Unit 4 - Lesson 8 - Built-in Dictionary Functions

About this unit

Built-in Dictionary Functions

Built-in Dictionary Functions

Unit • 100% completed

Basics of Dictionaries - Python

Assessment

About this unit

Built-in Dictionary Functions

Understanding dictionary functions

Question

Write a program to take tuple and list to create dictionary

Question

Write a program to print a dictionary with key order and value order

Question

Concatenate Two Dictionaries

Question

37.1.1. Understanding dictionary functions

06:30

Let's learn about the list of functions that can be applied to a dictionary:

1. **all()**:
   It returns **True** if all the keys of a dictionary are true, else returns **False**.

```
dict1 = {1:'alpha', 2:'beta', 3:'gamma', 4:'music'}
print(all(dict1)) # Result: True
dict2 = {1:'alpha', 2:'', '':'gamma', 4:'music'}
print(all(dict2)) # Result: False
```

2. **any()**:
   It returns **True** if atleast a single key of a dictionary is **True** else **False**.

```
dict2 = {1:'alpha', 2:'', '':'gamma', 4:'music'}
print(any(dict2)) # Result: True
```

3. **len()**:
   It calculates the length of the dictionary i.e., the number of key: value pairs.

```
dict1 =  {1:'alpha', 2:'beta', 3:'gamma', 4:'music'}
print(len(dict1)) # Result: 4
```

4. **sorted()**:
   It returns a sorted list of dictionary's keys.

```
dict1 =  {2:'alpha', 3:'beta', 1:'gamma', 4:'music', 6:'video'}
```

Sample Test Cases +

Explorer

Dictfuncs...

```
1   data1 = input("data1: ")
2   data2 = input("data2: ")
3   list1 = data1.split(",")
4   list2 = data2.split(",")
5   #create a dictionary using the lists
6   dict1 = dict(zip(list1,list2))
7   #perform all(), any(), len() and sorted() on the
8   print("all(dict1):",all(dict1))
9   print("any(dict1):",any(dict1))
10  print("len(dict1):",len(dict1))
11  print("sorted(dict1):",sorted(dict1))
12  print("key,value of dictionary:")
13  for i in sorted(dict1):
14      print(("{}:{}").format(i,dict1[i]))
15
16
```

Average time **0.010 s** 9.75 ms

Maximum time **0.010 s** 10.00 ms

2 out of 2 shown test case(s) passe

2 out of 2 hidden test case(s) passe

Test case 1 10 ms

Expected output	Actual output
data1: 1,2,3,4,5	data1: 1,2,3,4,5
data2: One,Two,Three,Four,Five	data2: One,Two,Three,Four
all(dict1): True	all(dict1): True
any(dict1): True	any(dict1): True
len(dict1): 5	len(dict1): 5
sorted(dict1): ['1', '2', '3', '4', '5']	sorted(dict1): ['1', '2',

Terminal Test cases

27.1.2. Write a program to take tuple and list to create dictionary

Given a dictionary with keys as tuples and values as lists. Write a program that prints the keys and values as shown in the **Sample Input and Output**.

Sample Input and Output:

```
Graham Chapman [4, 5, 6]
John Cleese [1, 2, 3]
Terry Gilliam [13, 14, 15, 16, 17, 18]
Eric Idle [7, 8, 9]
Terry Jones [10, 11, 12]
Michael Palin [19, 20]
```

Sample Test Cases

Test case 1

```
Graham Chapman [4, 5, 6]
John Cleese [1, 2, 3]
Terry Gilliam [13, 14, 15, 16, 17, 18]
Eric Idle [7, 8, 9]
Terry Jones [10, 11, 12]
Michael Palin [19, 20]
```

Question Hints

Explorer

DictTuple...

```
troupe = {('Cleese', 'John') : [1, 2, 3],
          ('Chapman', 'Graham') : [4, 5, 6],
          ('Idle', 'Eric') : [7, 8, 9],
          ('Jones', 'Terry') : [10, 11, 12],
          ('Gilliam', 'Terry') : [13, 14, 15, 16, 17
          ('Palin', 'Michael') : [19, 20]}

#write your code here
for i in (sorted(troupe)):
    print(" ".join(map(str,tuple(reversed(i)))),troup
```

Average time: 0.004 s (4.00 ms)

Maximum time: 0.004 s (4.00 ms)

1 out of 1 shown test case(s) passed

Test case 1 4 ms

Debug

Expected output	Actual output
Graham Chapman [4, 5, 6]	Graham Chapman [4, 5, 6]
John Cleese [1, 2, 3]	John Cleese [1, 2, 3]
Terry Gilliam [13, 14, 15, 16, 17, 18]	Terry Gilliam [13, 14, 15, 16, 1
Eric Idle [7, 8, 9]	Eric Idle [7, 8, 9]
Terry Jones [10, 11, 12]	Terry Jones [10, 11, 12]
Michael Palin [19, 20]	Michael Palin [19, 20]

Terminal | Test cases

37.1.3. Write a program to print a dictionary with key order and value order

02:10

Write a program to print a dictionary in **key order** and then in **value order** as shown in the example.

Sample Input and Output:

```
data1: 1,4,3,2
data2: 10,40,30,20
dictionary with key order
1 10
2 20
3 30
4 40
dictionary with value order
10 1
20 2
30 3
40 4
```

Sample Test Cases

Explorer

Histogra...

```
a = input("data1: ")
b = input("data2: ")
a = a.split(',')
b = b.split(',')
#create a dictionary with the lists
d = dict(zip(a,b))

print("dictionary with key order")
#write your logic to print in key value order
for key,value in sorted(d.items()):
    print(key,value)


print("dictionary with value order")
#write your logic to print in value key order
for key,value in sorted(d.items(),key=lambda item:item[1]):
    print(value,key)
```

Average time **0.012 s** 12.00 ms

Maximum time **0.012 s** 12.00 ms

1 out of 1 shown test case(s) passed

2 out of 2 hidden test case(s) passed

Debug

Test case 1 12 ms

Expected output

```
data1: 1,4,3,2
data2: 10,40,30,20
dictionary with key order
1 10
2 20
3 30
```

Actual output

```
data1: 1,4,3,2
data2: 10,40,30,20
dictionary with key order
1 10
2 20
3 30
```

Terminal | Test cases

< Prev | Reset | Submit | Next >

37.1.4. Concatenate Two Dictionaries

09:16

Create two dictionaries with the user-given inputs. Write a program to **concatenate** (add the values with the same keys) the two input dictionaries, and print the result as shown in the sample test cases.

Sample Test Cases +

Explorer

DictConc...

```
#write your code here
data1 = list(map(int,input("data1: ").split(",")))
data2 = list(map(int,input("data2: ").split(",")))
dict1 = dict(zip(data1,data2))
data3 = list(map(int,input("data3: ").split(",")))
data4 = list(map(int,input("data4: ").split(",")))
dict2 = dict(zip(data3,data4))
dict3={}

for i in dict1:
    dict3[i]=dict1[i]
for i in dict2:
    if i in dict3:
        dict3[i]+=dict2[i]
    else:
        dict3[i]=dict2[i]
print("concatenation:",sorted(dict3.items()))
```

Average time **0.014 s** 14.00 ms

Maximum time **0.016 s** 16.00 ms

2 out of 2 shown test case(s) passed

2 out of 2 hidden test case(s) passed

Debug

Test case 1 16 ms

Expected output	Actual output
data1: 1,2,3	data1: 1,2,3
data2: 10,20,30	data2: 10,20,30
data3: 4,5,6	data3: 4,5,6
data4: 40,50,60	data4: 40,50,60
concatenation: [(1, 10), (2, 20), (3, 30), (4, 40), (5, 50), (6, 60)]	concatenation: [(1, 10), (2, 20), (3, 30), (4, (6, 60)]

Terminal Test cases

Section - 1 [Section 1 of 1] | Q1 of 4 | Marks 5

Jump to

Test time left: 03:58:47

Write the code

Write a Python program that asks the user to enter a text as input from the user and return user a dictionary whose keys are the words of the text entered and the values are the reverse of the words that make up the text.

Note:

- **Refer to the Displayed test cases for a better understanding.**
- **In text words are separated by a single space character.**

Constraints:

1 <= no of elements <= 10

Sample Test case:

python is easy ----> Enter the text

{'python': 'nohtyp', 'is': 'si', 'easy': 'ysae'} ----> Print the dictionary as required.

Instructions:

- Your input and output must follow the input and output layout mentioned in the visible sample test case.
- Hidden test cases will only pass when the user's input and output match the expected input and output.

Sample Test Cases

Test Case 1:

Expected Output:

python is easy

```
{'python': 'nohtyp', 'is': 'si', 'easy': 'ysae'}
```

Test Case 2:

dictionary3.py

Submit

```
1  #write your code here
```

Execution Results

0 out of 4 shown cases successful

0 out of 3 hidden cases successful

Show only failed cases

✖ Test Case - 1 (Execution Time: 5 ms)

Expected Output	User Output
python is easy {'python': 'nohtyp', 'is': 'si', 'easy': 'ysae'}	python is easy Empty

⚠ indicates the mismatch in the expected output

✖ Test Case - 2 (Execution Time: 4 ms)

Expected Output	User Output
welcome to programming {'welcome': 'emoclew', 'to': 'ot', 'programming': 'gnimmargorp'}	welcome to programming Empty

Finish | Clear | Submit | Prev | Next

Printing a Dictionary

Take the keys and values of the dictionary and print the dictionary. After that, print the same dictionary with values as keys and keys as values.

Note:

- Refer to the Displayed test cases for a better understanding.
- keys must be strings and values must be integers.

Constraints:

1 <= no of key-value pairs <= 10

Sample Test Case:

keys: a,b,c,d,e -------> Enter keys
values: 1,2,3,4,5 -------> Enter values
{'a': 1, 'b': 2, 'c': 3, 'd': 4, 'e': 5} -------> Print the dictionary
{1: 'a', 2: 'b', 3: 'c', 4: 'd', 5: 'e', 6: 'f'} -----> print the dictionary as required

Sample Test Cases

Test Case 1:

Expected Output:

```
keys: a,b,c,d,e,f
values: 1,2,3,4,5,6
{'a': 1, 'b': 2, 'c': 3, 'd': 4, 'e': 5, 'f': 6}
{1: 'a', 2: 'b', 3: 'c', 4: 'd', 5: 'e', 6: 'f'}
```

Test Case 2:

Expected Output:

```
keys: a,b,c
values: 11,12
```

Finish Clear

dictionary3.py

Submit

```
1  # write your code here
2
```

Execution Results

0 out of 3

0 out of 3

Show only failed cases

✖ **Test Case - 1** (Execution Time: 4 ms)

Expected Output	User Output
keys: a,b,c,d,e,f	a,b,c,d,e,f
values: 1,2,3,4,5,6	Empty1,2,3,4,5,6
{'a': 1, 'b': 2, 'c': 3, 'd': 4, 'e': 5, 'f': 6}	Empty
{1: 'a', 2: 'b', 3: 'c', 4: 'd', 5: 'e', 6: 'f'}	Empty

⚠ : indicates the mismatch in the expected output

✖ **Test Case - 2** (Execution Time: 4 ms)

Expected Output	User Output
keys: a,b,c	a,b,c
values: 11,12	Empty 11,12

Submit Prev Next

Nested Dictionary

Take the keys and values of the dictionary and also the list of integers and print the dictionary as well as list. After this your task is to map each element of list with each item of dictionary, forming **nested dictionary** as resultant value.

Note:

- Refer to the Displayed test cases for a better understanding.
- keys must be strings and values must be integers.

Constraints:

1 <= no of key-value pairs <= 8

Sample Test Case:

keys: a,b,c,d -----> Enter keys of the dictionary
values: 1,2,3,4 -----> Enter values of the dictionary
4,5,6,7 ----> Enter the list of integers
{'a': 1, 'b': 2, 'c': 3, 'd': 4} ----> Print the dictionary with keys and values inputted in line 1 and 2.
[4, 5, 6, 7] ----> Print the list
{4: {'a': 1}, 5: {'b': 2}, 6: {'c': 3}, 7: {'d': 4}} ----> Print the dictionary by mapping each element in the list as the key and key-value pairs of the dictionary as the corresponding values.

Sample Test Cases

Test Case 1:

Expected Output:

```
keys: a,b,c,d
values: 1,2,3,4
4,5,6,7
{'a': 1, 'b': 2, 'c': 3, 'd': 4}
[4, 5, 6, 7]
{4: {'a': 1}, 5: {'b': 2}, 6: {'c': 3}, 7: {'d': 4}}
```

dictionary3.py

```
1 # write your code here
2
```

Submi

Execution Results

0 out of 3 sample cases passed

0 out of 2 hidden cases passed

Show only failed cases

✖ Test Case - 1 (Execution Time: 4 ms)

Expected Output	User Output
keys: a,b,c,d	a,b,c,d
values: 1,2,3,4	Empty 1,2,3,4
4,5,6,7	4,5,6,7
{'a': 1, 'b': 2, 'c': 3, 'd': 4}	Empty
[4, 5, 6, 7]	Empty
{4: {'a': 1}, 5: {'b': 2}, 6: {'c': 3}, 7: {'d': 4}}	Empty

⚠ indicates the mismatch in the expected output

✖ Test Case - 2 (Execution Time: 3 ms)

Expected Output	User Output

Write the code

Take an integer **n** from the user. Your task is to write a program to generate and print a dictionary that contains a number (between 1 and n) in the form (x : x*x*x).

Note:

- **Refer to the Displayed test cases for a better understanding.**

Constraints:

0 <= n <= 10

Sample Test case:

5
{1: 1, 2: 8, 3: 27, 4: 64, 5: 125}

Instructions:

- Your input and output must follow the input and output layout mentioned in the visible sample test case.
- Hidden test cases will only pass when the user's input and output match the expected input and output.

Sample Test Cases

Test Case 1:

Expected Output:

5
{1: 1, 2: 8, 3: 27, 4: 64, 5: 125}

Test Case 2:

Expected Output:

1

Finish Clear

dictionary1.py

```
1 #write your code here
```

Execution Results

0 out of 3 shown cases

0 out of 3 hidden cases

Show only failed

✖ Test Case - 1 (Execution Time: 3 ms)

Expected Output	User Output
5 {1: 1, 2: 8, 3: 27, 4: 64, 5: 125}	5 Empty

⚠ : indicates the mismatch in the expected output

✖ Test Case - 2 (Execution Time: 5 ms)

Expected Output	User Output
1 {1: 1}	1 Empty

⚠ : indicates the mismatch in the expected output

Submit Prev Next