Python Programming

Set 1

Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)

What?

```
# set: unordered
# (don't preserve insertion order / no values order)
# unique: duplicates are ignored
# items: must be immutable

st = set()
st.add(20)
st.add(10)
st.add(20)
st.add(-2537)
st.add(10)
print(st)   # {10, 20, -2537}
```

Set

```
3   st = {1, 5, 1, 3, 5}
4   print(st)    # {1, 3, 5}
5
6   st = set(['saad', 'most', 'saad'])  # takes iterable
7
8   print('al' in st)        # False
9   for item in st:          # No guarantee on order
10      print(item, end=' ') # most saad
11  print()
12
13  print(list(st))      # ['most', 'saad']
14  print(set({1:10, -2:30}))    # {1, -2}
15
16  print(set('Hey'))        # {'H', 'y', 'e'}
17  print(set(['Hey']))      # {'Hey'}
18  print(set({'Hey'}))      # {'Hey'}
```

Functions

```
st = {(1, 5), (2, 7), (1, 5), (2, 7)}
print(st)    # {(2, 7), (1, 5)}

# TypeError: unhashable type: 'list'
#st = {(1, 5), [2, 7]}

print(len(st))  # 2
print(max(st))  # (2, 7)
print(sorted(st))  #[(1, 5), (2, 7)]

print(sum({1, 1, 1, 1, 2, 2, 2, 2}))    # 3 = 1+2
print(all({1, 2, 'hey'}))       # True
print(all({1, 2, 'hey', ()}))   # False: empty tuple
```

Methods

```
st1 = {1, 3, 5, 7, 8, 10}

st1.add(-20)
st1.remove(10)
#st1.remove(30)  # if not exist, error => KeyError
st1.discard(30)  # if not exist, no problem
print(st1)  # {1, 3, 5, 7, 8, -20}

print(st1.pop())    # remove random element. If empty = error
st1.clear() # remove elements
```

Set comprehension

```
# Set comprehension
line = "I am mostafa saad Ibrahim"
unique_vowels = {i for i in line if i in 'aeiou'}

print(unique_vowels)    # {'o', 'a', 'i'}
```

Time Complexity

See dict -- the implementation is intentionally very similar.

Operation	Average case	Worst Case	notes
x in s	O(1)	O(n)	
Union s|t	O(len(s)+len(t))		
Intersection s&t	O(min(len(s), len(t))	O(len(s) * len(t))	replace "min" with "max" if t is not a set
Multiple intersection s1&s2&..&sn		(n-1)*O(l) where l is max(len(s1),..,len(sn))	
Difference s-t	O(len(s))		
s.difference_update(t)	O(len(t))		
Symmetric Difference s^t	O(len(s))	O(len(s) * len(t))	
s.symmetric_difference_update(t)	O(len(t))	O(len(t) * len(s))	

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."