GeeksforGeeks

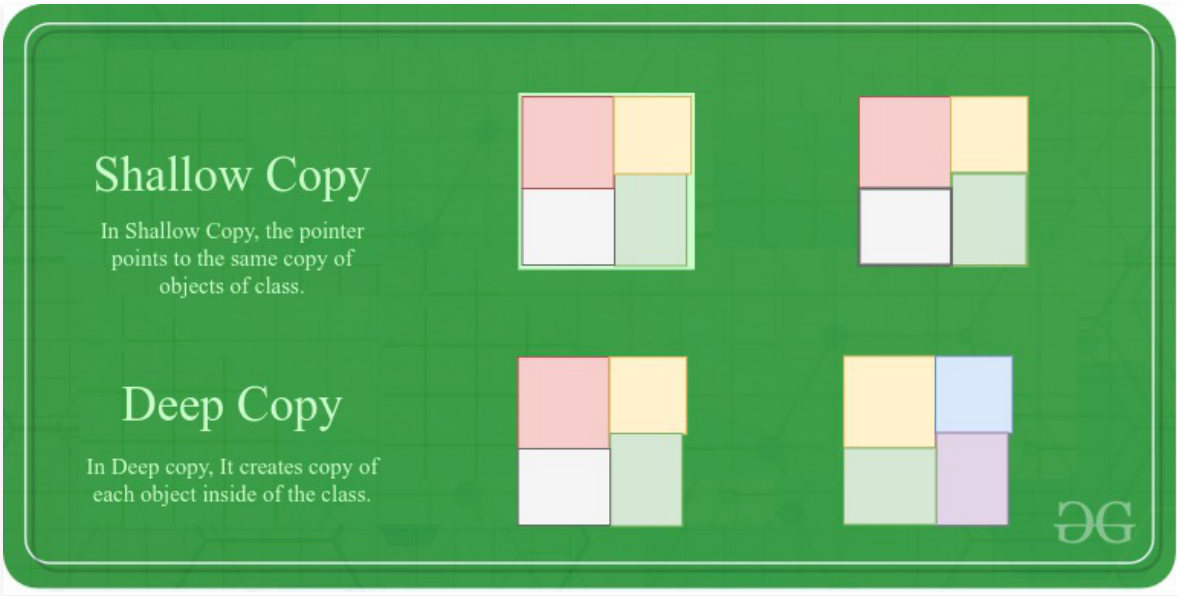
Related Articles

Difference between Shallow and Deep copy of a class

Difficulty Level : Medium ● Last Updated : 16 Nov, 2020

Shallow Copy: Shallow repetition is quicker. However, it's "lazy" it handles pointers and references. Rather than creating a contemporary copy of the particular knowledge the pointer points to, it simply copies over the pointer price. So, each the first and therefore the copy can have pointers that reference constant underlying knowledge.

Deep Copy: Deep repetition truly clones the underlying data. It is not shared between the first and therefore the copy.



Below is the tabular Difference between the Shallow Copy and Deep Copy:

Shallow Copy	Deep Copy
Shallow Copy stores the references of objects to the original memory address.	Deep copy stores copies of the object's value.

Shallow Copy	Deep Copy
Shallow Copy reflects changes made to the new/copied object in the original object.	Deep copy doesn't reflect changes made to the new/copied object in the original object.
Shallow Copy stores the copy of the original object and points the references to the objects.	Deep copy stores the copy of the original object and recursively copies the objects as well.
Shallow copy is faster.	Deep copy is comparatively slower.

Below is the program to explain the shallow and deep copy of the class.

Python3

```
# Python3 implementation of the Deep
# copy and Shallow Copy
from copy import copy, deepcopy

# Class of Car
class Car:
  def __init__(self, name, colors):

      self.name = name
      self.colors = colors

honda = Car("Honda", ["Red", "Blue"])

# Deepcopy of Honda
deepcopy_honda = deepcopy(honda)
deepcopy_honda.colors.append("Green")
print(deepcopy_honda.colors, \
       honda.colors)

# Shallow Copy of Honda
copy_honda = copy(honda)

copy_honda.colors.append("Green")
```

```
print(copy_honda.colors, \
      honda.colors)
```

Output:

```
['Red', 'Blue', 'Green'] ['Red', 'Blue']
['Red', 'Blue', 'Green'] ['Red', 'Blue', 'Green']
```