

Comparison Operators

Comparison operations compare some value or operand & based on a condition, produce a Boolean.

Python has six comparison operators as below:

- Less than (<)
- Less than or equal to (<=)
- Greater than (>)
- Greater than or equal to (>=)
- Equal to (==)
- Not equal to (!=)

```
● ● ●

# Take a variable
golden_ratio = 1.618
# Condition less than
print(golden_ratio<2) # The golden ratio is lower than 2, thus the
output is True
print(golden_ratio<1) # The golden ratio is greater than 1, thus
the output is False
```

```
<class 'set'>
<class 'set'>
```

```
● ● ●

# Take a variable
golden_ratio = 1.618
# Condition less than or equal to
print(golden_ratio<=2) # The golden ratio is lower than 2, thus the
condition is True.
print(golden_ratio<=1) # The golden ratio is greater than 1, thus
the condition is False.
print(golden_ratio<=1.618) # The golden ratio is equal to 1.618,
thus the condition is True.
```

```
True
False
True
```

```
● ● ●

# Take a variable
golden_ratio = 1.618
# Condition greater than
print(golden_ratio>2) # The golden ratio is lower than 2, thus the
condition is False.
print(golden_ratio>1) # The golden ratio is greater than 1, thus
the condition is True.
```

```
False
True
```

```
● ● ●

# Take a variable
golden_ratio = 1.618
# Condition greater than or equal to
print(golden_ratio>=2) # The golden ratio is not greater than 2,
thus the condition is False.
print(golden_ratio>=1) # The golden ratio is greater than 1, thus
the condition is True.
print(golden_ratio>=1.618) # The golden ratio is equal to 1.618,
thus the condition is True.
```

```
False
True
True
```




```
# Take a variable
golden_ratio = 1.618
# Condition equal to
print(golden_ratio==2) # The golden ratio is not equal to 1.618,
thus the condition is False.
print(golden_ratio==1.618) # The golden ratio is equal to 1.618,
thus the condition is True.
```

False
True



```
# Take a variable
golden_ratio = 1.618
# Condition not equal to
print(golden_ratio!=2) # The golden ratio is not equal to 1.618,
thus the condition is True.
print(golden_ratio!=1.618) # The golden ratio is equal to 1.618,
thus the condition is False.
```

True
False

The comparison operators are also employed to compare the letters/words/symbols according to the ASCII

(<https://www.asciitable.com/>) value of letters.



```
# Compare strings
print('Hello' == 'Python')
print('Hello' != 'Python')
print('Hello' <= 'Python')
print('Hello' >= 'Python')
print('Hello' < 'Python')
print('Hello' > 'Python')
print('B'>'A') # According to ASCII table, the values of A and B
are equal 65 and 66, respectively.
print('a'>'b') # According to ASCII table, the values of a and b
are equal 97 and 98, respectively.
print('CD'>'DC') # According to ASCII table, the value of C (67) is
lower than that of D (68)
# The values of uppercase and lowercase letters are different since
python is case sensitive.
```

False
True
True
False
True
False
True
False
False



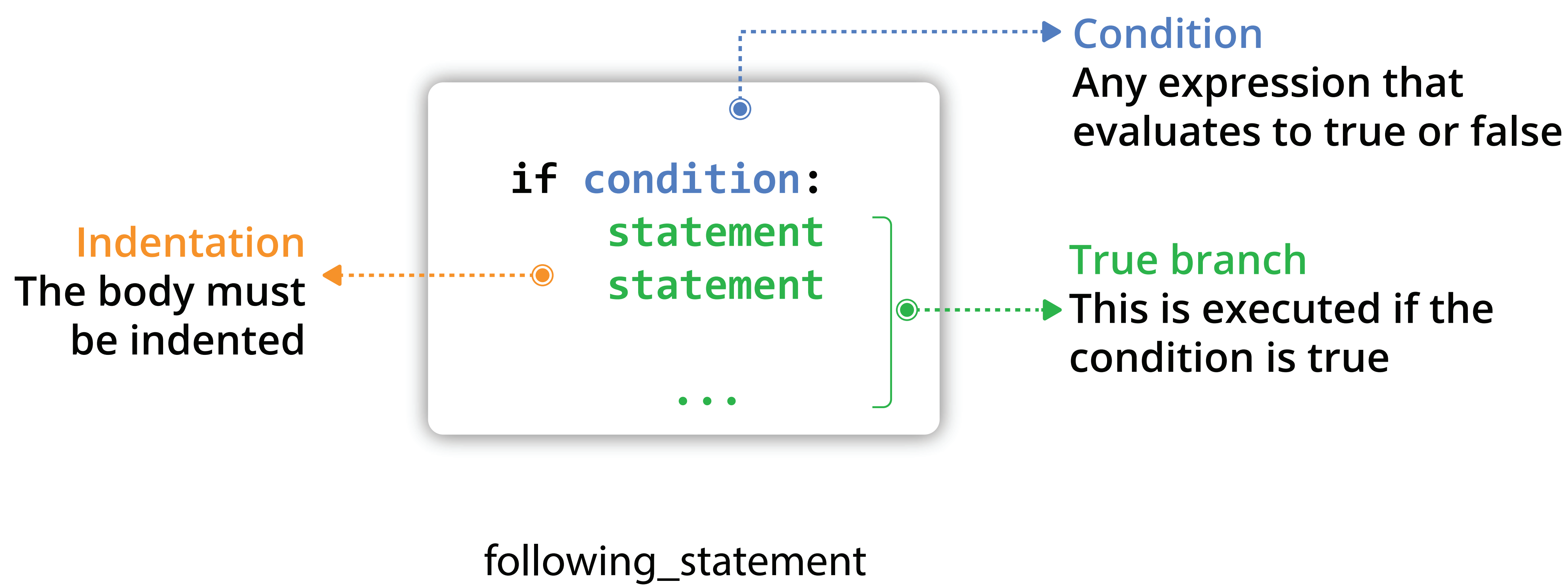
```
'M'>'Ython'
```

False

Branching (if, elif, else)

- Decision making is required when we want to execute a code only if a certain condition is satisfied.
- The if/elif/else statement is used in Python for decision making.
- An else statement can be combined with an if statement.
- An else statement contains the block of code that executes if the conditional expression in the if statement resolves to 0 or a False value
- The else statement is an optional statement and there could be at most only one else statement following if.
- The elif statement allows you to check multiple expressions for True and execute a block of code as soon as one of the conditions evaluates to True.
- Similar to the else, the elif statement is optional.
- However, unlike else, for which there can be at most one statement, there can be an arbitrary number of elif statements following an if.

If Statement



```
pi = 3.14
golden_ratio = 1.618
# This statement can be True or False.
if pi > golden_ratio:
    # If the conditions is True, the following statement will be print-
    # ed.
    print(f'The number pi {pi} is greater than the golden ratio {gold-
    en_ratio}.')
# The following statement will be printed in each situtation.
print('Done!')
```

The number pi 3.14 is greater than the golden ratio 1.618.
Done!

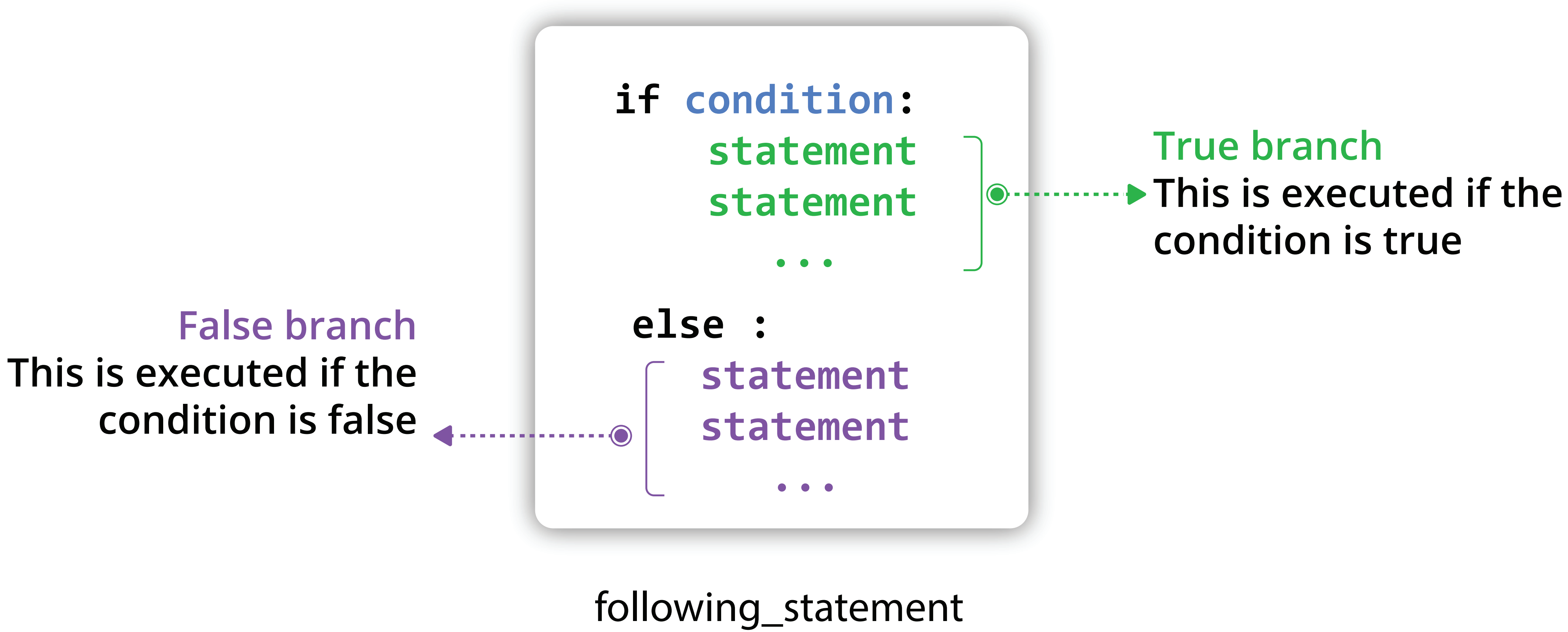
```
if 2:
    print('Hello, python!')
```

Hello, python!

```
if True:
    print('This is true.')
```

This is true.

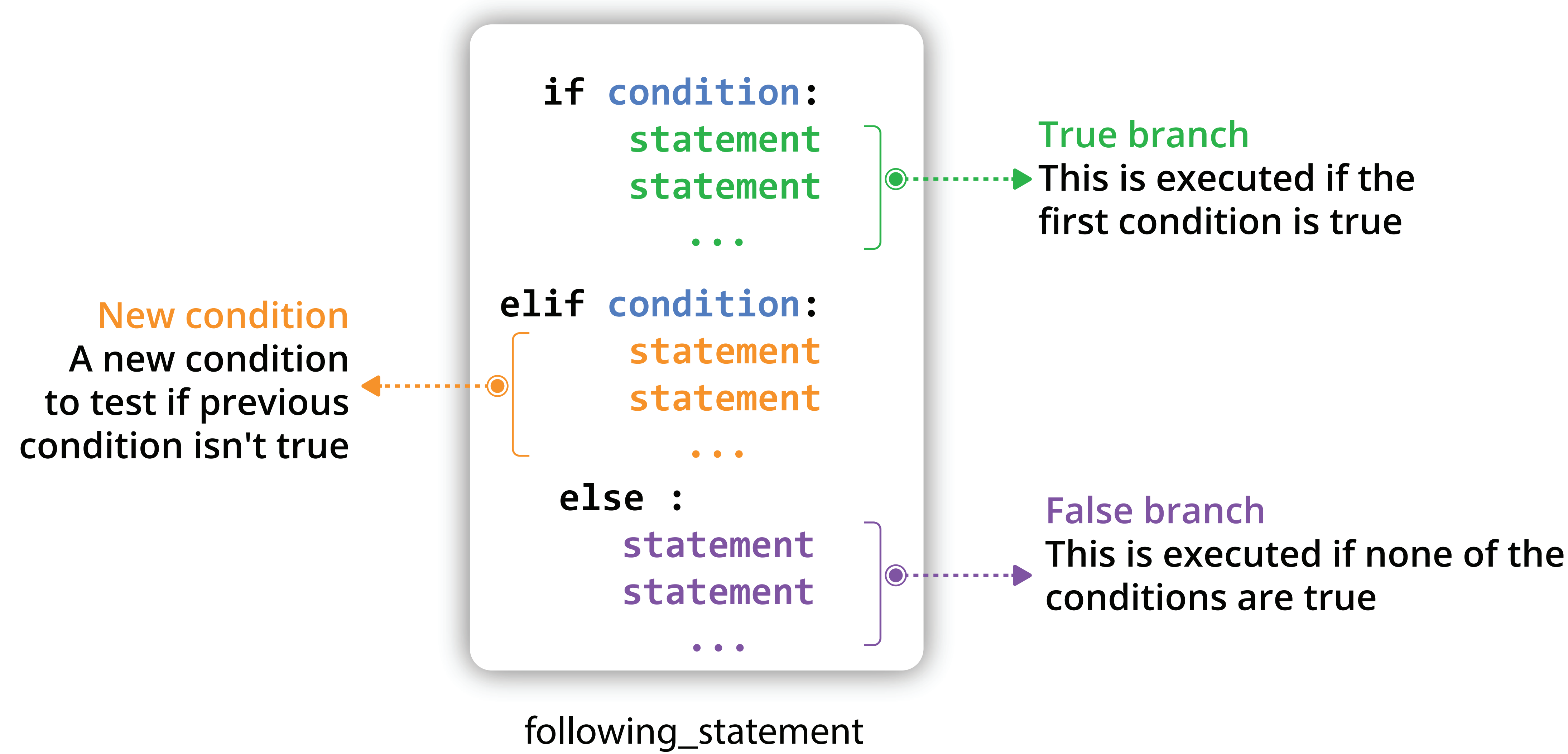
else Statement



```
pi = 3.14
golden_ratio = 1.618
if pi < golden_ratio:
    print(f'The number pi {pi} is greater than the golden ratio
    {golden_ratio}.')
else:
    print(f'The golden ratio {golden_ratio} is lower than the number
    pi {pi}.')
print('Done!')
```

The golden ratio 1.618 is lower than the number pi 3.14.
Done!

elif Statement



```
age = 5
if age > 6:
    print('You can go to primary school.' )
elif age == 5:
    print('You should go to kindergarten.')
else:
    print('You are a baby' )

print('Done!')
```

You should go to kindergarten.
Done!


```
album_year = 2000
album_year = 1990
if album_year >= 1995:
    print('Album year is higher than 1995.')
print('Done!')
```

Done!

```
album_year = 2000
# album_year = 1990
if album_year >= 1995:
    print('Album year is higher than 1995.')
else:
    print('Album year is lower than 1995.')
print('Done!')
```

Album year is higher than 1995.
Done!

```
imdb_point = 9.0
if imdb_point > 8.5:
    print('The movie could win Oscar.')
```

The movie could win Oscar.

```
movie_rating = float(input('Enter a rating number:'))
print(f'The entered movie rating is: {movie_rating}')
if movie_rating > 8.5:
    print('The movie is awesome with {} rating and you should watch it.'.format(movie_rating))
else:
    print('The movie has merit to be watched with {} rating.'.format(movie_rating))
```

Enter a rating number:7
The entered movie rating is: 7.0
The movie has merit to be watched with 7.0 rating.

```
note = float(input('Enter a note:'))
print(f'The entered note value is: {note}')
if note >= 90 and note <= 100:
    print('The letter grade is AA.')
elif note >= 85 and note <= 89:
    print('The letter grade is BA.')
elif note >= 80 and note <= 84:
    print('The letter grade is BB.')
elif note >= 75 and note <= 79:
    print('The letter grade is CB.')
elif note >= 70 and note <= 74:
    print('The letter grade is CC.')
elif note >= 65 and note <= 69:
    print('The letter grade is DC.')
elif note >= 60 and note <= 64:
    print('The letter grade is DD.')
elif note >=55 and note <= 59:
    print('The letter grade is ED.')
elif note >=50 and note <= 54:
    print('The letter grade is EE.')
elif note >=45 and note <=49:
    print('The letter grade is FE.')
else:
    print('The letter grade is FF.')
```

Enter a note:8
The entered note value is: 8.0
The letter grade is FF.

```
number = int(input('Enter a number:'))
print(f'The entered number is: {number}')
if number %2 == 0:
    print(f'The entered number {number} is even')
else:
    print(f'The entered number {number} is odd')
```

Logical Operators

Logical operators are used to combine conditional statements.

- and: Returns True if both statements are true
- or: Returns True if one of the statements is true
- not: Reverse the result, returns False if the result is true

Python Logical Operation

A	B	A and B
True	True	True
True	False	False
False	True	False
False	False	False

A	B	A or B
True	True	True
True	False	True
False	True	True
False	False	False

A	Not A
True	False
False	True

and

• • •

```
birth_year = 1990
if birth_year > 1989 and birth_year < 1995:
    print('You were born between 1990 and 1994')
print('Done!')
```

• • •

```
x = int(input('Enter a number:'))
y = int(input('Enter a number: '))
z = int(input('Enter a number:'))
print(f'The entered numbers for x, y, and z are {x}, {y}, and {z}, respectively.')
if x>y and x>z:
    print(f'The number x with {x} is the greatest number.')
elif y>x and y>z:
    print(f'The number y with {y} is the greatest number.')
else:
    print(f'The number z with {z} is the greatest number.')
```

OR

• • •

```
birth_year = 1990
if birth_year < 1980 or birth_year > 1989:
    print('You were not born in 1980s.')
else:
    print('You were born in 1990s.')
print('Done!')
```

NOT

• • •

```
birth_year = 1990
if not birth_year == 1991:
    print('The year of birth is not 1991.')
```

• • •

```
birth_year = int(input('Enter a year of birth: '))
print(f'The entered year of birth is: {birth_year}')
if birth_year < 1985 or birth_year == 1991 or birth_year == 1995:
    print(f'You were born in {birth_year}')
else:
    # For instance, if your year of birth is 1993
    print(f'Your year of birth with {birth_year} is wrong.')
```

• • •

```
birth_year = int(input('Enter a year of birth: '))
print(f'The entered year of birth is: {birth_year}')
if birth_year < 1985 or birth_year == 1991 or birth_year == 1995:
    # For instance, if your year of birth is 1995
    print(f'You were born in {birth_year}')
else:
    print(f'Your year of birth with {birth_year} is wrong.')
```

Loops in Python

- A for loop is used for iterating over a sequence (that is either a list, a tuple, a dictionary, a set, or a string).
- This is less like the for keyword in other programming languages, and works more like an iterator method as found in other object-orientated programming languages.
- With the for loop we can execute a set of statements, once for each item in a list, tuple, set etc.
- The for loop does not require an indexing variable to set beforehand.
- With the while loop we can execute a set of statements as long as a condition is true.
- Note: remember to increment i, or else the loop will continue forever.
- The while loop requires relevant variables to be ready, in this example we need to define an indexing variable, i, which we set to 1

range() function

- It is helpful to think of the range object as an ordered list.
- To loop through a set of code a specified number of times, we can use the range() function,
- The range() function returns a sequence of numbers, starting from 0 by default, and increments by 1 (by default), and ends at a specified number

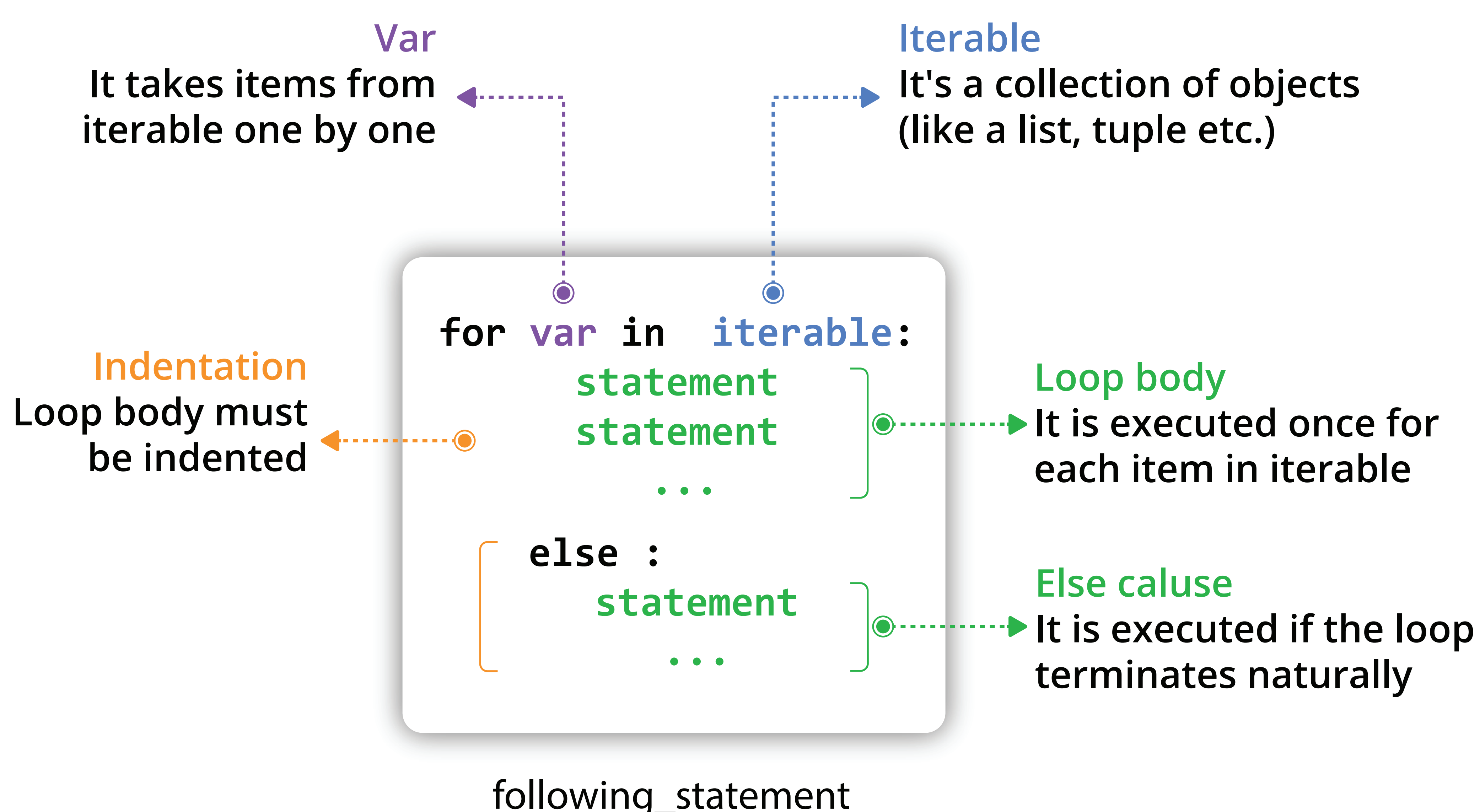
• • •

```
# Take a range() function
print(range(5))
print(range(10))
```

```
range(0, 5)
range(0, 10)
```


for loop

The for loop enables you to execute a code block multiple times.



```
# Take an example
# Diectly accessing to the elements in the list
years = [2005, 2006, 2007, 2008, 2009, 2010]
for i in years:
    print(i)
```

2005
2006
2007
2008
2009
2010

```
# Again, directly accessing to the elements in the list
years = [2005, 2006, 2007, 2008, 2009, 2010]
for year in years:
    print(year)
```

2005
2006
2007
2008
2009
2010

```
# Take an example
years = [2005, 2006, 2007, 2008, 2009, 2010]
for i in range(len(years)):
    print(years[i])
```

2005
2006
2007
2008
2009
2010

```
# Another for loop example
for i in range(2, 12):
    print(i)
```

2
3
4
5
6
7
8
9
10
11

```
# Striding in for loop
for i in range(2, 12, 3):
    print(i)
```

2
5
8
11

```
# Changing the elements in the list
languages = ['Java', 'JavaScript', 'C', 'C++', 'PHP']
for i in range(len(languages)):
    print('Before language', i, 'is', languages[i])
    languages[i] = 'Python'
    print('After language', i, 'is', languages[i])
```

Before language 0 is Java
After language 0 is Python
Before language 1 is JavaScript
After language 1 is Python
Before language 2 is C
After language 2 is Python
Before language 3 is C++
After language 3 is Python
Before language 4 is PHP
After language 4 is Python



```
# Enumaeration of the elements in the list
languages = ['Python', 'Java', 'JavaScript', 'C', 'C++', 'PHP']
for index, language in enumerate(languages):
    print(index, language)
```

0 Python
1 Java
2 JavaScript
3 C
4 C++
5 PHP



```
# Take the numbers between -3 and 6 using for loop
# Use range() function
for i in range(-3, 7):
    print(i)
```

-3
-2
-1
0
1
2
3
4
5
6



```
# Take a list and print the elements using for loop
languages = ['Python', 'Java', 'JavaScript', 'C', 'C++', 'PHP']
for i in range(len(languages)):
    print(i, languages[i])
```

0 Python
1 Java
2 JavaScript
3 C
4 C++
5 PHP



```
number1 = int(input('Enter a number:'))
number2 = int(input('Enter a number:'))
print(f'The entered numbers are {number1} and {number2}.')
for i in range(0, 11):
    print(('d x d = d' %(number1, i, number1*i)), ', ', ('d x d =
d' %(number2, i, number2*i )))
```

Enter a number:2
Enter a number:3
The entered numbers are 2 and 3.
2 x 0 = 0 , 3 x 0 = 0
2 x 1 = 2 , 3 x 1 = 3
2 x 2 = 4 , 3 x 2 = 6
2 x 3 = 6 , 3 x 3 = 9
2 x 4 = 8 , 3 x 4 = 12
2 x 5 = 10 , 3 x 5 = 15
2 x 6 = 12 , 3 x 6 = 18
2 x 7 = 14 , 3 x 7 = 21
2 x 8 = 16 , 3 x 8 = 24
2 x 9 = 18 , 3 x 9 = 27
2 x 10 = 20 , 3 x 10 = 30

Addition and average calculation in for loop



```
# Take a list
nlls = [0.577, 2.718, 3.14, 1.618, 1729, 6, 37]
# Write a for loop for addition
count = 0
for i in nlls:
    count+=i
print('The total value of the numbers in the list is', count)
# Calculate the average using len() function
print('The avearge value of the numbers in the list is',
count/len(nlls))
```

The total value of the numbers in the list is 1780.053
The avearge value of the numbers in the list is 254.29328571428573

for-else statement

```
for i in range(1,6):
    print(i, end=", ")
else:
    print()
    print('These are numbers from 1 to 5.')
```

1, 2, 3, 4, 5,
These are numbers from 1 to 5.

nested for loop

```
num = int(input('Enter a number:'))
print(f'The entered the number is {num}.')
i, j = 0, 0
for i in range(0, num):
    print()
    for j in range(0, i+1):
        print('+', end='')
```

Enter a number:2
The entered the number is 2.

+
++

continue in for loop

```
# Take a list
nlis = [1,2,4,5,6,7,8,9,10,11,12,13,14]
for i in nlis:
    if i == 5:
        continue
    print(i)
```

1
2
4
6
7
8
9
10
11
12
13
14

'\nYou see that the output includes the numbers without 5. \nThe continue function jumps when it meets with the reference.\n'

break in for loop

```
# Take a list
nlis = [1,2,4,5,6,7,8,9,10,11,12,13,14]
for i in nlis:
    if i == 5:
        break
    print(i)

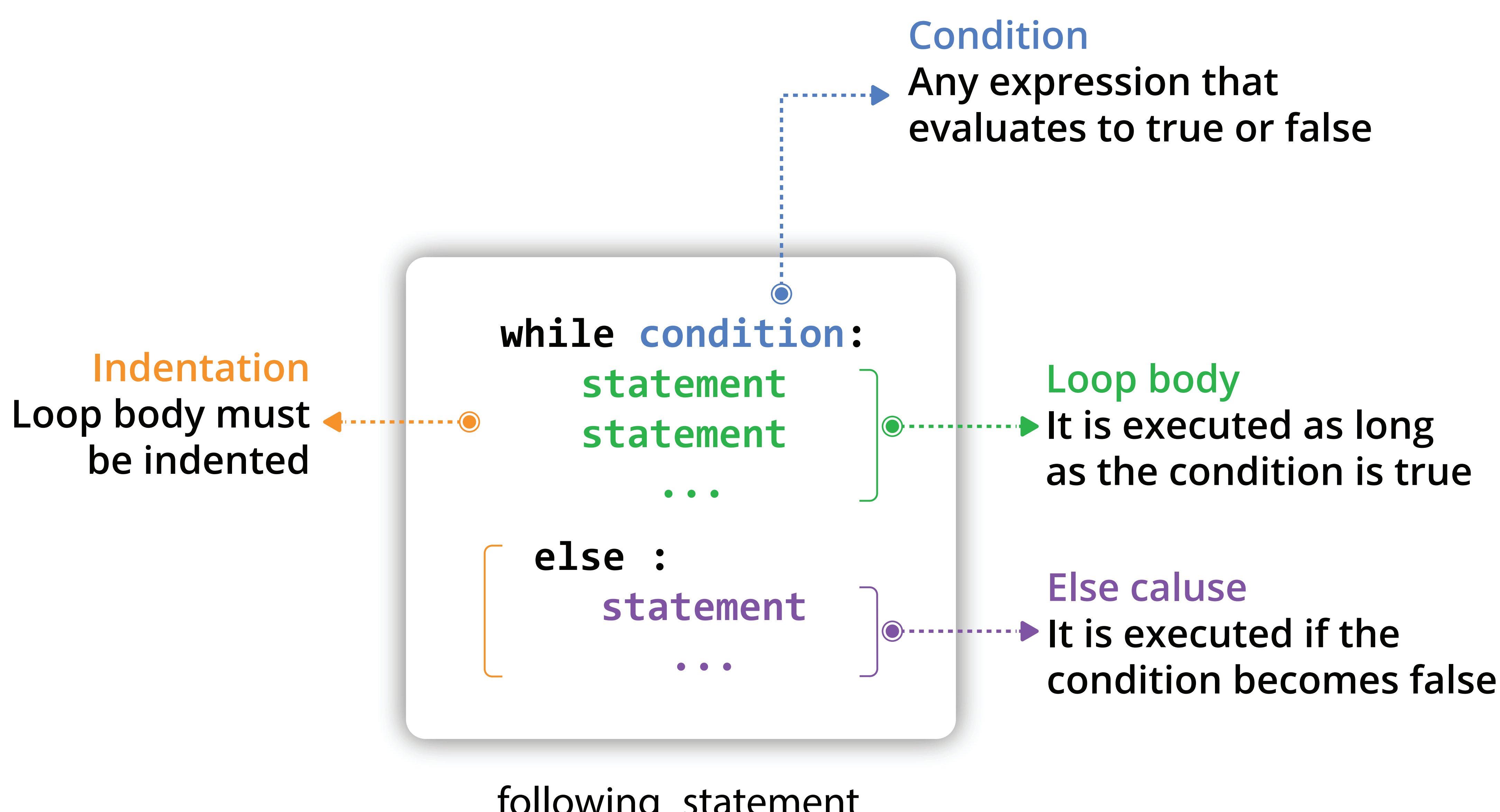
"""
You see that the output includes the numbers before 5.
The break functi on terminate the loop when it meets with the ref-
erence.
"""
```

1
2
4

'\nYou see that the output includes the numbers before 5. \nThe break functi on terminate the loop when it meets with the reference.\n'

while loop

The while loop exists as a tool for repeated execution based on a condition. The code block will keep being executed until the given logical condition returns a False boolean value.



```
# Take an example
i = 22
while i<27:
    print(i)
    i+=1
```

```
22
23
24
25
26
```

```
#Take an example
i = 22
while i>=17:
    print(i)
    i-=1
```

```
22
21
20
19
18
17
```

```
# Take an example
years = [2005, 2006, 2007, 2008, 2009, 2010]
index = 0
year = years[0]
while year !=2008:
    print(year)
    index+=1
    year = years[index]
print('It gives us only', index, 'repetititons to get out of loop')
```

```
2005
2006
2007
It gives us only 3 repetititons to get out of loop
```

```
# Print the movie ratings gretater than 6.
movie_rating = [8.0, 7.5, 5.4, 9.1, 6.3, 6.5, 2.1, 4.8, 3.3]
index = 0
rating = movie_rating[0]
while rating>=6:
    print(rating)
    index += 1
    rating = movie_rating[index]
print('There is only', index, 'movie rating, because the loop stops
when it meets with the number lower than 6.')
```

```
8.0
7.5
There is only 2 movie rating, because the loop stops when it meets with the
number lower than 6.
```

```
# Print the movie ratings gretater than 6.
movie_rating = [8.0, 7.5, 5.4, 9.1, 6.3, 6.5, 2.1, 4.8, 3.3]
index = 0
for i in range(len(movie_rating)):
    if movie_rating[i] >= 6:
        index += 1
        print(index, movie_rating[i])
print('There is only', index, 'films gretater than movie rating 6')
```

```
1 8.0
2 7.5
3 9.1
4 6.3
5 6.5
There is only 5 films gretater than movie rating 6
```




```
# Adding the element in a list to a new list
fruits = ['banana', 'apple', 'banana', 'orange', 'kiwi', 'banana',
'Cherry', 'Grapes']
new_fruits = []
index = 0
while fruits[index] == 'banana':
    new_fruits.append(fruits[index])
    index += 1
print(new_fruits)
```

['banana']



```
number1 = int(input('Enter a number:'))
number2 = int(input('Enter a number:'))
print(f'The entered numbers are {number1} and {number2}.')
i = 0
while i<=10:
    print((' %d x %d = %d' %(number1, i, number1*i)), ', ', (' %d x %d = %d' %(number2, i, number2*i )))
    i+=1
```

Enter a number:2
Enter a number:3
The entered numbers are 2 and 3.
2 x 0 = 0 , 3 x 0 = 0
2 x 1 = 2 , 3 x 1 = 3
2 x 2 = 4 , 3 x 2 = 6
2 x 3 = 6 , 3 x 3 = 9
2 x 4 = 8 , 3 x 4 = 12
2 x 5 = 10 , 3 x 5 = 15
2 x 6 = 12 , 3 x 6 = 18
2 x 7 = 14 , 3 x 7 = 21
2 x 8 = 16 , 3 x 8 = 24
2 x 9 = 18 , 3 x 9 = 27
2 x 10 = 20 , 3 x 10 = 30

while-else statement



```
index = 0
while index <=5:
    print(index, end=' ')
    index += 1
else:
    print('It gives us the numbers between 0 and 5.')
```

0 1 2 3 4 5 It gives us the numbers between 0 and 5.

continue in while loop



```
i = 0
while i<=5:
    print(i)
    i+=1
    if i == 3:
        continue
```

0
1
2
3
4
5

break in while loop



```
i = 0
while i<=5:
    print(i)
    i+=1
    if i == 3:
        break
```

0
1
2