Python Chapter 6 Dictionaries Activities

Contents

Time required: 60 minutes

How to Think Like a Computer Scientist (Interactive Edition)

Go through the following tutorials.

Dictionaries

Online Tutorials

Go through the following tutorials.

- Python Dictionaries
    - Access Items
    - Change Items
    - Add Items
    - Remove Items
    - Loop Dictionaries
    - Copy Dictionaries

Python Dictionaries

A dictionary in Python is an unordered, mutable collection of key-value pairs. It provides an efficient way to store and retrieve data, where each value is associated with a unique key.

Key Characteristics:

- **Unordered:** Elements in a dictionary are not stored in a specific order. Access is based on keys, not indices.
- **Mutable:** You can modify the content of a dictionary by adding, updating, or removing key-value pairs.
- **Unique Keys:** Each key in a dictionary must be unique. However, values can be duplicated.

Create a dictionary:

```
# Syntax: {key1: value1, key2: value2, ...}
my_dict = {'name': 'John', 'age': 25, 'city': 'New York'}
```

Access values:

```
# Accessing values using keys
print(my_dict.get('name')
Output: John
```

Adding and updating entries:

```
# Adding a new key-value pair
my_dict['occupation'] = 'Engineer'

# Updating an existing value
my_dict['age'] = 26
```

Removing entries:

```
# Removing a specific key-value pair
del my_dict['city']

# Clearing all entries
my_dict.clear()
```

Dictionary methods:

```
# Get a list of keys
keys = my_dict.keys()

# Get a list of values
values = my_dict.values()

# Check if a key exists
if 'name' in my_dict:
    print('Key "name" exists!')
```

Iterating through a dictionary:

```
# Iterate through keys and values
for key, value in my_dict.items():
    print(f'{key}: {value}')
```

Tutorial 1: Cats and Dogs

A simple dictionary example with cats and dogs.

```
"""
    Name: cats_and_dogs.py
    Author:
    Created:
    Purpose: Create and use a dictionary
"""
# Define the key : value pairs of the dictionary
dictionary = {
    "dog": "has a tail and goes woof!",
    "cat": "says meow",
    "mouse": "is chased by cats"
}

# Prompt the user to enter a dictionary key
print("This dictionary contains values for dog, cat, or mouse.")
word = input("Enter a word (key): ")

# Use the key entered by the user to access the value
# .get("key", "default value") default value is used if key doesn't exist)
print(f"Key: {word} Value: {dictionary.get(word, 'Value does not exist')}")
print(f"A {word} {dictionary.get(word, 'Value does not exist')}")
```

Example run:

```
This dictionary contains values for dog, cat, or mouse.
Enter a word (key): dog
Key: dog Value: has a tail and goes woof!
A dog has a tail and goes woof!
```

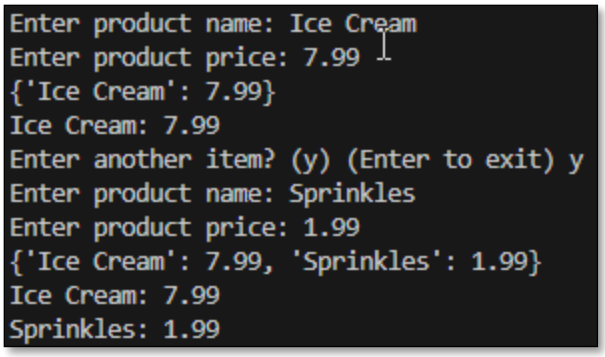
```
This dictionary contains values for dog, cat, or mouse.
Enter a word (key): key
Key: key Value: Value does not exist
A key Value does not exist
```

Tutorial 2: Add Items to a Dictionary

You can add items to a dictionary from user input.

```
"""
    Name: product_price_dictionary_2.py
    Author: William A Loring
    Created: 02/23/2022
    Purpose: Product name and price dictionary
"""

# Create empty dictionary
product_dict = {}

while True:
    # Get item from user
    product_name = input("Enter product name: ")
    product_price = float(input("Enter product price: "))

    # Insert item into dictionary
    product_dict[product_name] = product_price

    # Print the dictionary directly
    print(product_dict)

    # Print the dictionary in a nicer format
    # for each items key and value
    # loop through the dictionary
    for key, value in product_dict.items():
        print(f"{key}: {value}")

    choice = input("Enter another item? (y) (Enter to exit) ")
    if choice =="":
        break
```

Example run:



Tutorial 3: Pickle a Dictionary

Pickling in Python refers to the process of serializing objects, converting them into a byte stream. To pickle a dictionary, use the built in Python **pickle** module. Import it, open a file in binary mode, and use the **dump** function to serialize (convert to a byte stream) the dictionary.

Unpickling is the reverse process, where the byte stream is converted back into a Python object using the **load** function.

Example:

```
import pickle

# Pickling a dictionary
data = {'key': 'value'}
with open('filename.pkl', 'wb') as file:
    pickle.dump(data, file)

# Unpickling the dictionary
with open('filename.pkl', 'rb') as file:
    unpickled_data = pickle.load(file)

print(unpickled_data)
```

Update the previous tutorial to pickle and unpickle a dictionary.

```python
"""
    Name: product_dictionary_pickle.py
    Author: William A Loring
    Created: 10/08/2023
    Purpose: Pickle product and price dictionary
"""
import pickle
FILE_NAME = "product_dictionary.pkl"
# Create empty dictionary object
product_dict = {}

"""Unpickle the dictionary from file with pickle.load
    'with open' opens the file for access
    'r' read file
    'b' binary file type
"""
# Use try catch for exception if the file doesn't exist
try:
    with open(FILE_NAME, "rb") as file_handle:
        product_dict = pickle.load(file_handle)
    # When the program exits the 'with' block,
    # the file is closed: the file handle resource is released
    print("Load pickle dictionary")
    # Print the dictionary
    for product, price in product_dict.items():
        print(f"{product}: {price}")
except:
    pass
```

```
31  while True:
32      # Get item from user
33      product_name = input("Enter product name: ")
34      product_price = float(input("Enter product price: "))
35
36      # Insert item into dictionary using 'product_name' as the key
37      product_dict[product_name] = product_price
38
39      """Pickle the dictionary to a file with pickle.dump
40          'with open' opens the file for access
41          'w' write file
42          'b' binary file type
43      """
44      with open(FILE_NAME, "wb") as file_handle:
45          # Write list to file with binary protocol
46          pickle.dump(product_dict, file_handle)
47      # When the program exits the 'with' block,
48      # the file is closed: the file handle resource is released
49
50      print("Dump pickle dictionary")
51      # Print the dictionary
52      for product, price in product_dict.items():
53          print(f"{product}: {price}")
54
55      choice = input("Enter another item? (y) (Enter to exit) ")
56      if choice == "":
57          break
```

Example run:

```
Load pickle dictionary
Carrots: 2.99
beans: 4.5
corn: 3.4
Enter product name: radishes
Enter product price: 2.34
Dump pickle dictionary
Carrots: 2.99
beans: 4.5
corn: 3.4
radishes: 2.34
Enter another item? (y) (Enter to exit)
```

Assignment 1: Create Your Own Dictionary

Create your own dictionary with your own keys and values.

1. Dictionary Creation:
    a. Create an empty dictionary named **my_dict**
    b. Add at least three key-value pairs to the dictionary, representing different types of information (e.g., name, age, city).
2. Accessing and Printing Values:
    a. Print the entire dictionary.
    b. Print the value associated with one of the keys.
    c. Use the `get` method to retrieve and print each value.
3. Updating and Adding Elements:
    a. Modify the value of one existing key in the dictionary.
    b. Add a new key-value pair to the dictionary.
4. Iterating Through the Dictionary:
    a. Use a `for` loop to iterate through the keys and print them.
    b. Use another loop to iterate through the values and print them.
    c. Use another loop to iterate through the keys and the values.

Example run:

```
Entire Dictionary:
{'name': 'Alice', 'age': 25, 'city': 'Wonderland'}

Using 'get' method:
Alice
25
Wonderland

Modifying value of 'age':
{'name': 'Alice', 'age': 26, 'city': 'Wonderland'}

Adding a new key-value pair:
{'name': 'Alice', 'age': 26, 'city': 'Wonderland', 'gender': 'Female'}

Iterating through keys:
name
age
city
gender

Iterating through values:
Alice
26
Wonderland
Female

Iterate through the dictionary
name: Alice
age: 26
city: Wonderland
gender: Female
```

Assignment Submission

1. Attach the program files.
2. Attach screenshots showing the successful operation of the program.
3. Submit in Blackboard.