ISYS90088
Introduction to Application Development

Assignment Project Exam Help

https://powcoder.com

Week 8 – Contd. from week 6: tuples & Dictionaries

Add WeChat powcoder

Semester 2 , 2018

Dr Antonette Mendoza

s

Objectives

After completing this lecture, you will be able to:

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

- Work with Tuples
- Work with Dictionaries

Lists, tuples and Dictionaries

- A **list** allows the programmer to manipulate a sequence of data values of any types
  - Indicate by enclosing its elements in []

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

Lists, tuples and Dictionaries

- A **list** allows the programmer to manipulate a sequence of data values of any types
  - Indicate by enclosing its elements in []
- A **tuple** resembles a list, but is immutable
  - Indicate by enclosing its elements in ()

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

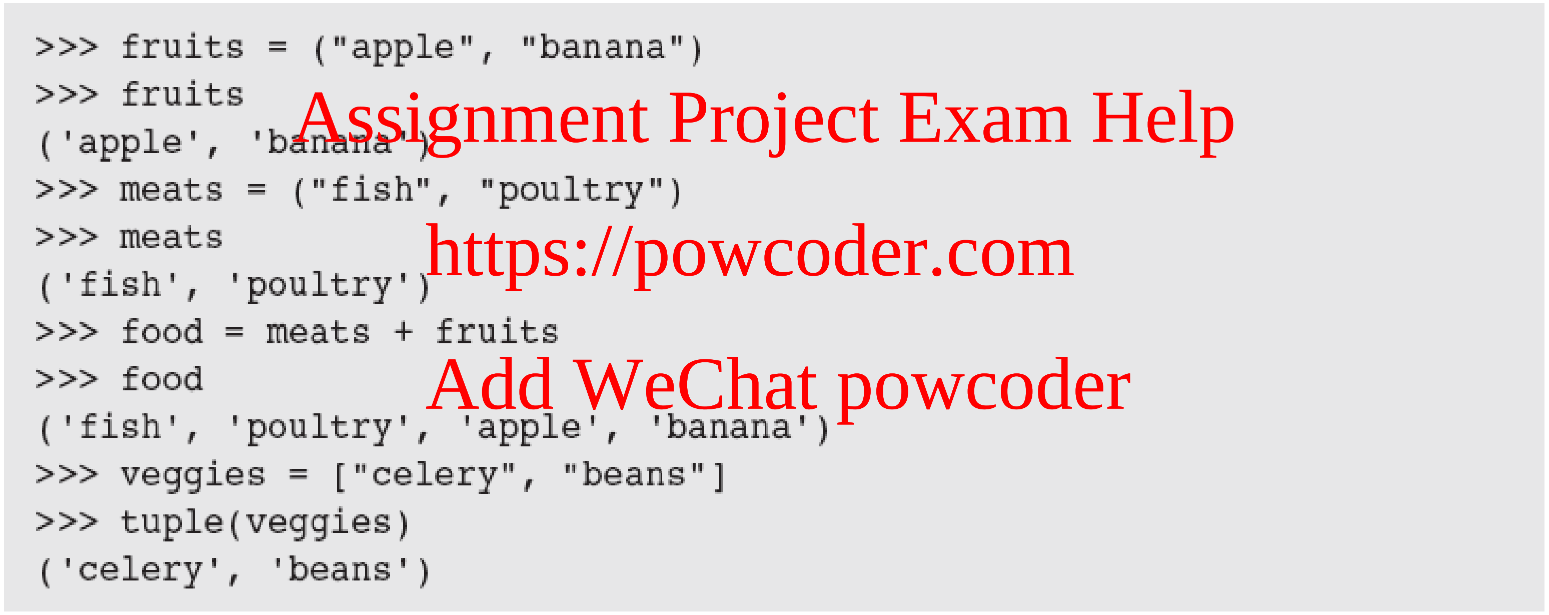
Tuples

- A **tuple** resembles a list, but is immutable
  - Indicate by enclosing its elements in ()
- The differences between tuples and lists are:
  - the tuples cannot be changed unlike lists
  - tuples use parentheses, whereas lists use square brackets

- Creating a tuple is as simple as putting different comma-separated values.

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Tuples

- Lists can be converted to tuples; two sets of tuples can be concatenated
- 

```
>>> fruits = ("apple", "banana")
>>> fruits
('apple', 'banana')
>>> meats = ("fish", "poultry")
>>> meats
('fish', 'poultry')
>>> food = meats + fruits
>>> food
('fish', 'poultry', 'apple', 'banana')
>>> veggies = ["celery", "beans"]
>>> tuple(veggies)
('celery', 'beans')
```

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

- Most of the operators and functions used with lists can be used in a similar fashion with tuples

Tuples

- Most of the operators and functions used with lists can be used in a similar fashion with tuples:

  - The empty tuple is written as two parentheses containing nothing

Assignment Project Exam Help

https://powcoder.com

```
tup1 = ();
```

Add WeChat powcoder

  - To write a tuple containing a single value you have to include a comma, even though there is only one value —

```
tup1 = (50,);
```

Tuples

- Most of the operators and functions used with lists can be used in a similar fashion with tuples:

  - The empty tuple is written as two parentheses containing nothing

Assignment Project Exam Help

```
      tup1 = ();
```

https://powcoder.com

  for lists: `list1 = []`

Add WeChat powcoder

  - To write a tuple containing a single value you have to include a comma, even though there is only one value —

```
      tup1 = (50,);
   For lists: list1 = [50]
```

Tuples

- Like string indices, tuple indices start at 0. The operations performed are: concatenation, iteration, in, slicing and indexing

- Accessing Values in Tuples: use the square brackets for slicing along with the index or indices to obtain value available at that index.

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

- Updating Tuples - Tuples are immutable which means you cannot update or change the values of tuple elements.

- Delete Tuple Elements - Removing individual tuple elements is not possible.

Tuples

➢ To explicitly remove an entire tuple, just use the **del** statement. For example:

```
tuple1 = ('physics', 'chemistry', 1997, 2000)
print (tuple1)
del tuple1
print ("After deleting tuple : ")
print (tuple1)
```

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

- This produces the following result (check example). Note an exception raised, this is because after **del tup** tuple does not exist any more.

Example:

```
>>>tuple3 = (1,2,3)
>>>list(tuple3)
[1,2,3]
```

Tuples

Built-in Tuple Functions can be used:

```
## length, max and min in a tuple
   tuple1, tuple2 = ('zar', 'xyz', 'zara'), (100, 500, 20)
   print ("Max value element : ", max(tuple1))
   print ("Max value element : ", max(tuple2))
   print ("Min value element : ", min(tuple1))
   print ("Min value element : ", min(tuple2))
   print ("First tuple length : ", len(tuple1))
   print ("Second tuple length : ", len(tuple2))

#convert a list of items into tuples
   Listofitems = [23, 'years', 'dogs', 'cats'];
   toaTuple = tuple(Listofitems)
   print ("Tuple elements : ", toaTuple)
```

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Difference between lists and tuples

- Lists are mutable. Lists however have this method called append. In order for most of your appends to be fast, python will actually create a larger array in memory *just in case* you append.
- This way, when you do append, it does not have to recreate a list every time. You can add items to the list .How would it know that you don't want to maybe add a 4th 5th 6th element? To play safe, we assume you might want more in the memory
- On the other hand, by using tuples, it tells python that you want an immutable structure. Give me space for 3 things, fill those slots up, and move on.
- Since tuples are immutable, this means that tuples are fixed. We can't do anything to them in memory.
- Performance: processing of tuples is said to be faster than list processing
- Using tuples is safe: Since they are immutable, we cant change content of the tuple. This can be useful when you don't want any data modified by your code

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Lists, tuples and Dictionaries

- A **list** allows the programmer to manipulate a sequence of data values of any types
  - Indicate by enclosing its elements in []
- A **tuple** resembles a list, but is immutable
  - Indicate by enclosing its elements in ()

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

- A **dictionary** organizes data values by association with other data values rather than by '**sequential position**'

- Lists and dictionaries provide powerful ways to organize data in useful and interesting applications

Dictionaries

- A dictionary organizes information by **association**, not position
  - Example: When you use a dictionary to look up the definition of "Mammal," you don't start at page 1; instead, you turn to the words beginning with "M"
- Data structures organized by association are also called **tables** or **association lists**
- In Python, a **dictionary** associates a set of **keys** with data values

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Dictionary Literals

- A Python dictionary is written as a sequence of key/value pairs separated by commas
  - Pairs are sometimes called **entries**
  - Enclosed in curly braces ({ and })
  - A colon (:) separates a key and its value
- Examples:

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

```
{'Sarah':'476-3321', 'Nathan':'351-7743'}  #A Phone book
{'Name':'Molly', 'Age':18}     # Personal information
{}                            #An empty dictionary
```

Mixing data types in a dictionary

- Keys in a dictionary are immutable but their associated values can be of any type. For example, the values can be lists.

d1 = {‘matt’: [23, 2000, 2010], ‘anne’: [25, 2545, 2012], ‘jack’: [34, 2500, 2011]}

Assignment Project Exam Help

- The values stored in a single dictionary can be of different types. For example one element in the dictionary can be a string, another an integer, another a float etc..

https://powcoder.com

Add WeChat powcoder

Example:

```
>>> employee_record = {'name':'kevin', 'Age': 43,
'ID':23145, 'payrate':24.99}
>>>employee_record
{'Age': 43, 'name': 'kevin', 'ID': 23145,
'payrate': 24.99}
```

Properties of Dictionary Keys

Dictionary values have no restrictions. However, same is not true for the keys. There exists a mapping between keys and the values. There are two important points to remember about dictionary keys –

(a) More than one entry per key not allowed. Which means no duplicate key is allowed. When duplicate keys encountered during assignment, the last assignment wins. For example –

Assignment Project Exam Help

https://powcoder.com

```
>>>dict = {'Name': 'Amie', 'Age': 10, 'Name': 'Jack'}
>>> print ("dict['Name']: ", dict['Name'])
```

Add WeChat powcoder

When the above code is executed, it produces the following result –

```
dict['Name']: ‘jack’  => Why?
```

Dictionaries: as hash tables to explain immutability

(b) Keys are immutable. For example, when you insert a key into a hash table, the hash table asks the key for its hash code, and remembers it along with the key itself and the associated value. When you later perform a lookup, the hash table asks the key you're looking for its hash code, and can very quickly find all the keys in the table that have the same hash code.

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

- If a key is mutable, then finding the value associated with the unique key is not possible – the hash table would be messed up as you wont know accurately the associated value to the specific key.

Adding Keys and Replacing Values

- Add a new key/value pair to a dictionary using **[]**:

```
<a dictionary>[<a key>] = <a value>
```

- Example:

Assignment Project Exam Help

```
>>> info = {}
>>> info["name"] = "Sandy"
>>> info["occupation"] = "hacker"
>>> info
{'name': 'Sandy', 'occupation': 'hacker'}
>>>
```

https://powcoder.com

Add WeChat powcoder

- Use **[]** to replace a value at an existing key:

```
>>> info["occupation"] = "manager"
>>> info
{'name': 'Sandy', 'occupation': 'manager'}
>>>
```

Accessing Values

- Use **[]** to obtain the value associated with a key
  - If key is not present in dictionary, an error is raised

```
>>> info["name"]
'Sandy'
>>> info["job"]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyError: 'job'
>>>
```

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

- If the existence of a key is uncertain, test for it using the dictionary method **get**

```
>>> print(info.get("job", None))
None
>>>
```

Deleting elements

- To delete an existing key-value pair from a dictionary, use the **`del`** statement. After the statement is executed, the key and its associated value will be deleted from the dictionary. Assignment Project Exam Help
- If the key does not exist, a KeyError exception is raised.

https://powcoder.com

Syntax:

Add WeChat powcoder

```
del dictionary_name[key]
```

Note: To prevent the KeyError from being raised, use the in opeartor to determine whether the key exists before you try to delete it and its associated value.

Dict: Del and Clear

>>>dict = {'Name': ’Anne', 'Age': 10, 'Class': ’Third’}

>>>del dict['Name']; # remove entry with key 'Name’

What is the output?

>>>dict.clear();   # remove all entries in dict

What is the output?

>>>del dict ;   # delete entire dictionary

What is the output?

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

Using `in` and `not in` operators to test for a value in a Dictionary

```
#illustrate in operator to find a value in dict
phonebook = {'jack':'0423123', 'jill':'2345433',
'jane':'3334444'}

if 'jack' in phonebook:
        print(phonebook['jack'])
else:
        print('not found')


# to illustrate not in operator to find a value in
dictionary
if 'jacky' not in phonebook:
        print('not found')
```

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Using the `for` loop to iterate over a Dictionary – traverse thru' a dictionary

```
for var in <dictionary_name>:
   statement
   statement
   ...
   ...
```

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

Example:

```
employee_record = {'name':'Kevin', 'Age': 43,
'ID':23145, 'payrate':24.99}
for key in employee_record:
   print(key)
```

Note: when the dict is printed, the order is different from the initial order in the dict. This means accessing elements using an index is not possible in dictionaries.

Traversing a Dictionary

- To print all of the keys and their values:

  info = {'apple': 'jack', 'banana':'jill', 'pears': 'brad'}

```
for key in info:
   print(key, info[key])
```

Assignment Project Exam Help

- Alternative: Use the dictionary method **items()**

https://powcoder.com

```
>>> grades = {90:"A", 80:"B", 70:"C"}
>>> grades.items()
[(80, 'B'), (90, 'A'), (70, 'C')]
```

Add WeChat powcoder

  – Entries are represented as tuples within the list

```
for (key, value) in grades.items():
   print(key, value)
```

- You can sort the list first:

```
theKeys = list(info.keys())
theKeys.sort()
for key in theKeys:
   print(key, info[key])
```

Assessing information: check this!

```
>>> d1 = {'matt': [23, 2000, 2010],
'anne': [25, 2545, 2012], 'jack’:
[34, 2500, 2011]}
```

Assignment Project Exam Help

https://powcoder.com

```
>>> if 'jack' in d1:
  print(d1['jack'])
```

Add WeChat powcoder

```
[34, 2500, 2011]
>>> d1['jack'][2]
2011
```

Example code : check this out!

```
info = {'apple': ['jack', 'jane'], 'banana':
['jill'], 'pears': ['brad', 'sally']}
keylist = list(info.keys())
keylist.sort() #sorted this list
for key in keylist:
    print(key, info[key])
    print(key, info[key][0])
```

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

What does this output?

Try this - dictionary with a list of values for a key

```
list1 = []
d1 = {'matt': [23, 2000, 2010], 'anne':
[25, 2545, 2012], 'jack':
                [34, 2500, 2011]}
for list1 in d1['matt']:
   print(list1)
```

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

What is the output?

Getting the number of elements in a dictionary using **`len`**

- Use the built in method called **`len`**

Assignment Project Exam Help

Example:

```
>>> employee_record = {'name':'kevin',
'Age': 43, 'ID':23145, 'payrate':24.99}
>>>len(employee_record)
4
```

https://powcoder.com

Add WeChat powcoder

Other Dictionary methods: **`clear`**

- Use the built in method called **`clear`**

`Syntax:`

```
dictionary_name.clear()
```

Assignment Project Exam Help

Example:

https://powcoder.com

```
>>> employee_record = {'name':'kevin',
'Age': 43, 'ID':23145, 'payrate':24.99}
>>> employee_record.clear()
>>> employee_record
{}
```

Add WeChat powcoder

Other Dictionary methods: **`get`**

- Use the built in method called **`get`**
- When executed, this methods outputs the value associated with the key that is searched. If not present, will output the default value as shown in the example.

Assignment Project Exam Help

Syntax:

```
dictionary_name.get(key, default)
```

https://powcoder.com

Add WeChat powcoder

Example:

```
>>> employee_record = {'name':'kevin', 'Age':
43, 'ID':23145, 'payrate':24.99}
>>> employee_record.get(‘name’, ‘not found’}
'kevin’
>>> employee_record.get('dob', 'not found')
'not found'
```

Other Dictionary methods: **`items`**

- Use the built in method called **`items`**
- When executed, it outputs all the key-value pairs in a ***dictionary view.***

Assignment Project Exam Help

Syntax:

```
dictionary_name.items()
```

https://powcoder.com

Add WeChat powcoder

Example:

```
>>> employee_record = {'name':'kevin',
'Age': 43, 'ID':23145, 'payrate':24.99}
>>> employee_record.items()
dict_items([('payrate', 24.99), ('name',
'kevin'), ('ID', 23145), ('Age', 43)])
```

Other Dictionary methods: **`keys`**

- Use the built in method called **`keys`**
- When executed, it outputs all the keys

Syntax:

```
dictionary_name.keys()
```

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

Example:

```
>>> employee_record = {'name':'kevin',
'Age': 43, 'ID':23145, 'payrate':24.99}
>>> employee_record.keys()
ID
payrate
name
Age
```

Other Dictionary methods: **`values`**

- Use the built in method called **`values`**
- When executed, it outputs all the values in the dict

Syntax:

```
dictionary_name.values()
```

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

Example:

```
>>> employee_record = {'name':'kevin',
'Age': 43, 'ID':23145, 'payrate':24.99}
>>> employee_record.values()
24.99
kevin
23145
43
```

Other Dictionary methods: **pop and popitem**

- Use the built in method called **pop**
- When executed, it outputs the value associated with the specific key and removes it from the dict.

Syntax:

Assignment Project Exam Help

```
   dictionary_name.pop(key, default)
```

https://powcoder.com

Example:

Add WeChat powcoder

```
>>> employee_record = {'name':'kevin', 'Age': 43,
'ID':23145, 'payrate':24.99}
>>> employee_record.pop('name')
'kevin'
>>> employee_record
{'payrate': 24.99, 'ID': 23145, 'Age': 43}
>>>
```

Other Dictionary methods: **pop and popitem**

- Use the built in method called **popitem**
- When executed, it outputs a key-value pair, and it removes that key-value pair from the dict. (front of the list/dict – last in first out)

Syntax:

Assignment Project Exam Help

```
   dictionary_name.popitem()
```

Example:

https://powcoder.com

Add WeChat powcoder

```
>>> employee_record
{'payrate': 24.99, 'name': 'kevin', 'ID': 23145, 'Age':
45}
>>> employee_record.popitem()
('payrate', 24.99)
>>> employee_record.popitem()
('name', 'kevin')
>>> employee_record.popitem()
('ID', 23145)
>>>
```

Traversing: using list and dict methods

```
>>> employee_record = {'name':'kevin',
'Age': 43, 'ID':23145, 'payrate':24.99}
>>> list(employee_record.keys())
['payrate', 'name', 'ID', 'Age']
>>> list(employee_record.values())
[24.99, 'kevin', 23145, 43]
>>> >>> list(employee_record.items())
[('payrate', 24.99), ('name', 'kevin'),
('ID', 23145), ('Age', 43)]
>>>
```

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Example: creating a dict from a list

```
from collections import defaultdict
employee_list = [('yosh',23,2001),
('farah', 22, 2010), ('matt', 34, 2000)]

#you can take a list of tuples and make it
#into a dict with key-value pairs
#start with an empty dict, start a for
loop #and append values to a key

d1 = defaultdict(list)
for key, age, start_date in employee_list:
   d1[key].append(age)
   d1[key].append(start_date)
print(d1, d1.items(), d1.values())
```

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder