Lesson-11: Dictionaries & Sets

- Dictionaries and Sets are unordered data structures.
- Dictionaries cannot have unhashable types (list, dictionary and set) as key values
- Sets cannot have unhashable types (list, dictionary and set)
- In place of **list** use **tuple**, in place of **set** use **frozenset**.
- There is not direct replacement for **dictionary**.
- Sets cannot have duplicates

Program: 1

```
dictone = { "subhash": 9845456000, "sharukh": 9999999999, "akshay": 8888888888 }
print(dictone)
print(dictone["subhash"])

dicttwo = { 123:"abc", 456:"def", 789:"ghi" }
print(dicttwo)
print(dicttwo[123])

dictthree = dict( )
print(dictthree)

dictfour = dict( )
dictfour["Hello"] = 123
dictfour["Bye"] = 456
print(dictfour)
print(dictfour["Bye"])

dictfive = { ("Subhash", "Gender"):(34, "Male"), ("Deepika", "Gender"):(32, "Female")}
print(dictfive)
print(dictfive[("Subhash", "Gender")])

l = [ ['apple',180.00],['mango',90.00]]
dictsix = dict(l)
print(dictsix)

dictseven = { "123":123, "345":345, "567":567 }
print(dictseven)
del dictseven["123"]
print(dictseven)
print( len(dictsix) )
print( "123" in dictseven )
```

Output:

```
{'subhash': 9845456000, 'sharukh': 9999999999, 'akshay': 8888888888}
9845456000
{123: 'abc', 456: 'def', 789: 'ghi'}
abc
{ }
{'Hello': 123, 'Bye': 456}
456
{('Subhash', 'Gender'): (34, 'Male'), ('Deepika', 'Gender'): (32, 'Female')}
(34, 'Male')
{'apple': 180.0, 'mango': 90.0}
{'123': 123, '345': 345, '567': 567}
{'345': 345, '567': 567}
2
False
```

Program: 2

```
dicteight = { "Jojo":"illa", "momo":"killa" }
print(dicteight)
dicteight["Jojo"] = "Jilla"
print(dicteight)
```

Output:

```
{'Jojo': 'illa', 'momo': 'killa'}
{'Jojo': 'Jilla', 'momo': 'killa'}
```

Guess The Output:

1.
```
>>> dict = { [1,2,3]: “123” }
        print(dict)
```

2.
```
>>> dict = { "S":1, "U":2, "B":3 }
>>> dict.values( )
```

3.
```
>>> dict = { "S":1, "U":2, "B":3 }
>>> dict.keys( )
```

4.
```
>>> dict = { "S":1, "U":2, "B":3 }
>>> dict.items()
```

5.
```
>>> s = {1: 'S'}
>>> s.pop(1)
>>>s
```

Program: 3

#Temperature Display Program (List Version)

```
terminate = False
daily_temps = [ 68.8, 70.2, 67.2, 71.8, 73.2, 75.6, 74.0]

print("This program will display the average temperature for a given day\n")

while not terminate:
    day = input("Enter 'sun', 'mon', 'tue', 'wed', 'thur', 'fri', or 'sat':")

    if day == 'sun':
        dayname = 'Sunday'
        temp = daily_temps[0]
    elif day == 'mon':
        dayname = 'Monday'
        temp = daily_temps[1]
    elif day == 'tue':
        dayname = 'Tuesday'
        temp = daily_temps[2]
    elif day == 'wed':
        dayname = 'Wednesday'
        temp = daily_temps[3]
    elif day == 'thur':
        dayname = 'Thursday'
        temp = daily_temps[4]
    elif day == 'fri':
        dayname = 'Friday'
        temp = daily_temps[5]
    elif day == 'sat':
        dayname = 'Saturday'
        temp = daily_temps[6]

    print( 'The average temperature for', dayname, 'was', temp, 'degrees\n')

    response = input('Continue with another day? (y/n): ')

    if response == 'n':
        terminate = True
```

Output:

Program: 4

#Temperature Display Program (Dictionary Version)

```
daily_temps = {'sun': 68.8, 'mon':70.2, 'tue':67.2, 'wed':71.8, 'thur':73.2, 'fri':75.6, 'sat':74.0 }

daynames = { 'sun' : 'Sunday', 'mon':'Monday', 'tue':'Tuesday', 'wed':'Wednesday', 'thur':'Thursday',
'fri':'Friday','sat': 'Saturday' }

print( "This program will display the average temperature for a given day\n")

day = input( "Enter 'sun', 'mon', 'tue', 'wed', 'thur', 'fri', or 'sat': ")
print( "The average temperature for", daynames[day], "was", daily_temps[day], "degrees")
```

Output:

Program: 5

```
d = { }
num_elements = int(input("How many elements you need in a dictionary:"))

for i in range(0,num_elements):
    k = int(input("Enter the key:"))
    v = int(input("Enter the value:"))
    d.update({k:v})

print(d)

print( )
print("The keys entered are as follows:")
for key in d.keys( ):
    print(key)

print( )
print("The values are as follows:")
for val in d.values():
    print(val)

k = d.get(3,'a')     #get the value of key '3' from dictionary 'd'. Else, return 'a'
print(k)

k = d.get('Subhash','0') #get the value of key 'Subhash' from dictionary 'd'. Else, return 'a'
print(k)
```

Output:

Program: 6

Output:

```
d = { }
s = "Book"

for c in s:
    d[c] = d.get(c,0) + 1

for k,v in d.items( ):
    print("key = { }\t Its occurrences = { } ".format(k,v))
```

Program: 7

Output:

```
#split will return a list of split items
str = "Vijay=23,Ganesh=20,Lakshmi=19,Nikhil=22"

lst = [ ]
for x in str.split(','):
    print(x)
    y = x.split('=')
    print(y)
    lst.append(y)

print(lst)

d = dict(lst)

d1 = { }
for k,v in d.items():
    d1[k] = int(v)

print(d1)
```

Program: 9

```
def func(dictionary):
    for i,j in dictionary.items():
        print(i, "=", j)

d = { "Subhash":6364024646, "Shahrukh":9999999999, "Akshay":8888888888 }

func(d)
```

Output:

Program : 10

```
#An OrderedDict is a subclass of dictionary that remembers the order that keys were first inserted

from collections import OrderedDict

d = OrderedDict( )
d[10] = 'A'
d[11] = 'B'
d[12] = 'C'
d[13] = 'D'

for i, j in d.items():
    print(i,j)
```

Output:

Program: 11

```
setone = set( )
print(setone)

settwo = { 1, 2, 3 }
print(settwo)

settwo.add(4)
print(settwo)

settwo.remove(4)
print(settwo)

setthree = { 5, 6, 7, 1, 2 }
print(setthree)

#union
setone = settwo | setthree
print(setone)

#intersection (common in both)
setone = settwo & setthree
print(setone)

#difference
setone = settwo – setthree
print(setone)

# symmetric difference
setone = settwo ^ setthree
print(setone)

s = "Godd"
setfour = set(s)
print(setfour)
setfour.add(3)
print(setfour)
setfour.add("Dogg")
print(setfour)

tone = (1,2,3)
ttwo = (4,5,6)
setfive = set(tone)
print(setfive)
setfive.add(ttwo)
print(setfive)
```

Output:

```
set()
{1, 2, 3}
{1, 2, 3, 4}
{1, 2, 3}
{1, 2, 5, 6, 7}
{1, 2, 3, 5, 6, 7}
{1, 2}
{3}
{3, 5, 6, 7}
{'d', 'o', 'G'}
{'d', 'o', 3, 'G'}
{3, 'd', 'G', 'Dogg', 'o'}
{1, 2, 3}
{1, 2, 3, (4, 5, 6)}
{1, 2, 3}
3
frozenset({'yellow', 'green', 'red'})
```

```
setsix = set([1,2,3])
print(setsix)
print(len(setsix))

setseven = frozenset(['red', 'yellow', 'green'])
print(setseven)
```

Guess The Output:

6.
```
>>> s = frozenset([1,2,3])
>>> s
frozenset({1, 2, 3})
>>> s.add(2)
```

7.
```
>>> animals = { }
>>> animals.setdefault('tiger',0)
>>> animals
```

8.
```
>>> s = {'2':0, '1': 1}
>>> sorted(s)
```

Program: 11
#This is a self-observation program

```
l = list( )
print(l)

l = list([1,2,3])
print(l)

l = list({3,4,5})
print(l)

l = list((1,2,3))
print(l)

l = list({"subhash":123, "charan":9999999999})
print(l)

l = list(({"subhash":123, "charan":9999999999}, {"shuba":8888888888}))
print(l)
```

```
l.extend([{"puppy":999, "kuppy": 888}])
print(l)

l.extend([1])
print(l)
```

Output:

```
[ ]
[1, 2, 3]
[3, 4, 5]
[1, 2, 3]
['subhash', 'charan']
[{'subhash': 123, 'charan': 9999999999}, {'shuba': 8888888888}]
[{'subhash': 123, 'charan': 9999999999}, {'shuba': 8888888888}, {'puppy': 999, 'kuppy': 888}]
[{'subhash': 123, 'charan': 9999999999}, {'shuba': 8888888888}, {'puppy': 999, 'kuppy': 888}, 1]
$
```

Solving Same Problem Using List – Dictionary – Set

Program Name: Display the vowels found in the given word.

Version-1:

```
vowels = [‘a’, ‘e’, ‘i’, ‘o’, ‘u’ }
word = input(“Provide a word to search for vowels”)
for letter in word:
        if letter in vowels:
                print(letter)
```

Version-2:

```
vowels = [‘a’, ‘e’, ‘i’, ‘o’, ‘u’ }
word = input(“Provide a word to search for vowels”)
found = [ ]
for letter in word:
        if letter in vowels:
                if letter not in found:
                        found.append(letter)
```

Version-3:

```
vowels = ['a', 'e', 'i', 'o', 'u' ]
word = input("Provide a word to search for vowels")
found =  { }

found['a'] = 0
found['e'] = 0
found['i' ]  = 0
found['o'] = 0
found['u'] = 0

for letter in word:
        if letter in vowels:
                found[letter]  += 1

for k,v in sorted(found.items( )):
        print(k, 'was found', v, 'times(s).')
```

Crashing Version:

```
vowels = ['a', 'e', 'i', 'o', 'u' }
word = input("Provide a word to search for vowels")
found =  { }

for letter in word:
        if letter in vowels:
                found[letter]  += 1

for k,v in sorted(found.items( )):
        print(k, 'was found', v, 'times(s).')
```

Version-4:

```
vowels = ['a', 'e', 'i', 'o', 'u' }
word = input("Provide a word to search for vowels")
found =  { }

for letter in word:
        if letter in vowels:
                found.setdefault(letter,0)
                found[letter]  += 1
```

```
for k,v in sorted(found.items( )):
        print(k, 'was found', v, 'times(s).')
```

Version-5:

```
vowels = set('aeiou')
word = input("Provide a word to search for vowels:")
found = vowels.intersection(set(word))
for vowel in found:
        print(vowel)
```

Guess The Output:

```
9)                                      #Error
count = 0

def example( ):
        count = count + 1

example( )
```

```
10)                                     #No Error
list = list( )

def example( ):
        list.append(1)

example( )
print(list)
```

```
11)                                     #No Error
d = dict( )

def example( ):
        d.update({'subhash':6364024646})

example( )
print(d)
```

```
12)                                     #No Error
s = set( )
```

```
def example( ):
        s.add(1)

example( )
print(s)
```

13)

```
l = list([1,2])

def example( ):
        l = list( )
        print(id(l))
        print(l)

example( )
print(id(l))
print(l)
```

14)

```
l = list([1,2])

def example( ):
   global l
   l = list()
   print(id(l))
   print(l)

example( )
print(id(l))
print(l)
```

Programming Assignments:

1. WAP to read a line from a file and remove extra space between words and re-write in another file
2. WAP to print all the digits found in a file that is filled with text and digits
3. WAP to reverse the words in a string
4. WAP to find the number of words in a file
5. WAP to read a file and print the number of times the letter 'a' has occurred in the file.
6. WAP to interleave two lines in a file with each letters. For example: two lines can be "Hello How Are you", "Hello, I am Fine". Output must be: "HHelllloo,, HIoawmFAirnye You"