PES UNIVERSITY

Introduction to Sets

- At the end of this class, students will be able to-
  - Use the variable type – Set
  - Create and modify Set using Set built – in functions

Set

- A set is a data structure with zero or more elements with the following attributes.
- Elements are unique – does not support repeated elements
- set is not ordered – we cannot assume the order of elements in a set.
- set is an iterable – eager and not lazy
- we cannot index on a set. The set is iterable, but is not a sequence.
- We can check for membership using the in operator. This would be faster in case of a set compared to a list, a tuple or a string.

- Sets support many mathematical operations on sets. ◦ Membership : in
- union : |
- intersection : &
- set diference : -
- symmetric diference : ^
- equality and inequality : = !=
- subset and superset : < <= > >=
- set constructor { … }
- To create an empty set, we must use the set constructor set() and not { }. The latter would become a dict.
- Set is mutable but contains immutable objects.

Use of Sets

a) deduplication : removal of repeated elements

b) finding unique elements

c) comparing two iterables for common elements or difference

d) No nested set is allowed.

Set Operations

- s1 = {1, 2, 3, 4, 5}
- s2 = {1, 3, 5, 7, 9}

union p

- print(s1 | s2) # {1, 2, 3, 4, 5, 7, 9}

intersection

- print(s1 & s2) # {1, 3, 5}

set difference

- print(s1 - s2) # {2, 4}

symmetric difference

print(s1 ^ s2) # {2, 4, 7, 9}

Set Operations

- empty set creation

```
s3 = set()
```

- Finding unique elements in a string

```
s5 = set("mississippi")
print(s5) # {'s', 'i', 'm', 'p'}
```

- Finding unique words in a string str1; words separated by white space

```
str1={"hello","bye","hi","world","bye","hi"}
a=list(set(str1))
['hello', 'hi', 'bye', 'world']
```

Write a Python program to add and update member(s) in a set.

```
a=set()# creating an empty set
print(a)
a.add(10)# Single element only can be added with add()
c=20,30
a.add(c)#Single element:tuple
print(a)
a.update([40,50])# union of [40,50] with the set
print(a)
```

Write a Python program to create a set.

```
a=set()
for i in range(0,6):
        n=input("enter element")
        a.add(n)
print(a)
b=set()
for i in range(0,6):
        b.add(input("enter element"))
print(b)
```

Write a Python program to iteration over sets and print max and min value.

```
a={1,2,3,4,5,6}
for i in a:
        print(i)
print(max(a))
print(min(a))
```

Write a Python program to remove item(s) from set

```
a={1,2,3,4}
print(a.pop())
#Remove and return an arbitrary set element.
#Raises KeyError if the set is empty.
a.remove(5)
#Remove an element from a set; it must be a member.
#If the element is not a member, raise a KeyError.
a.discard(5)
#Remove an element from a set if it is a member.
#If the element is not a member, do nothing.
```

Write a Python program to test whether every element in s is in t and every element in t is in s.

```
setx = set(["apple", "mango"])
sety = set(["mango", "orange"])
setz = set(["mango"])
issubset = setx <= sety
print(issubset)   #False
issuperset = setx >= sety
print(issuperset)  #False
issubset = setz <= sety
print(issubset)   #True
issuperset = sety >= setz
print(issuperset )      #True
```

Clear()

- Write a Python program to remove all the item(s) from set

The clear() method empties the set:

```
a={"hello","bye","hi"}
print(a)
print(id(a))
a.clear()
print(a)
print(id(a))
```

O/P

```
{'bye', 'hi', 'hello'}
1526421872200
set()
1526421872200
```

- Write a Python program to delete the set

The del keyword will delete the set completely

```
a={"hello","bye","hi"}
print(a)
print(id(a))
del a
print(a)
print(id(a))
```

O/P

{'hi', 'bye', 'hello'}
2479628639816
NameError: name 'a' is not defined

Set Methods

Method	Description
add()	Adds an element to the set
clear()	Removes all the elements from the set
copy()	Returns a copy of the set
difference()	Returns a set containing the difference between two or more sets
difference_update()	Removes the items in this set that are also included in another, specified set
discard()	Remove the specified item
intersection()	Returns a set, that is the intersection of two other sets
intersection_update()	Removes the items in this set that are not present in other, specified set(s)

Set Methods

Method	Description
isdisjoint()	Returns whether two sets have a intersection or not
issubset()	Returns whether another set contains this set or not
issuperset()	Returns whether this set contains another set or not
pop()	Removes the specified element
remove()	Removes the specified element
symmetric_difference()	Returns a set with the symmetric differences of two sets
symmetric_difference_update()	inserts the symmetric differences from this set and another

Built-in Functions with Set

Function	Description
all()	Return True if all elements of the set are true (or if the set is empty).
any()	Return True if any element of the set is true. If the set is empty, return False.
enumerate()	Return an enumerate object. It contains the index and value of all the items of set as a pair.
len()	Return the length (the number of items) in the set.
max()	Return the largest item in the set.
min()	Return the smallest item in the set.
sorted()	Return a new sorted list from elements in the set(does not sort the set itself).
sum()	Return the sum of all elements in the set.

- Python Program to Check Common Letters in Two Input Strings

```
str1=input("enter string one:")
str2=input("enter string two:")
a=list(set(str1)&set(str2))
for i in a:
        print(i)
```

O/P

enter string one: This is python program
enter string two: This is C program
g s o r i a p m T h

Python Program that Displays which Letters are in the First String but not in the Second

```
str1=input("enter string one")
str2=input("enter string two")
a=list(set(str1)-set(str2))
for i in a:
        print(i,end=" ")
```

O/P

enter string one: This is python program
enter string two: This is C program
g s o r i a p m T h

Sieve of Eratosthenes

- generate prime numbers (no division; most efficient algorithm)
- sieve of Eratosthenes
- get a number(say n)
- make a set of numbers from 2 to n - say sieve
- while sieve is not empty
- find the smallest (small)
- print it (that is a prime)
- remove small and its multiples from the sieve

```
n= int(input("enter the number:"))
seive=set(range(2,n+1))
while seive:
        small=min(seive)
        print(small,end=" ")
        seive-=set(range(small,n+1,small))
```

O/P

enter the number:50

2 3 5 7 11 13 17 19 23 29 31 37 41 43 47

Declare a set named vowels containing the strings 'a','e','i','o', and 'u'. Give a program segment that prompts the user for any English word, and displays how many vowels it contains.

```
vowels = set('aeiouAEIOU')
num_vowels = 0
word = input('Enter word: ')
for ch in word:
        if ch in vowels:
                num_vowels = num_vowels + 1
print('There are', num_vowels, 'vowels in the word',
  word)
```

O/P

Enter word: python program
There are 3 vowels in the word python program

- Give a program segment that prompts the user for two English words, and displays which letters of the alphabet are in neither of the two words.

```
alphabet = set([chr(k) for k in range(ord('a'), ord('z')+1)])
word1 = input('Enter a word: ')
word2 = input('Enter another word: ')
word1_charset = set(word1.lower())
word2_charset = set(word2.lower())
nonappearing_chars =  alphabet - (word1_charset | word2_charset)
if len(nonappearing_chars) == 0:
  print('There are no unused letters of the alphabet in the words',word1, 'and', word2)
else:
        nonappearing_chars_list = list(nonappearing_chars)
        nonappearing_chars_list.sort()
        print('The following letters of the alphabet do not appear in either',word1, 'or', word2 + ':')
        print(nonappearing_chars_list)
```

O/P

Enter a word: Python
Enter another word: Java
The following letters of the alphabet do not appear in either Python or Java:
['b', 'c', 'd', 'e', 'f', 'g', 'i', 'k', 'l', 'm', 'q', 'r', 's', 'u', 'w', 'x', 'z']

split()

Syntax: S.split(sep=None, maxsplit=-1) -> list of strings

a="this is python program"

b=a.split()

print(b)

Output:['this', 'is', 'python', 'program']

c=a.split()[0]

print(c)

Output: this

```
a="""hi xyz
hello xyz
bye xyz
"""
b=a.split()
print(b)
```

Output: ['hi', 'xyz', 'hello', 'xyz', 'bye', 'xyz']

```
a="""hi xyz
hello xyz
bye xyz"""
b=a.split("\n")
print(b)
```

Output: ['hi xyz', 'hello xyz', 'bye xyz']

```
a="""hi xyz
hello xyz
bye xyz
"""
b=a.split("\n")
print(b)
```

Output: ['hi xyz', 'hello xyz', 'bye xyz', '']

```
a="""hi xyz hello xyz bye xyz """
b=a.split(maxsplit=2)
print(b)
```

Output: ['hi', 'xyz', 'hello xyz bye xyz ']

```
a="""hi xyz
hello xyz
bye xyz
"""
b=a.split(maxsplit=3)
print(b)
```

Output: ['hi', 'xyz', 'hello', 'xyz\nbye xyz\n']

Consider the following text for the problems below.

Print the count and the Wining team names

Example (Wining) (Losing)

highest_inings="""England Australia 481

England Pakistan 444

SriLanka Netherlands 443

SouthAfrica WestIndies 439

SouthAfrica Australia 438

SouthAfrica India 438”””

Op: count : 3

Teams are {'England', 'SouthAfrica', 'SriLanka'}

Count= 3

```
highest_inings="""England Australia 481
England Pakistan 444
SriLanka Netherlands 443
SouthAfrica WestIndies 439
SouthAfrica Australia 438
SouthAfrica India 438"""
s=highest_inings.split("\n")
print(s)
set1=set()
for i in s:
        set1.add(i.split()[0])
print("Teams are",set1)
print("Count=",len(set1))
```

O/P

['England Australia 481', 'England Pakistan 444', 'SriLanka Netherlands 443', 'SouthAfrica WestIndies 439', 'SouthAfrica Australia 438', 'SouthAfrica India 438']
Teams are {'England', 'SouthAfrica', 'SriLanka'}
Count= 3

Mutable vs. Immutable Set Types in Python

Finally, there are two set types in Python—the mutable **set** type, and the immutable **frozenset** type. Methods add and remove are not allowed on sets of frozenset type. Thus, all the members of a frozenset type are declared when it is defined,

```
>>> apple_colors = frozenset(['red', 'yellow', 'green'])
```

Frozensets are used when an immutable type is desired or needed, such as when used as key values in a given dictionary.

Summary

- A set is a data structure with zero or more elements with the following attributes.
- Elements are unique – does not support repeated elements
- set is not ordered