Python if else elif Statement

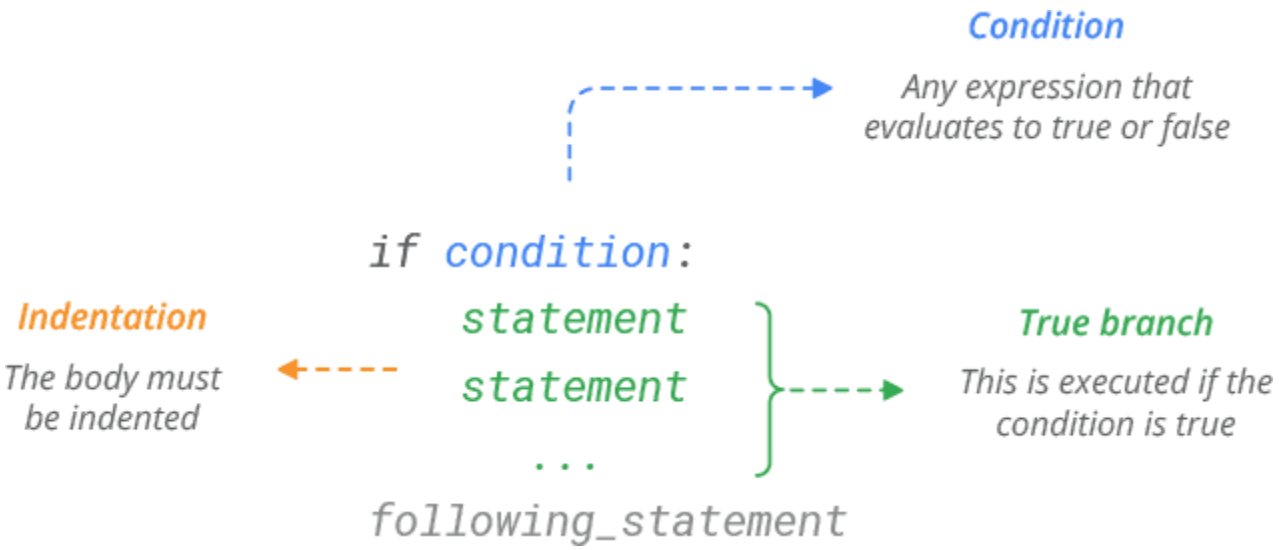
Often you need to execute some statements, only when certain condition holds. You can use following conditional statements in your code to do this.

- if Statement: use it to execute a block of code, if a specified condition is true
- else Statement: use it to execute a block of code, if the same condition is false
- elif (else if) Statement: use it to specify a new condition to test, if the first condition is false

The if Statement

Use `if` statement to execute a block of Python code, if the condition is true.

Syntax



Basic Example

```
x, y = 7, 5
if x > y:
        print('x is greater')
```

```
# Prints x is greater
```

Likewise, you can use following comparison operators to compare two values:

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

More Examples

In Python, any non-zero value or nonempty container is considered TRUE, whereas Zero, None, and empty container is considered FALSE. That's why all the below `if` statements are valid.

```
# any non-zero value
if -3:
    print('True')
# Prints True


# mathematical expression
x, y = 7, 5
if x + y:
    print('True')
# Prints True


# nonempty container
L = ['red','green']
if L:
    print('True')
# Prints True
```

Significance of Indentation

Indentation has a special significance in Python. It is used to define a block of code (often referred to as, a suite). Contiguous statements that are indented to the same level are considered as part of the same block.

`if` statement without indentation raises syntax error.

```
x, y = 7, 5
if x > y:
print('x is greater')
# Triggers SyntaxError: expected an indented block
```

Nested if Statement

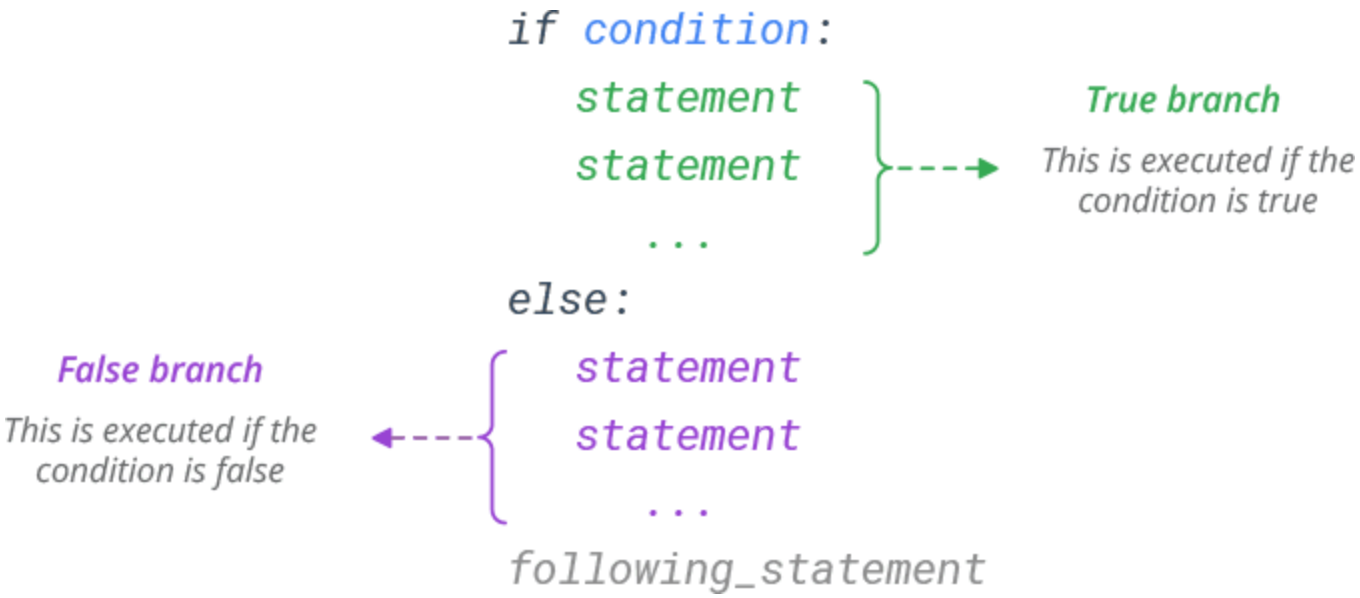
You can nest statements within a code block to begin a new code block, as long as they follow their respective indentations.

```
x, y, z = 7, 4, 2
if x > y:
    print("x is greater than y")
    if x > z:
        print("x is greater than y and z")


# Prints x is greater than y
# Prints x is greater than y and z
```

The else Statement

Use `else` statement to execute a block of Python code, if the condition is false.

Syntax



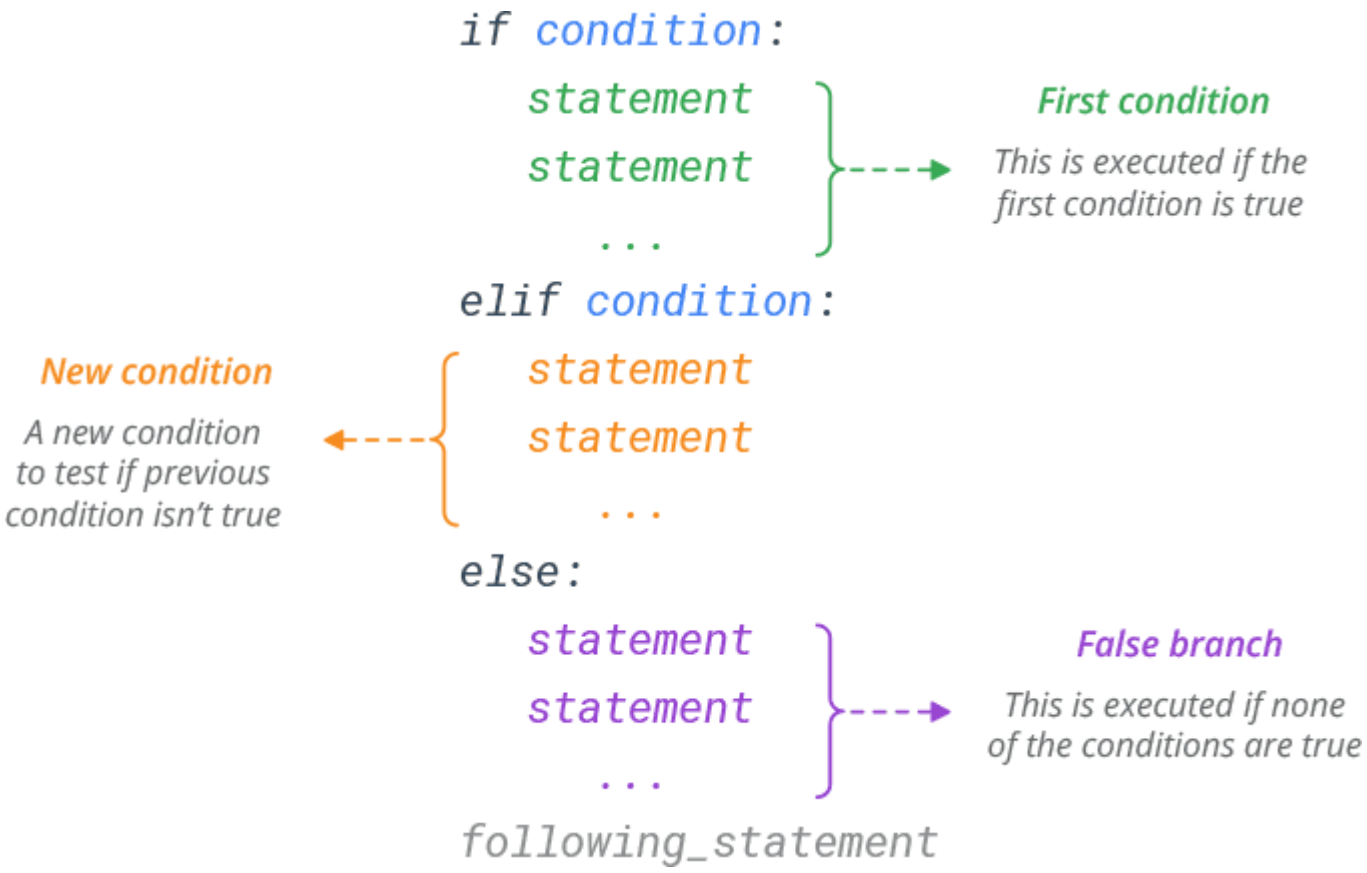
Basic Example

```
x, y = 7, 5
if x < y:
   print('y is greater')
else:
   print('x is greater')

# Prints x is greater
```

The elif (else if) Statement

Use `elif` statement to specify a new condition to test, if the first condition is false.

Syntax



Basic Example

```
x, y = 5, 5
if x > y:
   print('x is greater')
elif x < y:
   print('y is greater')
else:
   print('x and y are equal')

# Prints x and y are equal
```

Substitute for Switch Case

Unlike other programming languages, Python does not have a ‘switch‘ statement. You can use `if…elif…elif` sequence as a substitute.

```
if choice == 1:
            print('case 1')
elif choice == 2:
            print('case 2')
elif choice == 3:
            print('case 3')
elif choice == 4:
            print('case 4')
else:
            print('default case')
```

Multiple Conditions

To join two or more conditions into a single if statement, use logical operators viz. and, or and not.

`and` expression is True, if all the conditions are true.

```
x, y, z = 7, 4, 2
if x > y and x > z:
  print('x is greater')

# Prints x is greater
```

`or` expression is True, if at least one of the conditions is True.

```
x, y, z = 7, 4, 9
if x > y or x > z:
  print('x is greater than y or z')

# Prints x is greater than y or z
```

`not` expression is True, if the condition is false.

```
x, y = 7, 5
if not x < y:
  print('x is greater')

# Prints x is greater
```

One Line if Statement

Python allows us to write an entire if statement on one line.

```
# Short Hand If - single statement
x, y = 7, 5
if x > y: print('x is greater')

# Prints x is greater
```

You can even keep several lines of code on just one line, simply by separating them with a semicolon `;` .

```
# Short Hand If - multiple statements
```

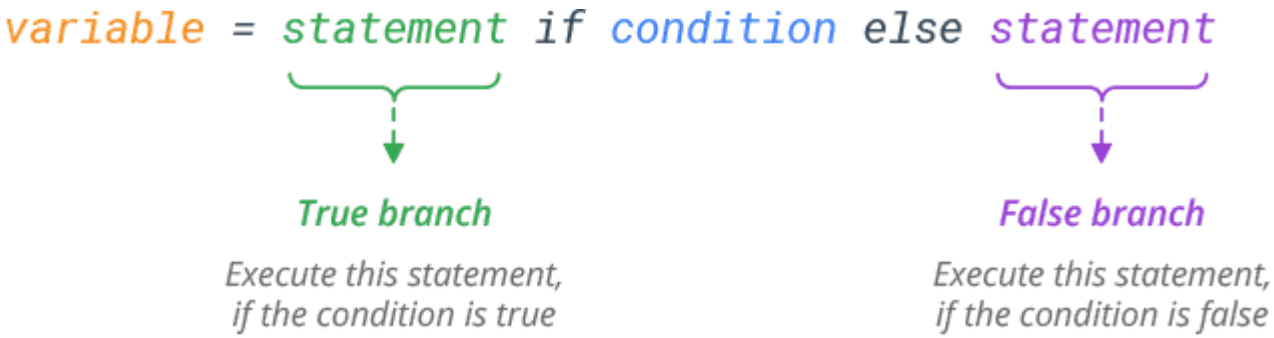
```
x, y = 7, 5

if x > y: print('x is greater'); print('y is smaller'); print('x and y are not equal')


# Prints x is greater

# Prints y is smaller
# Prints x and y are not equal
```

Conditional Expressions (ternary operator)

Conditional expression (sometimes referred to as 'ternary operator') allows us to select one of two statements depending on the specified condition.

The syntax of the conditional expression is :

Syntax



Examples

```
x, y = 7, 5

print('x is greater') if x > y else print('y is greater')


# Prints x is greater
```

You can also use it to select variable assignment.

```
x, y = 7, 5

max = x if x > y else y

print(max)
```

```
# Prints 7
```

Check If Item Present in a Sequence

The `in` operator is used to check if a value is present in a sequence (list, tuple, string etc.).

```
# list
L = ['red', 'green', 'blue']
if 'red' in L:
    print('yes')
# Prints yes


# tuple
T = ('red', 'green', 'blue')
if 'red' in T:
    print('yes')
# Prints yes


# string
S = 'Hello, World!'
if 'Hello' in S:
    print('Yes')
# Prints yes
```