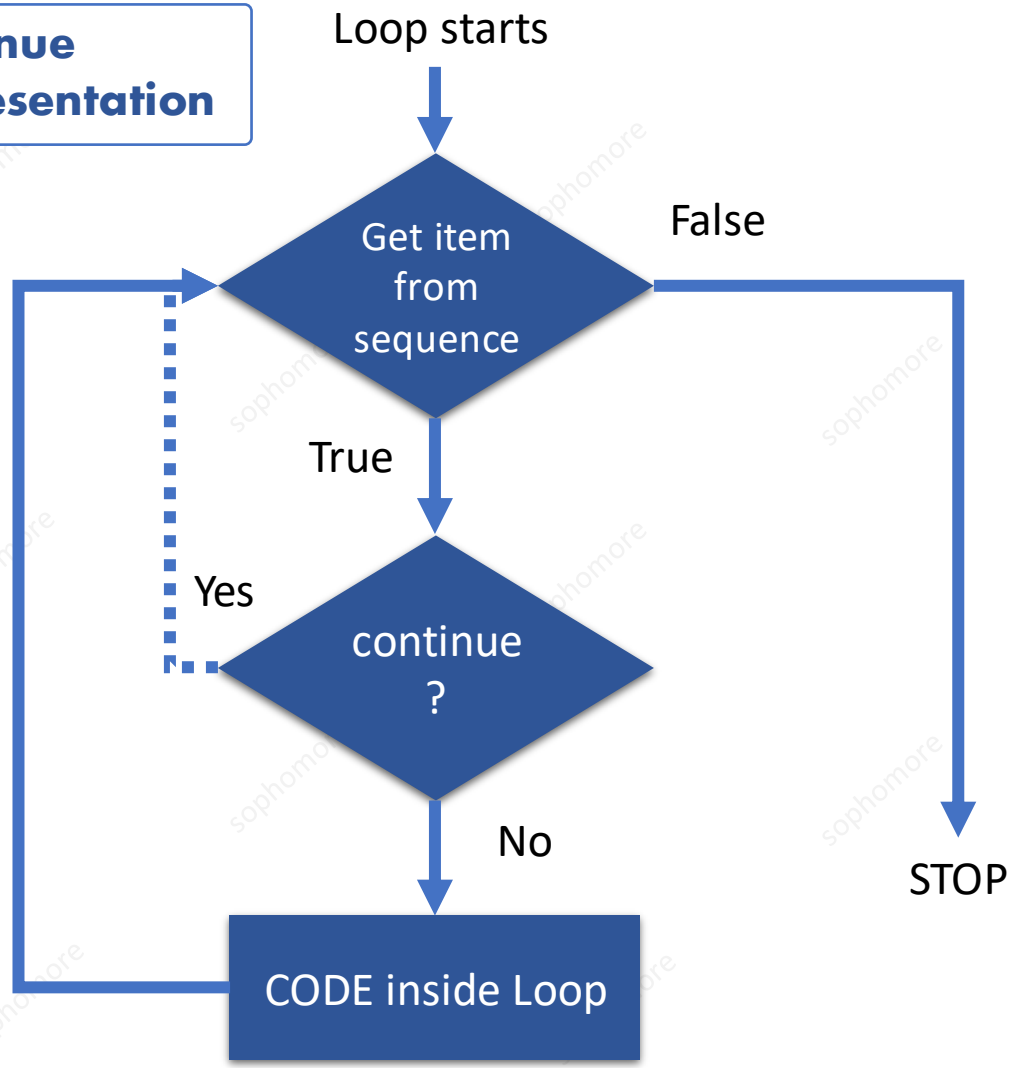
```
while test expression:
    # codes inside while loop
    if  condition:
        break
    # codes inside while loop

# codes outside while loop
```

# Break and Continue

The continue statement is used to skip the rest of the code inside a loop for the current iteration only.

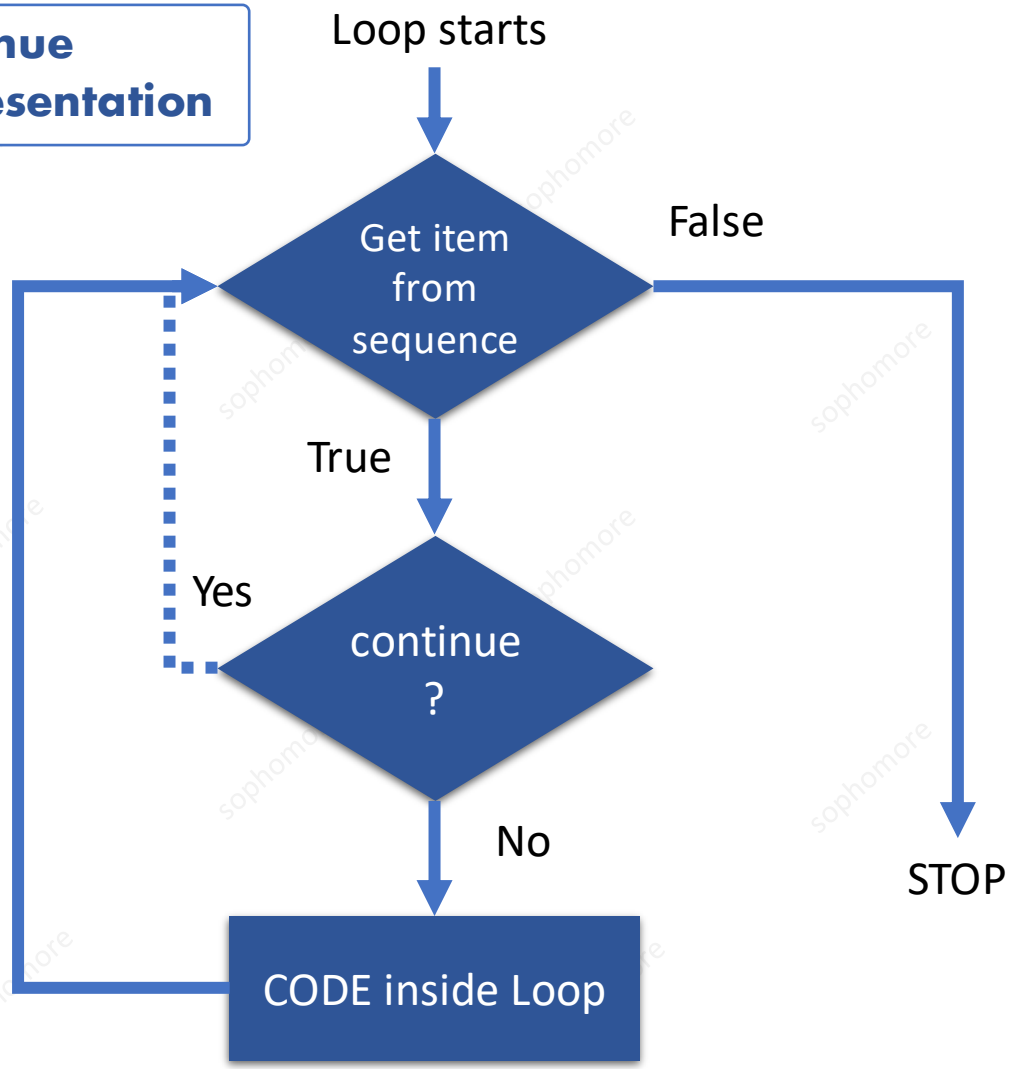
Loop does not terminate but continues on with the next iteration

**continue representation**



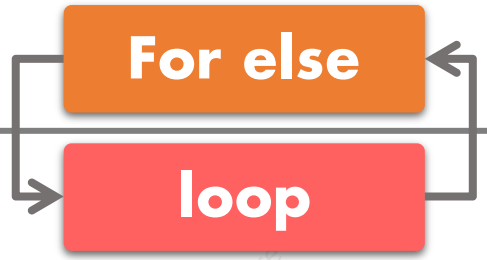
# Break and Continue

**continue representation**




```
for var in sequence:
    # codes inside for loop
    if  condition:
        continue
    # codes inside for loop

# codes outside for loop
```

```
while test expression:
    # codes inside while loop
    if  condition:
        continue
    # codes  inside while loop

# codes outside while loop
```




# Special loop

for loops can also have an optional else clause.

The else clause executes after the loop completes normally

This means that the loop did not encounter a break statement

The **common use** is to loop and search for an item. If the item is found, we break out of the loop using the break statement

```
fruits = ['apple', 'banana', 'mango']
for fruit in fruits:
    print('we have',fruit)
else:
    print('thats all')
```

```
we have apple
we have banana
we have mango
thats all
```

# Le Code time

# Activity

Create this pattern

```
1 2 3 4 5
2 4 6 8 10
3 6 9 12 15
4 8 12 16 20
5 10 15 20 25
```

Assignment for

# level 5

click here

# THE END