


Unit 4 - Lesson 9 - Built-in Dictionary Methods

About this unit

Built-in Dictionary Methods

Built-in Dictionary Methods

Unit • 100% completed



Built-in Dictionary Functions and Methods - Python

Assessment



Built-in Dictionary Methods

About this unit

Built-in Dictionary Methods

? Understand the Methods in Dictionary

Question



38.1.1. Understand the Methods in Dictionary

update()	<pre>dict1.update(dict2) print(dict1) # Result: {'cyan': 1, 'vi</pre>	the calling dictionary(dict1 in example)
values()	<pre>dict1 = {'cyan': 1, 'vi dict1.values() # Result: dict_values([</pre>	Returns a list of dictionary's values

Create two dictionaries **dict1** and **dict2** with the user given (key, value) pairs.

1. Print the **dict1** in sorted order.
2. Create a copy of **dict1**, and print it.
3. Print the keys and values(separately) of the **dict1** in sorted order.
4. Update the sorted **dict1** with the **dict2**, print the result as shown in the example.

Sample Input and Output:

```
data1: 1,2,3
data2: 10,20,30
data3: a,b,c
data4: Blue,Yellow,Cyan
sorted dictionary: [('1', '10'), ('2', '20'), ('3', '30')]
copy of sorted dictionary: [('1', '10'), ('2', '20'), ('3', '30')]
sorted keys: ['1', '2', '3']
sorted values: ['10', '20', '30']
sorted dictionary after updation: [('1', '10'), ('2', '20'), ('3', '30'), ('a', 'Blue'), ('b', 'Yellow'), ('c', 'Cyan')]
```

Sample Test Cases




```
1 #write your code here
2 a = input("data1: ")
3 b = input("data2: ")
4 c = input("data3: ")
5 d = input("data4: ")
6 l1=a.split(",")
7 l2=b.split(",")
8 l3=c.split(",")
9 l4=d.split(",")
10 d1=dict(zip(l1,l2))
11 d2=dict(zip(l3,l4))
12 sd=sorted(d1.items())
13 print("sorted dictionary:",sd)
14 csd=sd.copy()
15 print("copy of sorted dictionary:",csd)
16 skd=sorted(d1.keys())
17 print("sorted keys of dictionary:",skd)
18 svd=sorted(d1.values())
19 print("sorted values of dictionary:",svd)
20 d1.update(d2)
21 sdu=sorted(d1.items())
22 print("sorted dictionary after updation:",sdu)
```

Average time

0.021 s

21.25 ms



Maximum time

0.022 s

22.00 ms



2 out of 2 shown test case(s) passed

2 out of 2 hidden test case(s) passed

Terminal

Test cases




```
1 #write your code here
2 a = input("data1: ")
3 b = input("data2: ")
4 c = input("data3: ")
5 d = input("data4: ")
6 ll=a.split(",")
```

Average time
0.021 s
21.25 ms

Maximum time
0.022 s
22.00 ms

✓ 2 out of 2 shown test case(s) passed

✓ 2 out of 2 hidden test case(s) passed

✓ Test case 1 22 ms

Debug

Expected output

data1: 1,2,3
data2: 10,20,30
data3: a,b,c
data4: Blue,Yellow,Cyan
sorted dictionary: [('1', '10'), ('2', '20'), ('3', '30')]
copy of sorted dictionary: [('1', '10'), ('2', '20'), ('3', '30')]
sorted keys of dictionary: ['1', '2', '3']
sorted values of dictionary: ['10', '20', '30']
sorted dictionary after updation: [('1', '10'), ('2', '20'), ('3', '30'), ('a', 'Blue'), ('b', 'Yellow'), ('c', 'Cyan')]

Actual output

data1: 1,2,3
data2: 10,20,30
data3: a,b,c
data4: Blue,Yellow,Cyan
sorted dictionary: [('1', '10'), ('2', '20'), ('3', '30')]
copy of sorted dictionary: [('1', '10'), ('2', '20'), ('3', '30')]
sorted keys of dictionary: ['1', '2', '3']
sorted values of dictionary: ['10', '20', '30']
sorted dictionary after updation: [('1', '10'), ('2', '20'), ('3', '30'), ('a', 'Blue'), ('b', 'Yellow'), ('c', 'Cyan')]

✓ Test case 2 20 ms

Terminal Test cases

Write the code

Write a Python program to filter a dictionary based on **values** that are the multiples of 5.

Note:

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

Constraints:

1 <= no of key value pairs <= 10

Sample Test case:

keys: a,b,c,d,e ----> Enter keys

values: 13,15,60,45,79 ----> Enter values.

{'a': 13, 'b': 15, 'c': 60, 'd': 45, 'e': 79} ----> Print the dictionary

{'b': 15, 'c': 60, 'd': 45} ----> Print the filtered dictionary who values are only the multiples of 5, in order of their occurrence.

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:

```
keys: a,b,c,d,e
values: 13,15,60,45,79
{'a': 13, 'b': 15, 'c': 60, 'd': 45, 'e': 79}
{'b': 15, 'c': 60, 'd': 45}
```

dictionary3.py

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

Execution Results

0 out of 3 shown cases succeeded

0 out of 3 hidden cases succeeded

Show only failed

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

Expected Output

```
keys: a,b,c,d,e
values: 13,15,60,45,79
{'a': 13, 'b': 15, 'c': 60, 'd': 45, 'e': 79}
{'b': 15, 'c': 60, 'd': 45}
```

User Output

```
a,b,c,d,e
Empty 13,15,60,45,79
Empty
Empty
```

⚠ indicates the mismatch in the expected output

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

Expected Output

```
keys: a,b,c,d,e,f
values: 1,2,3,4,6,7
```

User Output

```
a,b,c,d,e,f
Empty 1,2,3,4,6,7
```

Write the code

Write a program to do the following operations:

1. Create an empty dictionary with `dict()` method, and print it.
2. Add keys and values to the dictionary and print it
3. Update the dictionary by giving another key value pair (Entered key-value pair must be separated with comma).

Note:

- Refer to the Displayed test cases for a better understanding.
- Take keys as strings and values as integers.
- If the key doesn't exist in the dictionary, then add key-value pair as a new entry in the dictionary at the end.

Constraints:

1 <= no of key value pairs <= 10

Sample Test case:

`{}` ----> print Empty dictionary

keys: a,b,c,d ----> Enter the keys

values: 1,2,3,4 ----> Enter the values

`{'a': 1, 'b': 2, 'c': 3, 'd': 4}` ----> Print the dictionary

e,5 ----> Enter the key,value pair that need to be updated

`{'a': 1, 'b': 2, 'c': 3, 'd': 4, 'e': 5}` ----> Print the updated dictionary.

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

dictionary3.py

Submit

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

Finish

Clear

Submit

Prev

Next

Write the code

Write a program to do the following operations:

1. Create an empty dictionary with **dict()** method, and print it.
2. Add keys and values to the dictionary and print it
3. Update the dictionary by giving another key value pair (Entered key-value pair must be separated with comma).

Note:

- Refer to the Displayed test cases for a better understanding.
- Take keys as strings and values as integers.
- If the key doesn't exist in the dictionary, then add key-value pair as a new entry in the dictionary at the end.

Constraints:

1 <= no of key value pairs <= 10

Sample Test case:

{ } ----> print Empty dictionary

keys: a,b,c,d ----> Enter the keys

values: 1,2,3,4 ----> Enter the values

{'a': 1, 'b': 2, 'c': 3, 'd': 4} ----> Print the dictionary

e,5 ----> Enter the key,value pair that need to be updated

{'a': 1, 'b': 2, 'c': 3, 'd': 4, 'e': 5} ----> Print the updated dictionary.

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

Finish

Clear

dictionary3.py

Submit

1 #write your code here

Execution Results

0 out of 3 shown cases successful

0 out of 3 hidden cases successful

Show only failed cases

✖ Test Case - 1 (Execution Time: 6 ms)

Expected Output

User Output

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

```
{ }
Empty a,b,c,d
Empty 1,2,3,4
Empty
e,5
Empty
```

⚠ indicates the mismatch in the expected output

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

Expected Output

User Output

Submit

Prev

Next



Write the code

Write a Python program to print a dictionary in table format. For this you are supposed to take values as the list of strings.

Note:

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

Constraints:

1 <= no of key value pairs <= 10

Sample Test case:

keys: 11,22 ----> Enter keys (Here number of keys are 2)

values: ----> Enter values as list of strings

a,b ----> list 1

c,d ----> list 2

{11: ['a', 'b'], 22: ['c', 'd']} ----> Print the dictionary with keys and values entered.

11 22 ----> Print the dictionary in table format, here each column represents key and values in the list.

a c

b d

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:

Finish

Clear

dictionary3.py

1 #write your code here

Execution Results

0 out of 2 shown cases successful

0 out of 3 hidden cases successful

Show only failed cases

× Test Case - 1 (Execution Time: 6 ms)

Expected Output

keys: 11,22

values:

a,b

c,d

{11: ['a', 'b'], 22: ['c', 'd']}

11 22

a c

b d

User Output

11,22

Empty

a,b

c,d

Empty

Empty

Empty

Empty

⚠ : indicates the mismatch in the expected output.

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

Submit

Prev

Next



Write the code

Write a Python program to remove key value pair's having duplicate(Repeated) values from the Dictionary.

Note:

- Refer to the Displayed test cases for a better understanding.
- Take keys as integers and values as strings.
- Dictionary can consists of keys without values or values without keys.

Constraints:

1 <= no of key value pairs <= 10

Sample Test case:

keys: 1,1,1,11,1,1,1 ----> Enter the keys

values: a,b,c,d,c,d,c,da,a,b ----> Enter the values

{1: 'c', 11: 'd'} ----> Print the dictionary

{1: 'c', 11: 'd'} ----> Print the dictionary after removing duplicate values.

Brief editorial:

Here key value pair that is part of the result is the recent key value pair i.e. 1: 'c' (Rest all after coming enteries are treated as duplicates) and 11: 'd' is already having having unique value.

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:

dictionary3.py

Submit

1 #write your code here

Execution Results

0 out of 3 shown cases successful

0 out of 3 hidden cases successful

Show only failed cases

Test Case - 1 (Execution Time: 6 ms)

Expected Output

keys: 1,1,1,11,1,1,1
values: a,b,c,d,c,d,c,da,a,b
{1: 'c', 11: 'd'}
{1: 'c', 11: 'd'}

User Output

1,1,1,11,1,1,1
Empty a,b,c,d,c,d,c,da,a,b
Empty
Empty

⚠ indicates the mismatch in the expected output.

Test Case - 2 (Execution Time: 3 ms)

Expected Output

keys: 1,11,111,1111,11111
values: a,\$,\$,\$,\$,\$

User Output

1,11,111,1111,11111
Empty a,\$,\$,\$,\$,\$

Submit

Prev

Next

