List Comprehensions

## Python List Comprehension Syntax

The syntax of the List Comprehensions in [Python](https://www.tutorialgateway.org/python-tutorial/) Programming Language is as shown below:

**[output\_expression for item in List]**

**[output\_expression for item in List if condition]**

**[output\_expression If Else conditions for item in List]**

## Python List Comprehensions Examples

The following list of examples helps us to learn the Python List Comprehensions.

### Simple Python List Comprehensions Examples

**numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
   
new\_list = []  
   
for num in numbers:  
    new\_list.append(num)  
   
print(new\_list)  
   
# List Comprehension Example  
print("===================")  
  
my\_list = [num for num in numbers]  
print(my\_list)**

**numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
   
new\_list = []  
   
for num in numbers:  
    new\_list.append(num \* 2)  
   
print(new\_list)  
   
# List Comprehension Example  
print("===================")  
   
my\_list = [num \* 2 for num in numbers]  
print(my\_list)**

**numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
   
double\_list = [num \* 2 for num in numbers]  
print(double\_list)  
   
triple\_list = [num \* 3 for num in numbers]  
print(triple\_list)  
   
square\_list = [num \*\* 2 for num in numbers]  
print(square\_list)**

**numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
   
new\_list1 = []  
new\_list2 = []  
new\_list3 = []  
  
for num in numbers:  
    new\_list1.append(num \* 2)  
    new\_list2.append(num \* 3)  
    new\_list3.append(num \*\* 2)  
   
print(new\_list1)  
print(new\_list2)  
print(new\_list3)  
   
# List Comprehension Example  
print("===================")  
double\_list = [num \* 2 for num in numbers]  
print(double\_list)  
   
triple\_list = [num \* 3 for num in numbers]  
print(triple\_list)  
   
square\_list = [num \*\* 2 for num in numbers]  
print(square\_list)**

**numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
   
double\_list = [num \* num for num in numbers]  
print(double\_list)  
   
triple\_list = [num \* num + 1 for num in numbers]  
print(triple\_list)  
   
square\_list = [num \* (num + 2) for num in numbers]  
print(square\_list)  
   
s\_list = [num \* (num + 4) for num in numbers]  
print(s\_list)**

**numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
   
new\_list = []  
   
for num in numbers:  
    if num % 2 == 0:  
        new\_list.append(num)  
   
print(new\_list)  
   
# List Comprehension Example  
print("===================")  
   
my\_list = [num for num in numbers if num % 2 == 0]  
print(my\_list)**

**Python List Comprehension Multiple Conditions**

**new\_list = []  
   
for num in range(1, 150):  
    if num % 2 == 0 and num % 5 == 0:  
        new\_list.append(num)  
   
print(new\_list)  
   
# List Comprehension Example  
print("===================")  
   
my\_list = [num for num in range(1, 150) if num % 2 == 0 if num % 5 == 0]  
print(my\_list)**

### **Python List Comprehension If Else**

**numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
 print(numbers)  
   
my\_list = ["Even" if num % 2 == 0 else "Odd" for num in numbers]  
   
print(my\_list)**

### **Python List Comprehensions Nested For Loop**

**numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
   
my\_list = [[i \* j for j in range(1, 11)] for i in range(2, 4)]  
   
print(my\_list)  
   
# Same using For loop  
print("------------------------------")  
for i in range(2, 4):  
    for j in range(1, 11):  
        print(f"{i} \* {j} = {i \* j}")**

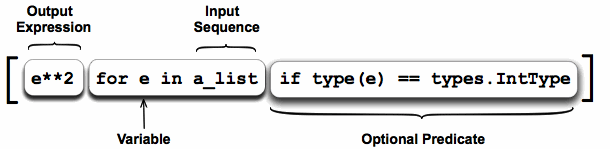
### **Python List Comprehension String List**

**fruits = ['ApplE', 'OraNGe', 'GrAPe', 'BaNAna']  
print(fruits)  
   
l\_list = [a.lower() for a in fruits]  
print(l\_list)  
   
u\_list = [a.upper() for a in fruits]  
print(u\_list)**

**fruits = ['ApplE', 'OraNGeS', 'GrAPe', 'BaNAna']  
print(fruits)  
   
swap\_list = [a.swapcase() for a in fruits]  
print(swap\_list)  
   
len\_list = [len(a) for a in fruits]  
print(len\_list)  
   
first\_letters = [item[0] for item in fruits]  
print(first\_letters)**

**a\_list = [1, ‘4’, 9, ‘a’, 0, 4]**

**squared\_ints = [ e\*\*2 for e in a\_list if type(e) == types.IntType ]**

****

List comprehensions provide a concise way to create lists. Common applications are to make new lists where each element is the result of some operations applied to each member of another sequence or iterable, or to create a subsequence of those elements that satisfy a certain condition.

For example, assume we want to create a list of squares, like:

>>>

**>>>** squares = []

**>>> for** x **in** range(10):

**...**  squares.append(x\*\*2)

**...**

**>>>** squares

[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]

squares = [x\*\*2 **for** x **in** range(10)]

x = [i **for** i **in** range(10)]

**print** x

# This will give the output:

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

Multiplying parts of a list.

Multiply every part **of** a list **by** three **and** assign it to a **new** list.

list1 = [3,4,5]

multiplied = [item\*3 **for** item **in** list1]

**print** multiplied

[9,12,15]

>>> squares = [i \* i for i in range(10)]

>>> squares

[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]

Rather than creating an empty list and adding each element to the end, you simply define the list and its contents at the same time by following this format:

>>>

new\_list = [expression for member in iterable]

Every list comprehension in Python includes three elements:

1. **expression** is the member itself, a call to a method, or any other valid expression that returns a value. In the example above, the expression i \* i is the square of the member value.
2. **member** is the object or value in the list or iterable. In the example above, the member value is i.
3. **iterable** is a list, [set](https://realpython.com/python-sets/), sequence, [generator](https://realpython.com/introduction-to-python-generators/), or any other object that can return its elements one at a time. In the example above, the iterable is range(10).

S = [x\*\*2 for x in range(10)]

V = [2\*\*i for i in range(13)]

M = [x for x in S if x % 2 == 0]

**for n in numbers:**

**if n%2==0:**

**new\_list.append(n\*\*2)**

**# Print `new\_list`**

**print(new\_list)**

# Create `new\_list`

new\_list = [n\*\*2 for n in numbers if n%2==0]

# Print `new\_list`

print(new\_list)