Web Application Development using Python

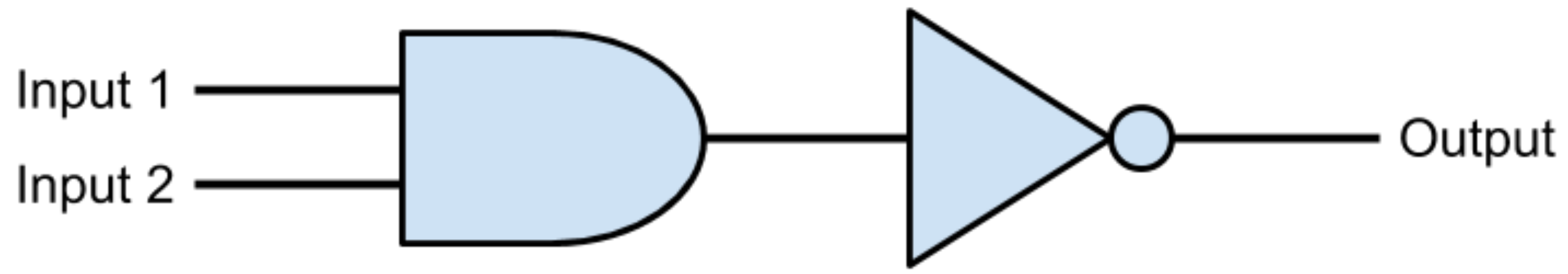
Introduction to Flow Control

Prepared by George Khoury

Outline

- Comparison and logical operators
- **Decisions**
    - `if` statements
- **Loops**
    - `for` statements
    - `while` statements
- `range()` and `enumerate()`
- `break`, `continue`, `pass` statements

Comparison and Logical Operators



Comparison Operators

W1/S3/ex0.py

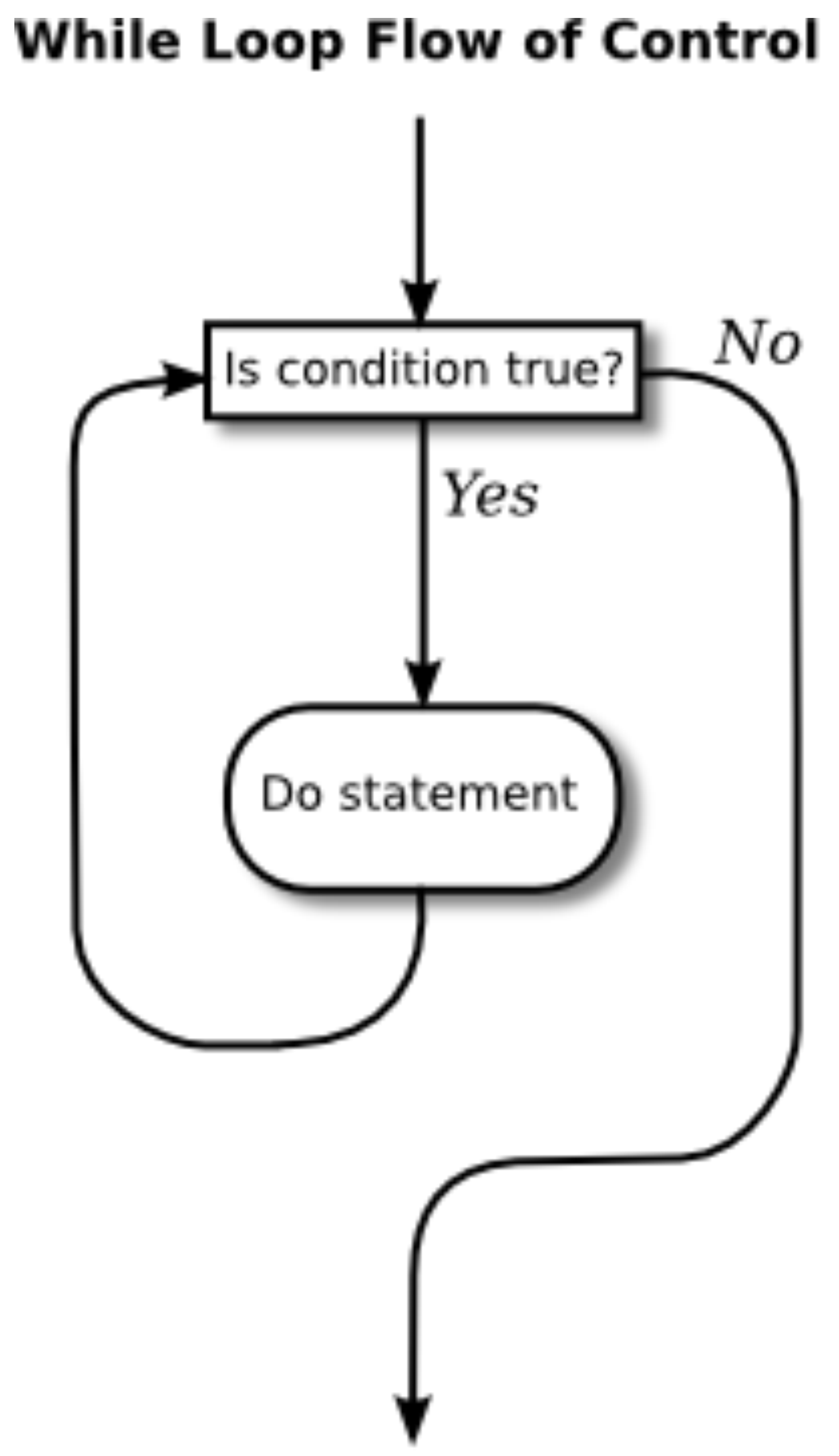
Operator	Meaning	Example
>	Greater than	x > y
<	Less than	x < y
==	Equal to	x == y
!=	Not equal to	x != y
>=	Greater than or equal to	x >= y
<=	Less than or equal to	x <= y

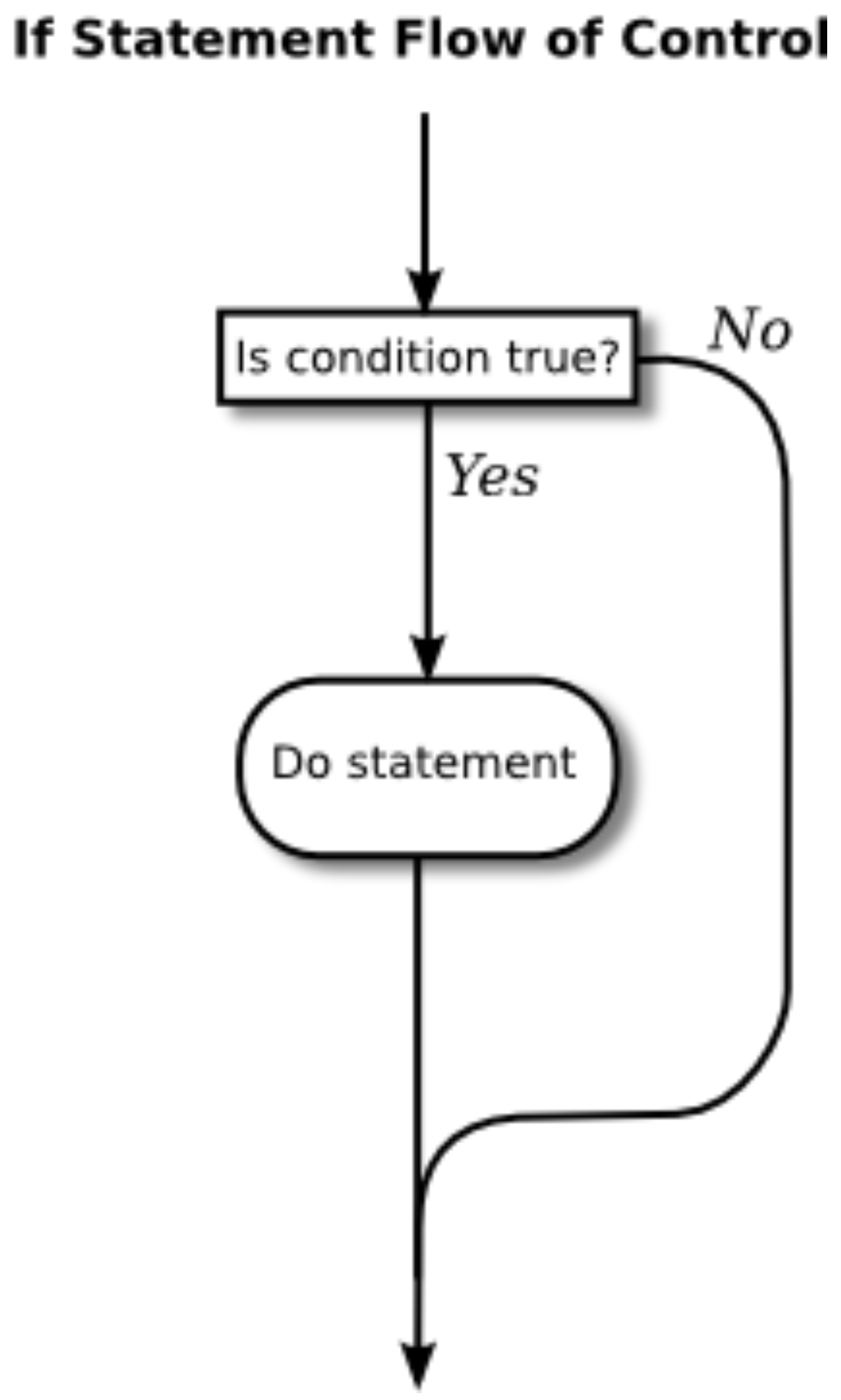
Logical Operators

W1/S3/ex0.py

Operator	**Meaning**	**Example**
and	True if both operands are true	x **and** y
or	True if either of operands is true	x **or** y
not	True if operand is false	**not** x

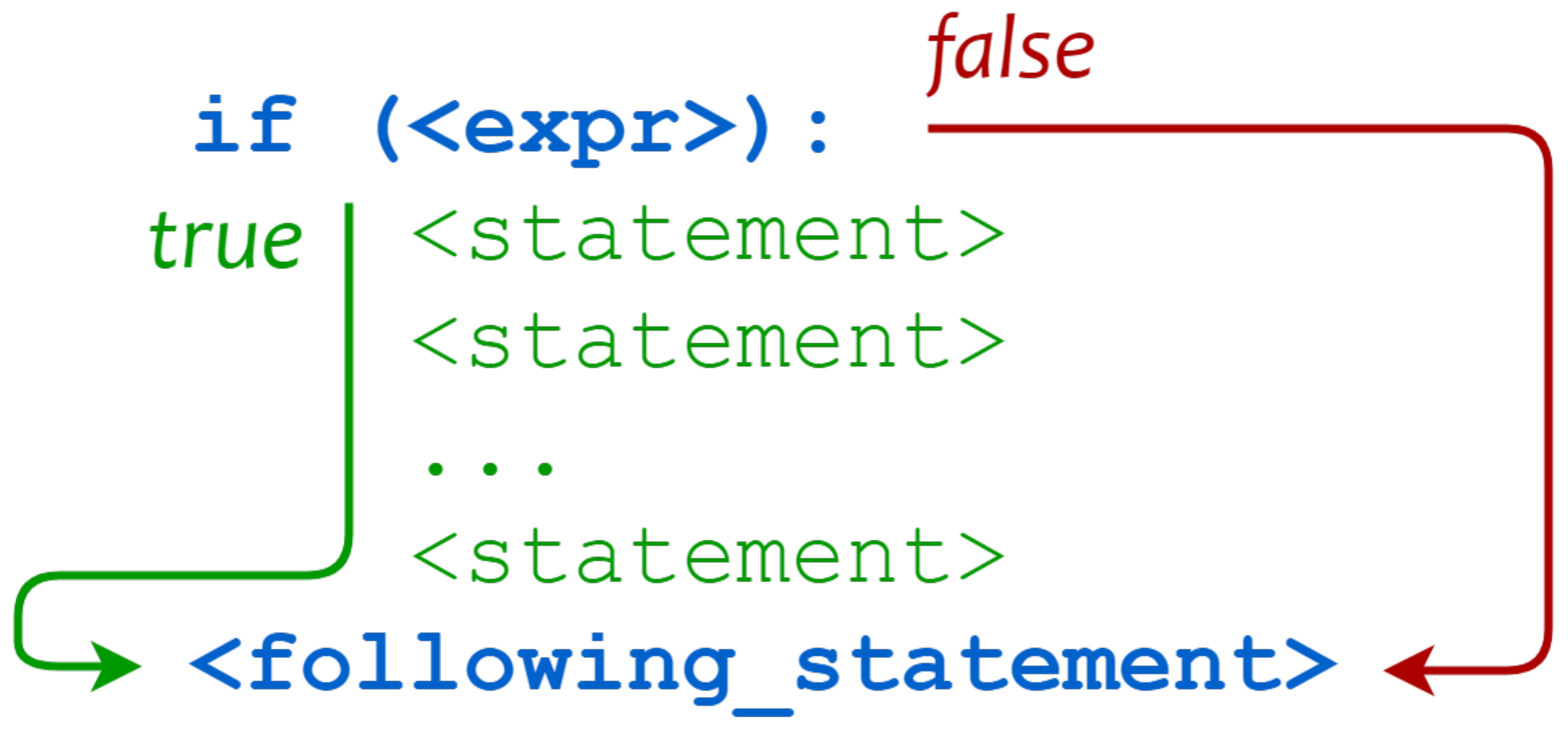
Flow Control





Decisions
W1/S3/ex1.py

- Decisions are an important part of flow control.
- Decisions allow us to execute certain blocks of code when a particular condition is met.
- In Python we use `if` statements to make decisions.
- **Indentation for the `if` block is very crucial in Python.**

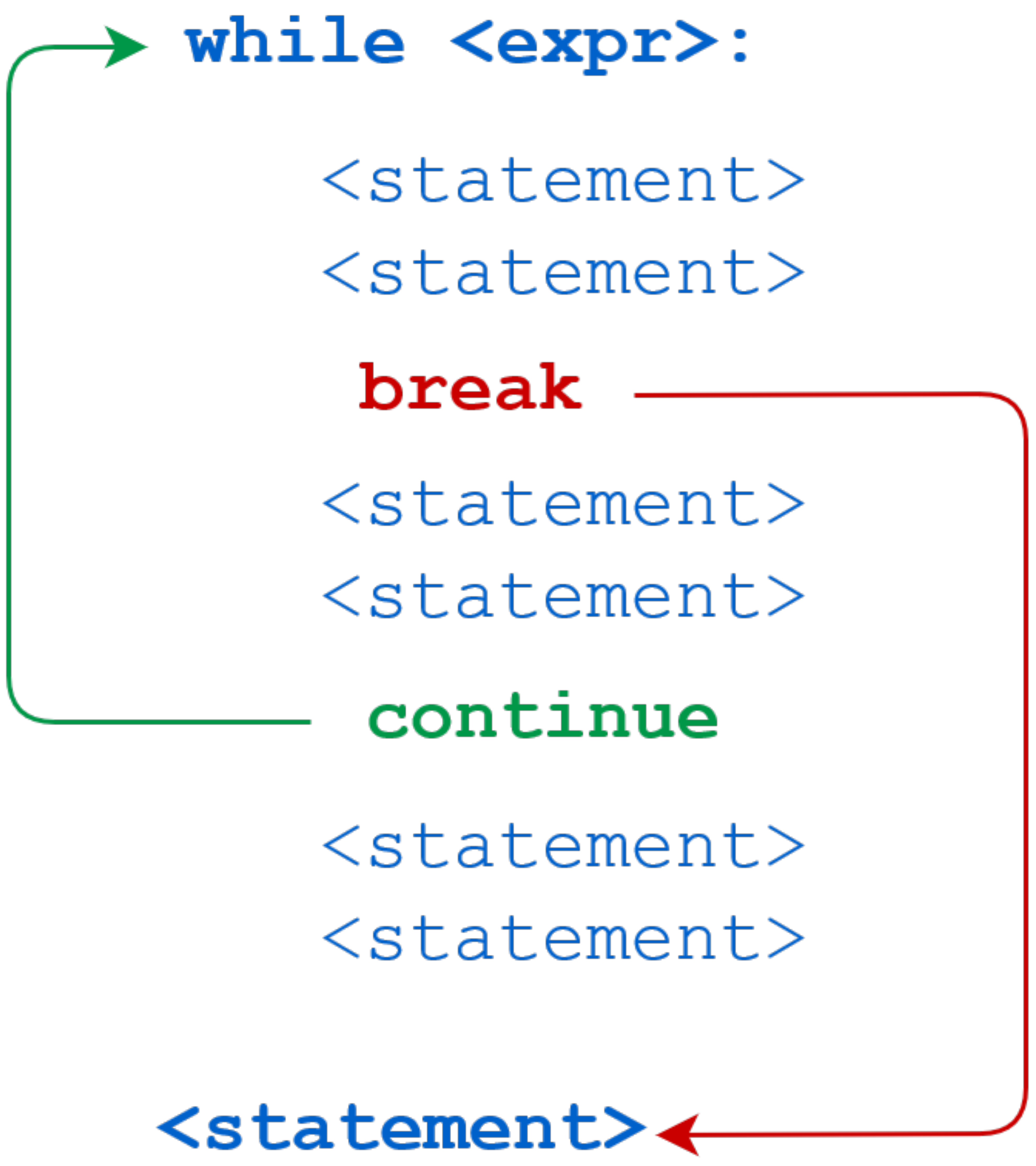


```
if (<expr>):
    <statement>
    <statement>
    ...
    <statement>
<following_statement>
```

Loops

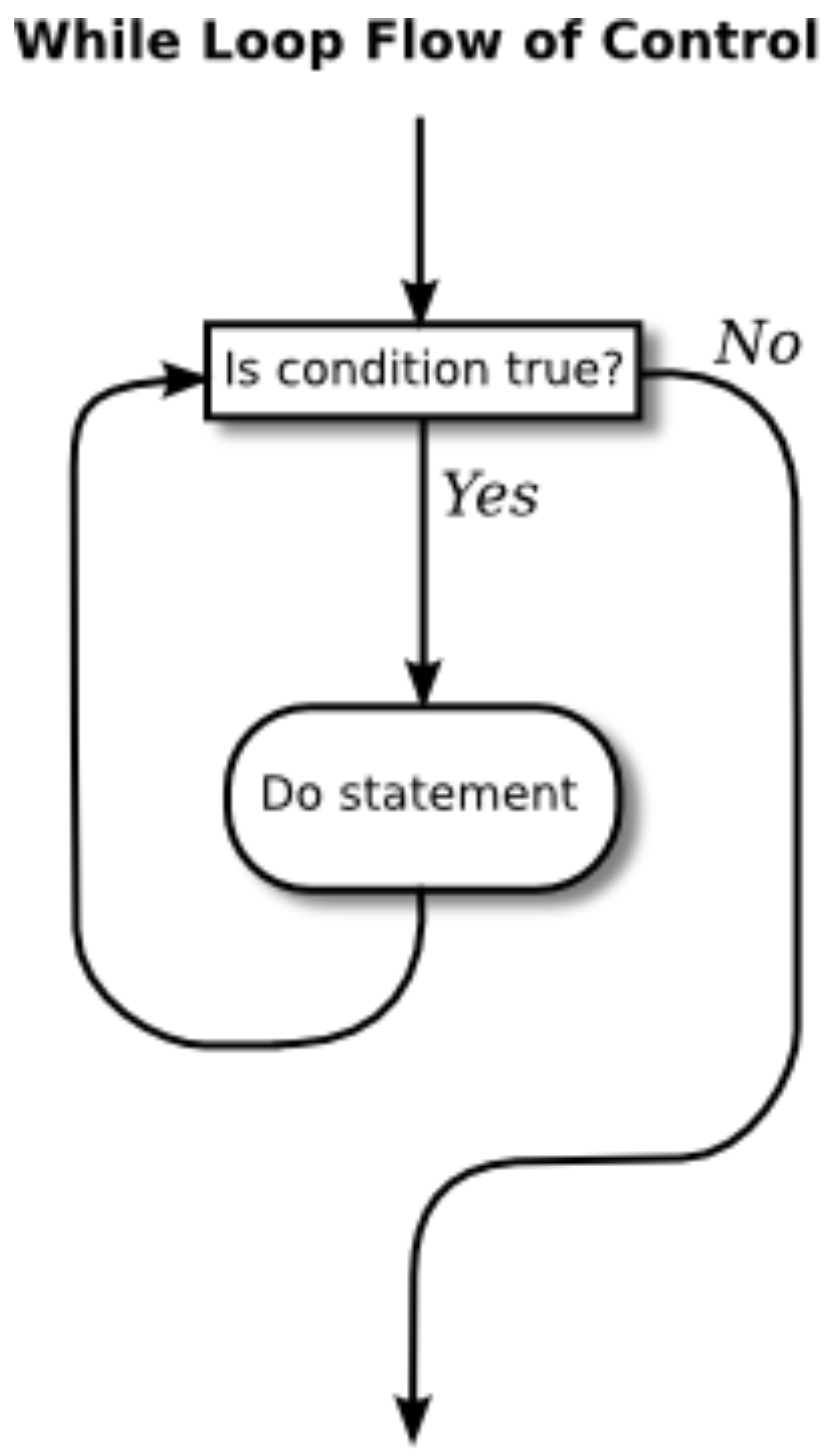
W1/S3/ex2.py

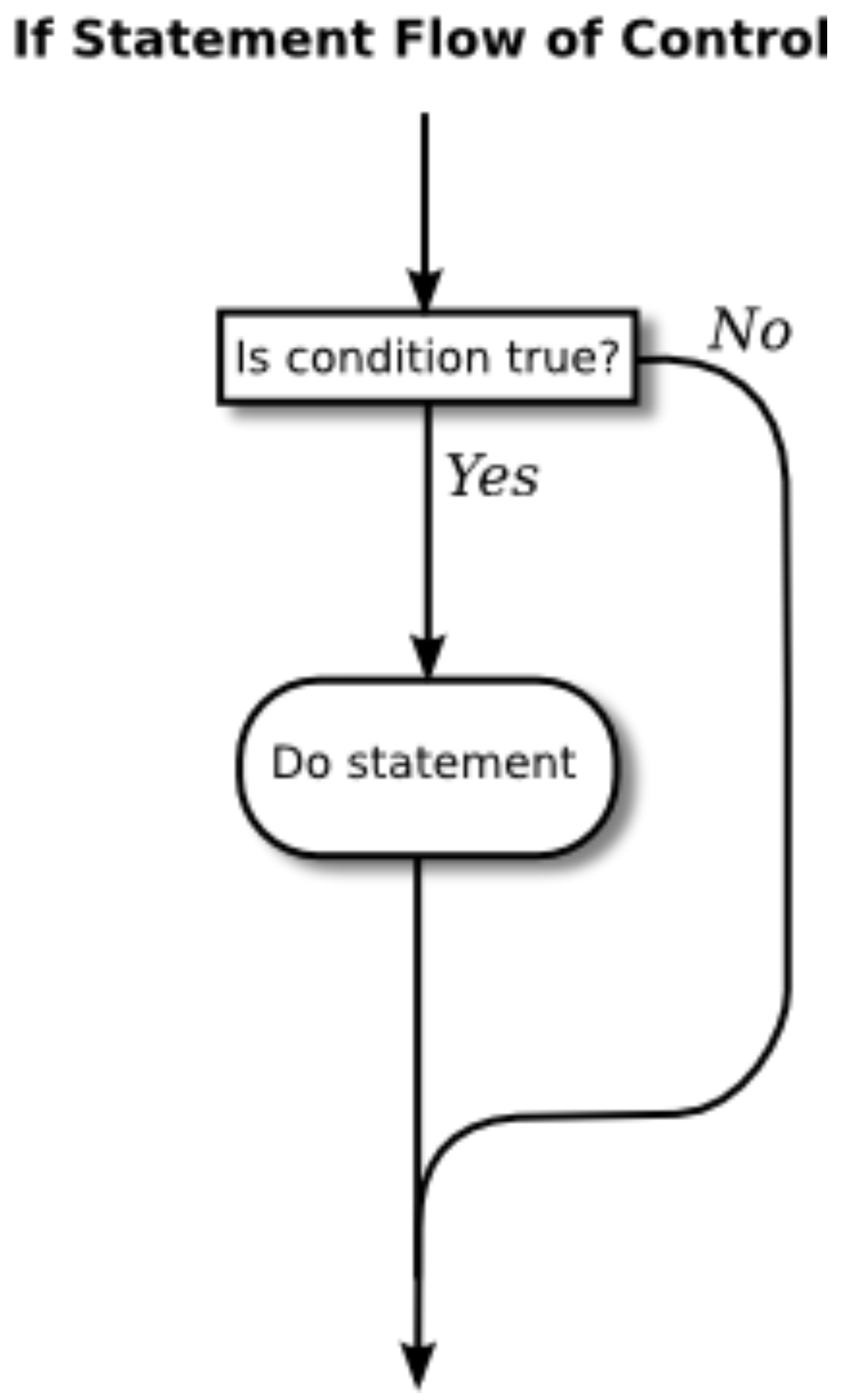
- Many Python objects are "iterable".
  - Which ones did we learn so far?
- Loops allow us to execute certain blocks of code over a controlled number of iterations.
- **Indentation for the `for` or `while` block is very crucial in Python.**



```
while <expr>:
    <statement>
    <statement>
    break
    <statement>
    <statement>
    continue
    <statement>
    <statement>
<statement>
```

`range()` and `enumerate()`

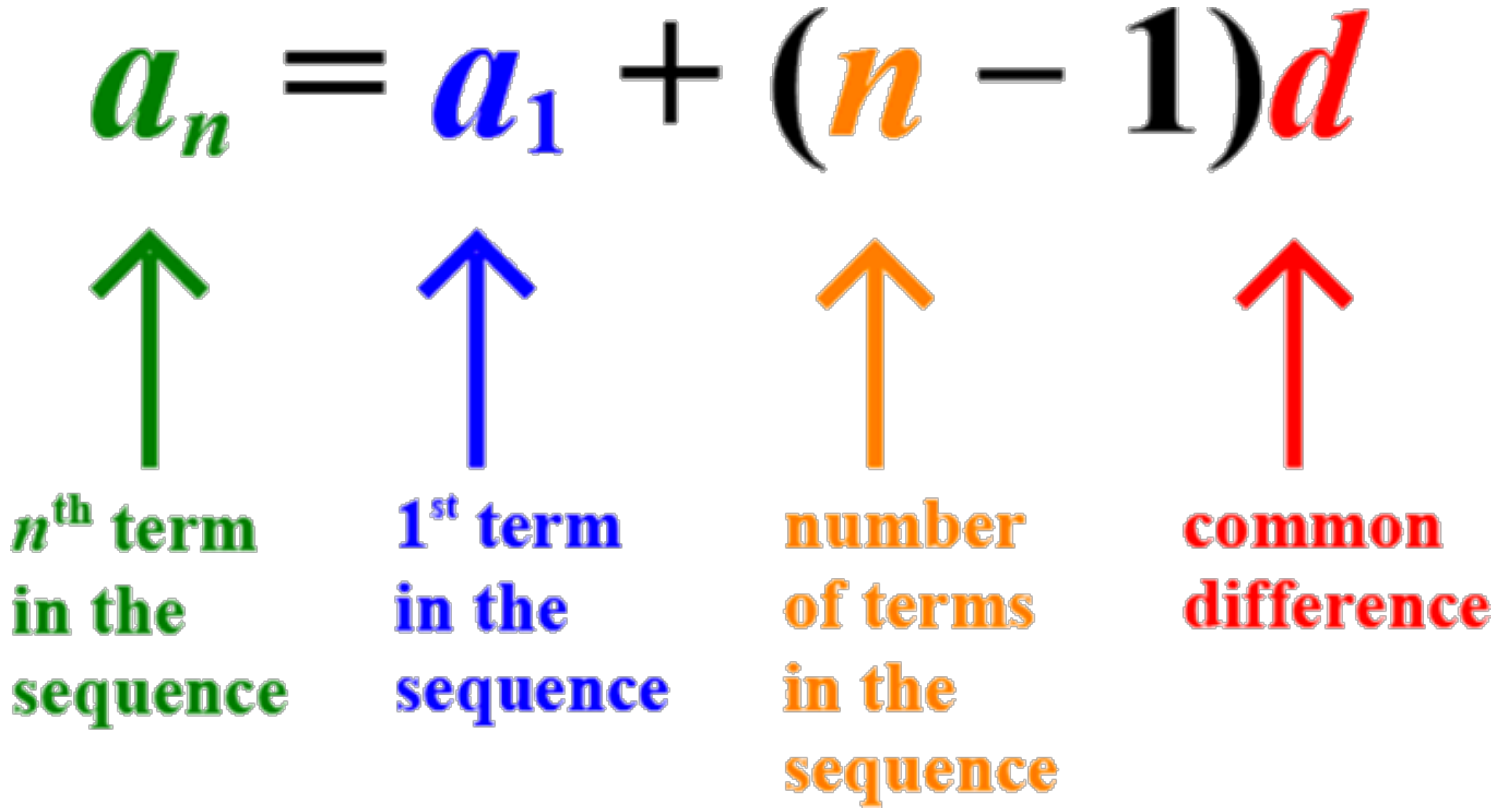




`range()`

W1/S3/ex3.py

- If you need to iterate over a sequence of numbers, the `range()` function can help you generate arithmetic progressions.
- The upper bound is never part of the generated sequence
    - `range(10)` generates 10 values, the legal indices for items of a sequence of length 10.

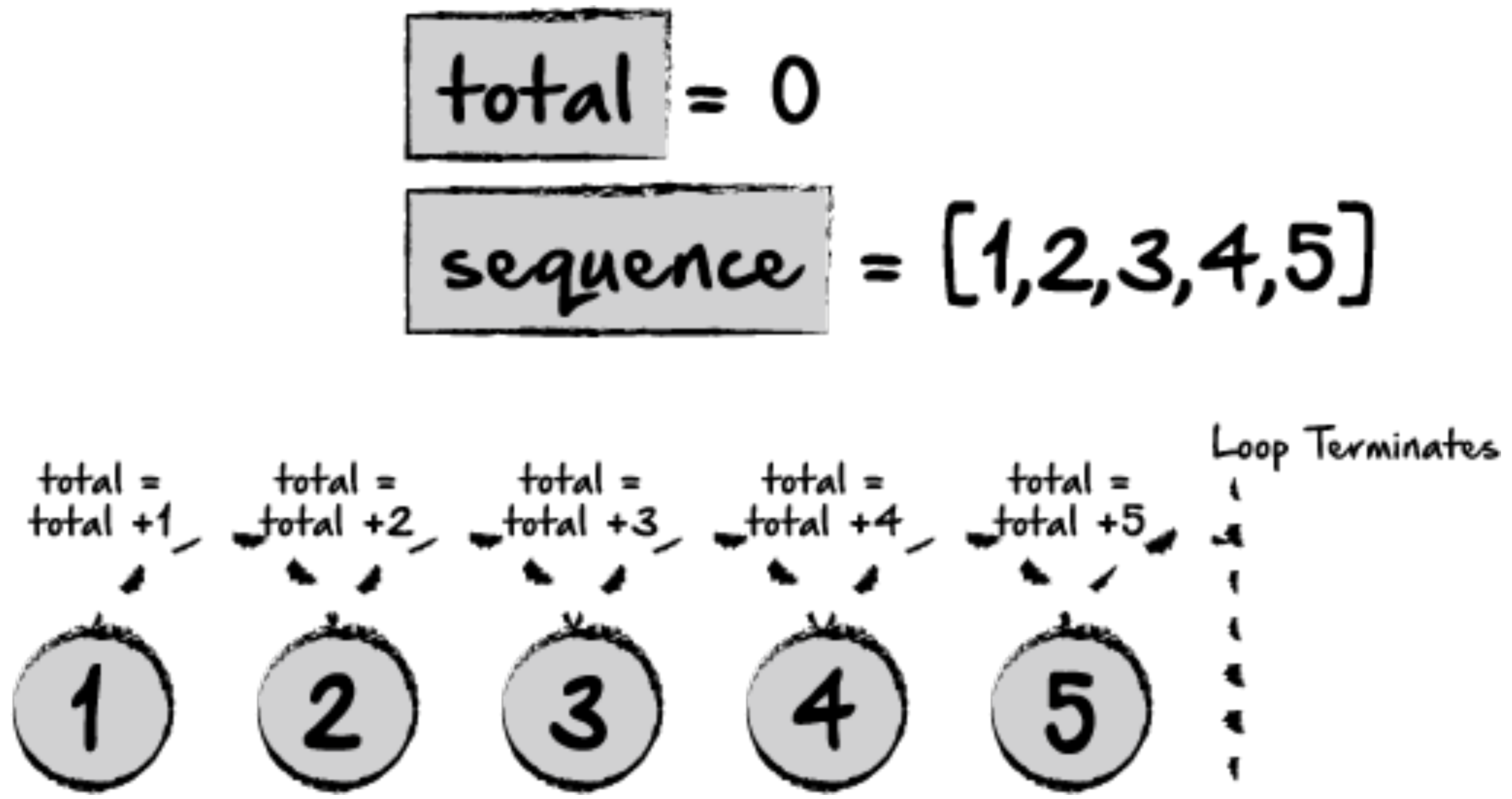


$$a_n = a_1 + (n - 1)d$$

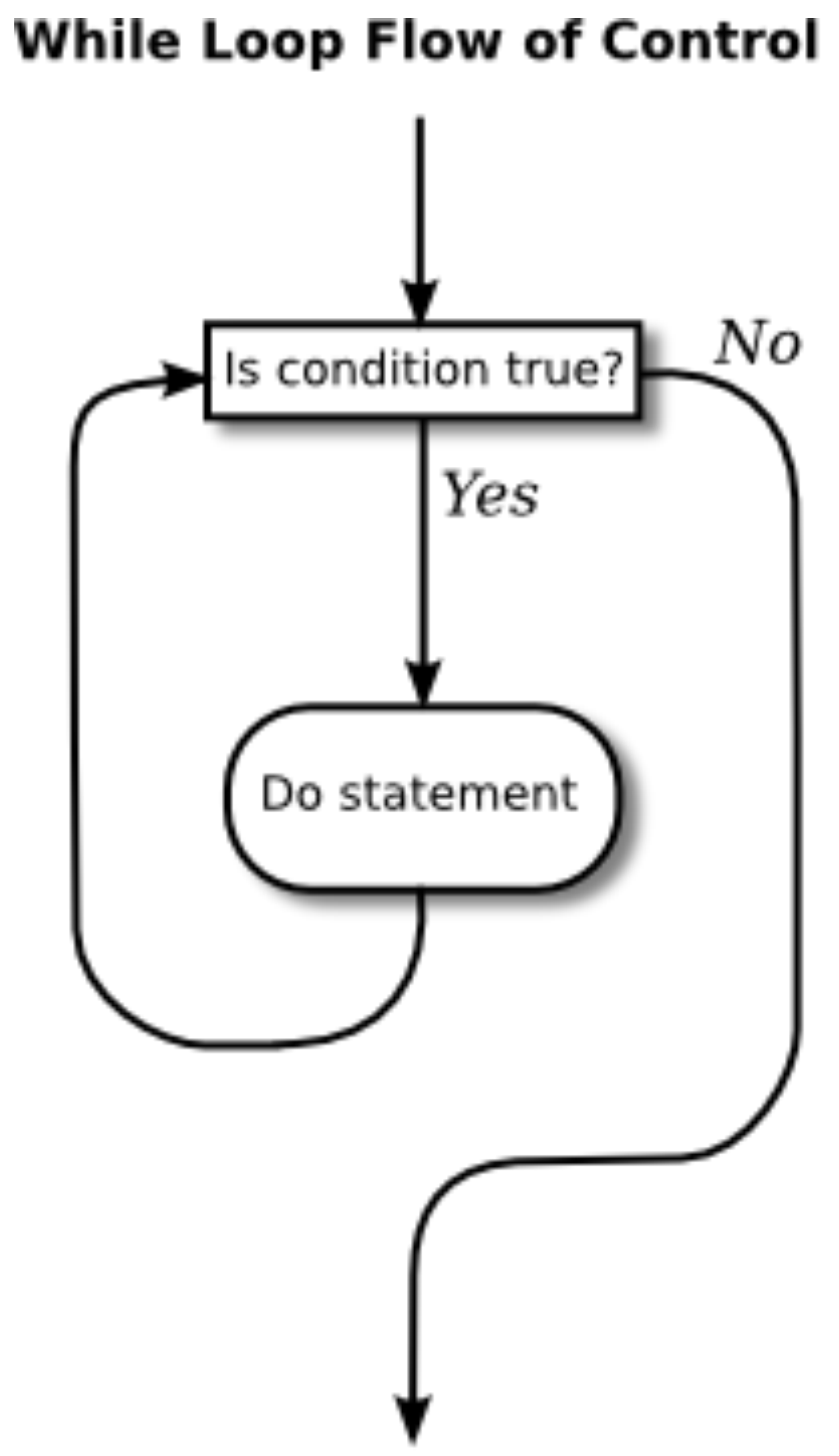
`enumerate()`

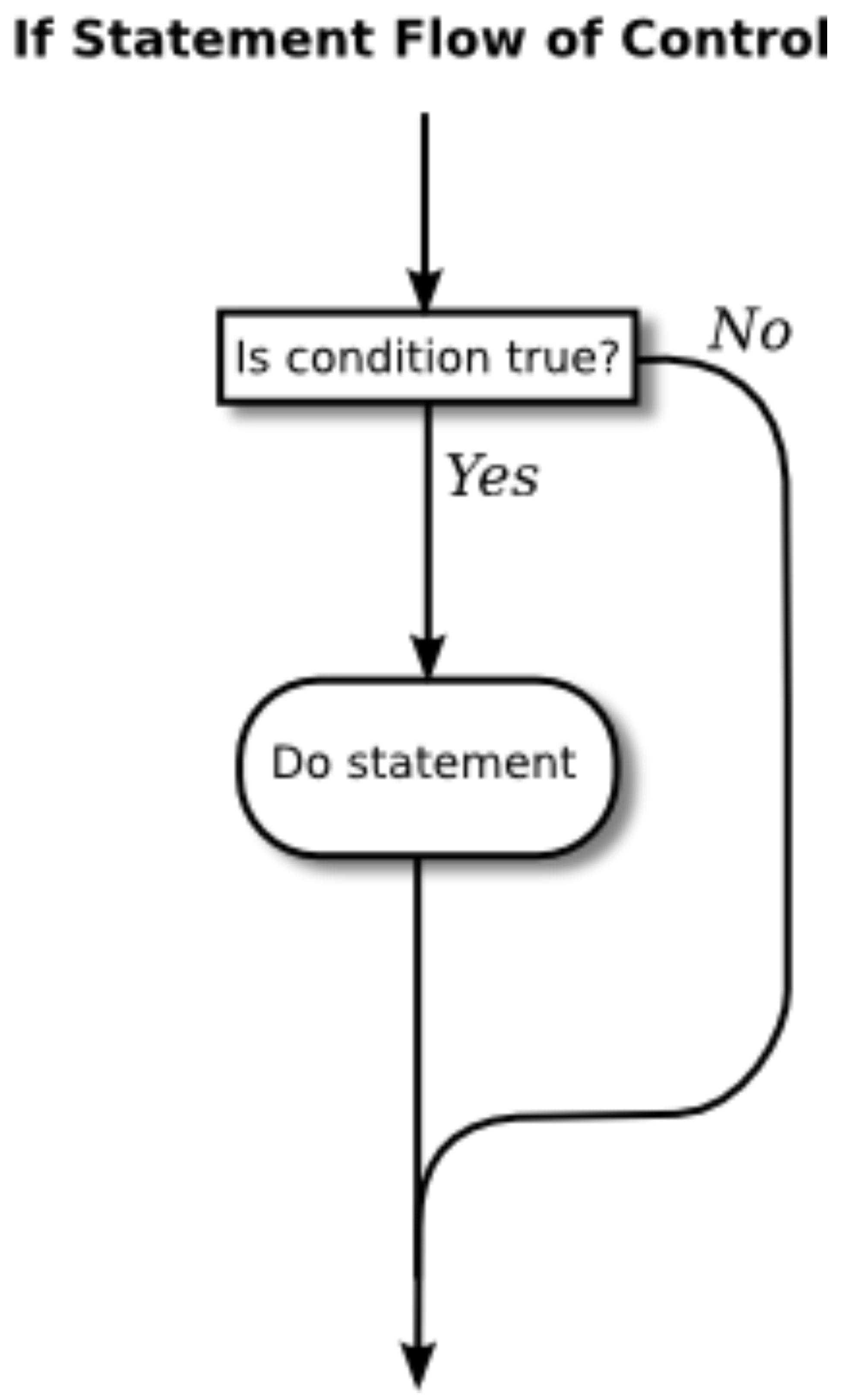
W1/S3/ex3.py

- Python `enumerate()` function can be used to iterate the list in an optimized manner.
- The `enumerate()` function adds a counter to the list or any other iterable and returns it as an enumerate object by the function.
- Thus, it reduces the overhead of keeping a count of the elements while the iteration operation.



`break`, `continue`, `pass` statements

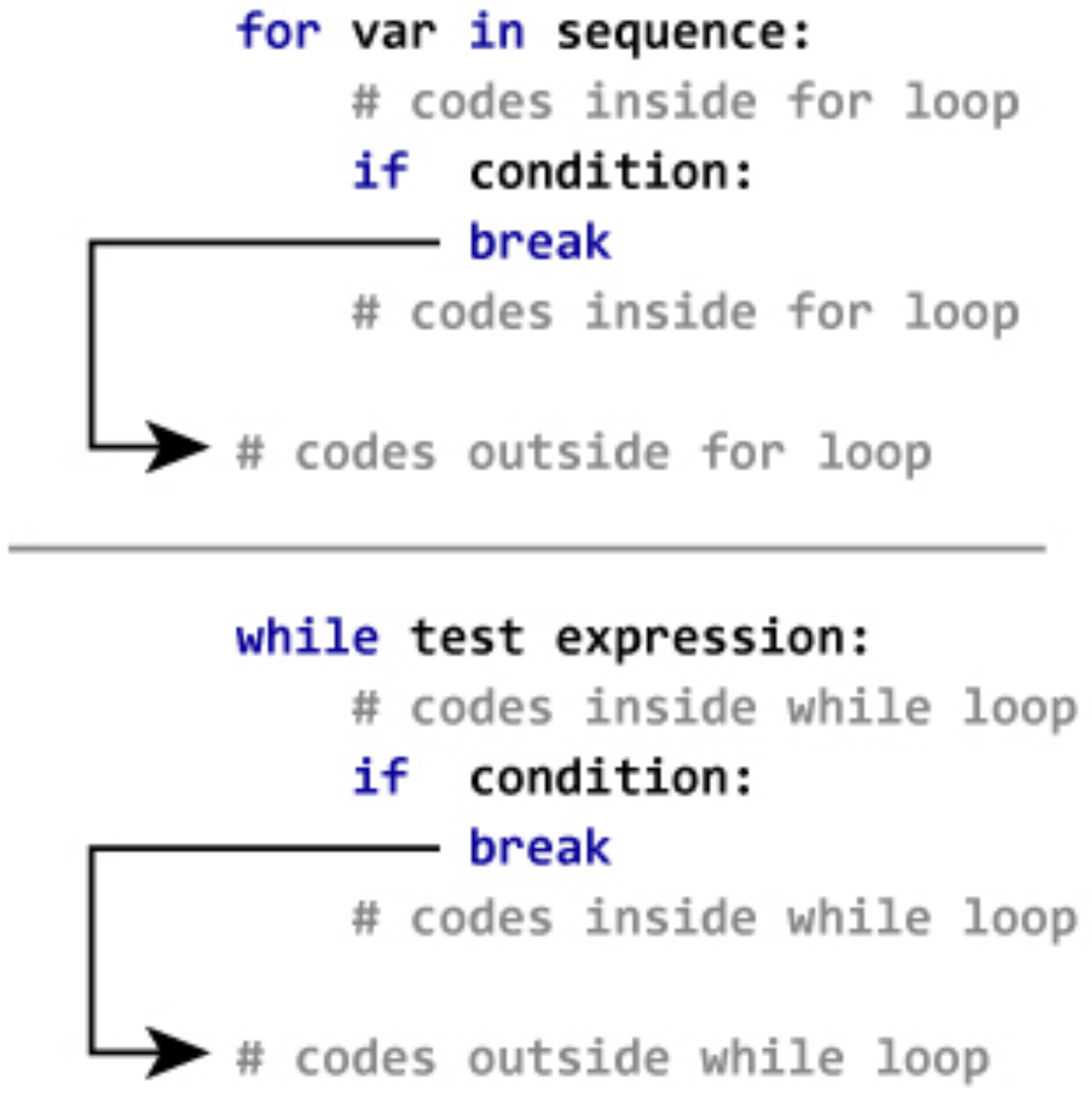




`break` **and** `continue`

W1/S3/ex3.py

- The `break` statement, like in C, breaks out of the innermost enclosing for or while loop.
- The `continue` statement, also borrowed from C, continues with the next iteration of the loop.



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

# codes outside for loop
```

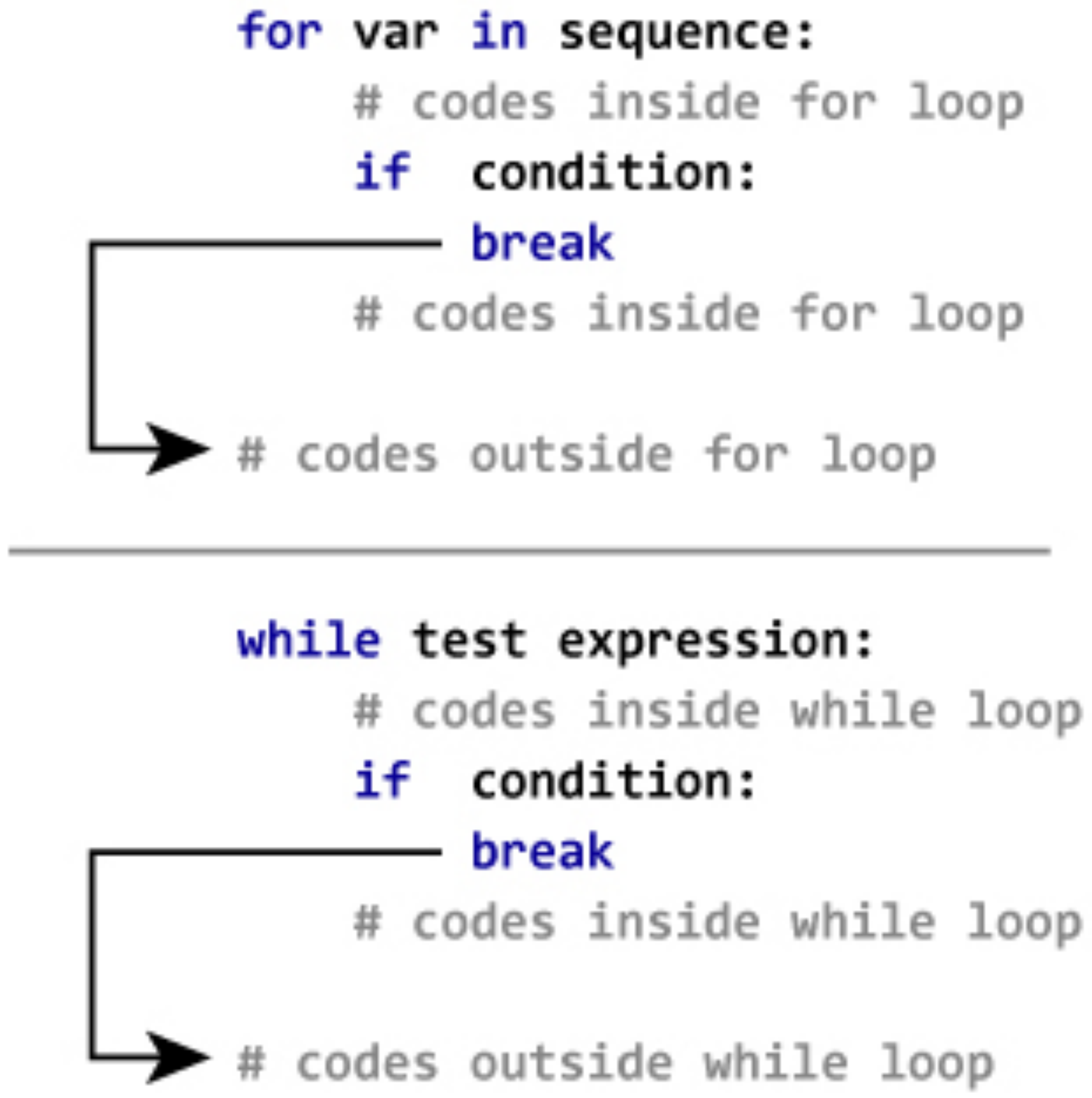
```
while test expression:
    # codes inside while loop
    if  condition:
        break
    # codes inside while loop

# codes outside while loop
```

`pass` statements

W1/S3/ex3.py

- The pass statement does nothing.
- It can be used when a statement is required syntactically but the program requires no action.
- It can also be used as a place-holder for a function or conditional body when you are working on new code, allowing you to keep thinking at a more abstract level.
- The pass is silently ignored.



```
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
```

Learning Resources

- https://docs.python.org/3/tutorial/controlflow.html
- https://docs.python.org/3/tutorial/controlflow.html#if-statements
- https://docs.python.org/3/tutorial/controlflow.html#for-statements
- https://docs.python.org/3/tutorial/controlflow.html#the-range-function
- https://docs.python.org/3/library/functions.html#enumerate
- https://docs.python.org/3/tutorial/controlflow.html#break-and-continue-statements-and-else-clauses-on-loops
- https://docs.python.org/3/tutorial/datastructures.html#tut-loopidioms