



Python: Replace Item in List (6 Different Ways)

October 19, 2021

Python Replace Item in List

In this tutorial, you'll learn how to use Python to replace an item or items in a list. You'll learn how to replace an item at a particular index, how to replace a particular value, how to replace multiple values, and how to replace multiple values with multiple values.

Being able to work with lists is an important skill for any Python developer, given how prevalent and understandable these Python data structures are.

By the end of this tutorial, you'll have learned:

- How to use list assignment to replace an item in a Python list
- How to use for loops and while loops to replace an item in a Python list
- How to replace multiple items in a list

Table of Contents



Replace an Item in a Python List at a Particular Index

[Python lists](#) are ordered, meaning that we can access (and modify) items when we know their index position. Python list indices start at 0 and go all [the way to the length of the list](#) minus 1.

You can also access items from their negative index. The negative index begins at `-1` for the last item and goes from there. To learn more about Python list indexing, check out [my in-depth overview here](#).

PYTHON LIST INDEXING

['DATAGY', 1, [3, 4, 5], 14.3, 32, 3]

Positive index	0	1	2	3	4	5
Negative index	-6	-5	-4	-3	-2	-1

If we want to replace a [list item at a particular index](#), we can simply directly assign new values to those indices.

Let's take a look at what that looks like:

```
# Replace an item at a particular index in a Python list
a_list = [1, 2, 3, 4, 5, 6, 7, 8, 9]

# Modify first item
a_list[0] = 10
print(a_list)
# Returns: [10, 2, 3, 4, 5, 6, 7, 8, 9]

# Modify last item
a_list[-1] = 99
print(a_list)
# Returns: [10, 2, 3, 4, 5, 6, 7, 8, 99]
```

In the next section, you'll learn how to replace a particular value in a Python list using a for loop.

Want to learn how to use the Python `zip()` function to iterate over two lists? [This tutorial teaches](#) you exactly what the `zip()` function does and shows you some creative ways to use the function.

Replace a Particular Value in a List in Python Using a For Loop

Python lists allow us to easily also modify particular values. One way that we can do this is by using a for loop.

One of the key attributes of [Python lists is that they can contain duplicate values](#). Because of this, we can loop over each item in the list and check its value. If the value is one we want to replace, then we replace it.

Let's see what this looks like. In our list, the word `apple` is misspelled. We want to replace the misspelled version with the corrected version.

```
# Replace a particular item in a Python list
a_list = ['apple', 'orange', 'apple', 'banana', 'grape', 'apple']

for i in range(len(a_list)):
    if a_list[i] == 'apple':
```

```
a_list[i] = 'apple'


print(a_list)

# Returns: ['apple', 'orange', 'apple', 'banana', 'grape', 'apple']
```

Let's take a look at what we've done here:

1. We loop over each index in the list
2. If the [index position of the list](#) is equal to the item we want to replace, we re-assign its value

In the next section, you'll learn how to turn this for loop into a Python list comprehension.

Want to learn more about Python for-loops? Check out [my in-depth tutorial](#) that takes your from beginner to advanced for-loops user! Want to watch a video instead? Check out my [YouTube tutorial here](#) .

Replace a Particular Value in a List in Python Using a List Comprehension

One of the key attributes of Python list comprehensions is that we can often turn for loops into much shorter comprehensions. Let's see how we can use a list comprehension in Python to replace an item in a list.

We'll use the same example we used in the for loop, to help demonstrate how elegant a Python list comprehension can be:

```
# Replace a particular item in a Python list using a list comprehension
a_list = ['apple', 'orange', 'apple', 'banana', 'grape', 'apple']


a_list = ['apple' if item == 'apple' else item for item in a_list]
print(a_list)

# Returns: ['apple', 'orange', 'apple', 'banana', 'grape', 'apple']
```

We can see here that there are two main benefits of list comprehensions compared to for loops:

1. We don't need to initialize an empty list
2. The comprehension reads in a relatively plain English

In the next section, you'll learn how to change all values in a list using a formula.

Want to learn more about Python list comprehensions? Check out [this in-depth tutorial](#) that covers off everything you need to know, with hands-on examples. More of a visual learner, [check out my YouTube tutorial here](#) .

Change All Values in a List in Python Using a Function

Neither of the approaches above are immediately clear as to what they are doing. Because of this, developing a formula to do this is a helpful approach to help readers of your code understand what it is you're doing.

Let's take a look at how we can use the list comprehension approach and turn it into a formula:

```
# Replace a particular item in a Python list using a function
a_list = ['apple', 'orange', 'apple', 'banana', 'grape', 'apple']

def replace_values(list_to_replace, item_to_replace, item_to_replace_with):
    return [item_to_replace_with if item == item_to_replace else item for item in
            list_to_replace]

replaced_list = replace_values(a_list, 'apple', 'apple')
print(replaced_list)

# Returns: ['apple', 'orange', 'apple', 'banana', 'grape', 'apple']
```

Here, we simply need to pass in the list, the item we want to replace, and the item we want to replace it with. The function name makes it easy to understand what we're doing, guiding our readers to better understand our actions.

In the next section, you'll learn how to replace multiple values in a Python list.

Replace Multiple Values in a Python List

There may be many times when you want to replace not just a single item, but multiple items. This can be done quite simply using the for loop method shown earlier.

Let's take a look at an example where we want to replace all known typos in a list with the word `typo`.

```
# Replace multiple items in a Python list with the same value
a_list = ['apple', 'orange', 'apple', 'banana', 'grape', 'apple']
for i in range(len(a_list)):
    if a_list[i] in ['apple', 'orange']:
        a_list[i] = 'typo'

print(a_list)

# Returns: ['typo', 'typo', 'typo', 'banana', 'grape', 'typo']
```

Similar to the for loop method shown earlier, we check whether or not an item is a typo or not and replace its value.

In the next section, you'll learn how to replace multiple values in a Python list with different values.

Replace Multiple Values with Multiple Values in a Python List

The approach above is helpful if we want to replace multiple values with the same value. There may be times where you want to replace multiple values with *different* values.

Looking again at our example, we may want to replace all instances of our typos with their corrected spelling. We can again use a for loop to illustrate how to do this.

Instead of using a single `if` statement, we'll nest in some `elif` statements that check for a value's value before replacing.

```
# Replace multiple items in a Python list with the different values
a_list = ['apple', 'orange', 'apple', 'banana', 'grape', 'apple']
for i in range(len(a_list)):
    if a_list[i] == 'apple':
        a_list[i] = 'apple'
    elif a_list[i] == 'orange':
```

```
a_list[i] = 'orange'

print(a_list)
# Returns: ['apple', 'orange', 'apple', 'banana', 'grape', 'apple']
```

Need to check if a key exists in a Python dictionary? Check [out this tutorial](#), which teaches you five different ways of seeing if a key exists in a Python dictionary, including how to return a default value.

Conclusion

In this tutorial, you learned how to use Python to replace items in a list. You learned how to use Python to replace an item at a particular index, how to replace a particular value, how to modify all values in a list, and how to replace multiple values in a list.

To learn more about Python list indexing, check out the [official documentation here](#).

Additional Resources

To learn more about related topics, check out the tutorials below:

- [Pandas Replace: Replace Values in Pandas Dataframe](#)
- [Python: Select Random Element from a List](#)
- [Python: Combine Lists – Merge Lists \(8 Ways\)](#)
- [Python: Count Number of Occurrences in List \(6 Ways\)](#)

Tags:

PYTHON

PYTHON LISTS

PREVIOUS

[Pandas: Convert Column Values to Strings](#)

NEXT

[Pandas: Iterate over a Pandas Dataframe Rows](#)

2 thoughts on “Python: Replace Item in List (6 Different Ways)”