Sets in Python

A Set is an unordered collection data type that is iterable, mutable, and has no duplicate elements. Python's set class represents the mathematical notion of a set. The major advantage of using a set, as opposed to a list, is that it has a highly optimized method for checking whether a specific element is contained in the set. This is based on a data structure known as a hash table.

Frozen Sets Frozen sets are immutable objects that only support methods and operators that produce a result without a?ecting the frozen set or sets to which they are applied.

```
# Python program to demonstrate differences
# between normal and frozen set

# Same as {"a", "b","c"}
normal_set = set(["a", "b","c"])

# Adding an element to normal set is fine
normal_set.add("d")

print("Normal Set")
print(normal_set)

# A frozen set
frozen_set = frozenset(["e", "f", "g"])

print("Frozen Set")
print(frozen_set)

# Uncommenting below line would cause error as
# we are trying to add element to a frozen set
# frozen_set.add("h")
```

Run on IDE

Output:

```
Normal Set
set(['a', 'c', 'b', 'd'])
Frozen Set
frozenset(['e', 'g', 'f'])
```

Methods for Sets

1. add(x) Method: Adds the item x to set if it is not already present in the set.


```
people = {"Jay", "Idrish", "Archil"}
people.add("Daxit")
```

-> This will add Daxit in people set.

2. **union(s) Method**: Returns a union of two set.Using the '|' operator between 2 sets is the same as writing set1.union(set2)

```
people = {"Jay", "Idrish", "Archil"}
vampires = {"Karan", "Arjun"}
population = people.union(vampires)
```

OR

```
population = people|vampires
```

-> Set population set will have components of both people and vampire

3. **intersect(s) Method:** Returns an intersection of two sets.The '&' operator comes can also be used in this case.

```
victims = people.intersection(vampires)
```

-> Set victims will contain the common element of people and vampire

4. difference(s) Method: Returns a set containing all the elements of invoking set but not of the second set. We can use '-' operator here.

```
safe = people.difference(vampires)
```

OR

```
safe = people - vampires
```

-> Set safe will have all the elements that are in people but not vampire

5. clear() Method: Empties the whole set.

```
victims.clear()
```

-> Clears victim set

However there are two major pitfalls in Python sets:

1. The set doesn't maintain elements in any particular order.
2. Only instances of immutable types can be added to a Python set.

Operators for Sets

Sets and frozen sets support the following operators:

key in s # containment check

key not in s # non-containment check

s1 == s2 # s1 is equivalent to s2

s1 != s2 # s1 is not equivalent to s2

s1 <= s2 # s1is subset of s2 s1 < s2 # s1 is proper subset of s2 s1 >= s2 # s1is superset of s2

s1 > s2 # s1 is proper superset of s2

s1 | s2 # the union of s1 and s2

s1 & s2 # the intersection of s1 and s2

s1 – s2 # the set of elements in s1 but not s2

s1 ^ s2 # the set of elements in precisely one of s1 or s2

Code Snippet to illustrate all Set operations in Python

```
# Python program to demonstrate working# of
# Set in Python

# Creating two sets
set1 = set()
set2 = set()

# Adding elements to set1
for i in range(1, 6):
    set1.add(i)

# Adding elements to set2
for i in range(3, 8):
    set2.add(i)

print("Set1 = ", set1)
print("Set2 = ", set2)
print("\n")

# Union of set1 and set2
set3 = set1 | set2# set1.union(set2)
print("Union of Set1 & Set2: Set3 = ", set3)
```

```
# Intersection of set1 and set2
set4 = set1 & set2# set1.intersection(set2)
print("Intersection of Set1 & Set2: Set4 = ", set4)
print("\n")

# Checking relation between set3 and set4
if set3 > set4: # set3.issuperset(set4)
    print("Set3 is superset of Set4")
elif set3 < set4: # set3.issubset(set4)
    print("Set3 is subset of Set4")
else : # set3 == set4
    print("Set3 is same as Set4")

# displaying relation between set4 and set3
if set4 < set3: # set4.issubset(set3)
    print("Set4 is subset of Set3")
    print("\n")

# difference between set3 and set4
set5 = set3 - set4
print("Elements in Set3 and not in Set4: Set5 = ", set5)
print("\n")

# checkv if set4 and set5 are disjoint sets
if set4.isdisjoint(set5):
    print("Set4 and Set5 have nothing in common\n")

# Removing all the values of set5
set5.clear()

print("After applying clear on sets Set5: ")
print("Set5 = ", set5)
```

Run on IDE

Output:

```
('Set1 = ', set([1, 2, 3, 4, 5]))
('Set2 = ', set([3, 4, 5, 6, 7]))


('Union of Set1 & Set2: Set3 = ', set([1, 2, 3, 4, 5, 6, 7]))
('Intersection of Set1 & Set2: Set4 = ', set([3, 4, 5]))


Set3 is superset of Set4
Set4 is subset of Set3


('Elements in Set3 and not in Set4: Set5 = ', set([1, 2, 6, 7]))


Set4 and Set5 have nothing in common

After applying clear on sets Set5:
('Set5 = ', set([]))
```

Recent articles on Python Set.

This article is contributed by **Jay Patel.** If you like GeeksforGeeks and would like to contribute, you can also write an article and mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above

Article Tags : Python python-set

Login to Improve this Article

Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.

Recommended Posts:

Set update() in Python to do union of n arrays
Array in Python | Set 1 (Introduction and Functions)
Dictionary Methods in Python | Set 1 (cmp(), len(), items()…)
Tuples in Python
Python List Comprehension and Slicing
Function Decorators in Python | Set 1 (Introduction)
Python | Count the Number of matching characters in a pair of string
Python | How and where to apply Feature Scaling?
Testing in Python using doctest module
Python | Simple GUI calculator using Tkinter

(Login to Rate)

2.3 Average Difficulty : **2.3/5.0**
Based on **10** vote(s)

Basic Easy Medium Hard Expert

Add to TODO List
Mark as DONE

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments Share this post!

A computer science portal for geeks

710-B, Advant Navis Business Park,
Sector-142, Noida, Uttar Pradesh - 201305
feedback@geeksforgeeks.org