Dictionaries

In [3]:
```
# declaring a dictionary vairable
empty = {}          # empty dictionary
person = {"name":"Hrishikesh Devdikar"} #dictionary with one key-pair
customer = {
    "name" :  "Morty",
    "age"  :   22
}             # dicitonary with 2  pair key/value
print(empty, person , customer)
```

```
{} {'name': 'Hrishikesh Devdikar'} {'name': 'Morty', 'age': 22}
```

In [4]:
```
# accessing data from dictionary
print(person ["name"])
```

```
Hrishikesh Devdikar
```

In [5]:
```
# using the get method to access dictionary data
person={"name":"Harry Potter"}
print(person.get("name"))      #retrieves value of name key as before\
print(person.get("age","Age is not Available")) #secure way of retreiving info as error
```

```
Harry Potter
Age is not Available
```

In [6]:
```
# storing a list within a dictionary and accessing it
data = {"sports":["Chess","Football","Hockey","Tennis"]}
print(data["sports"][0])# first access key then the index
```

```
Chess
```

In [7]:
```
# improper storing a list within a dictionary
sports = ["Chess","Football","Hockey","Tennis"]
#sports_dict = dict(sports)          # will produce error, no key
sports_dict = dict({"sports":sports})
print(sports_dict)
```

```
{'sports': ['Chess', 'Football', 'Hockey', 'Tennis']}
```

In [8]:
```
# storing a dictionary within a list and accessing it
data = ["Rick","Dalton",{"name":"Kristen"}]
print(data)
print(data[2])      #dicitonary is index 2
print(data[2]["name"])
```

```
['Rick', 'Dalton', {'name': 'Kristen'}]
{'name': 'Kristen'}
Kristen
```

In [9]:
```
# storing a dictionary within a dictionary and accessing it
data = {
    "team":"Boston Red Sox",
    "wins":{"2018": 108, "2017": 93}
}
print(data.get("wins"))
print(data["wins"].get("2018"))
# alternative: print(data["wins"]["2018"])
```

```
{'2018': 108, '2017': 93}
108
```

In [10]:
```
"""
Exercise 1:Ask the user for their name and age, and then create a dictionary
with those key-value pairs. Output the dictionary once created.
"""
name = input("Enter Name: ")
age = input("Enter Age: ")
data ={
    "Name":name,
    "Age":age
}
print(data)
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
<ipython-input-10-9e248ecfdf03> in <module>
      3 with those key-value pairs. Output the dictionary once created.
      4 """
----> 5 name = input("Enter Name: ")
      6 age = input("Enter Age: ")
      7 data ={

~\anaconda3\lib\site-packages\ipykernel\kernelbase.py in raw_input(self, prompt)
    858                 "raw_input was called, but this frontend does not support input
 requests."
    859             )
--> 860         return self._input_request(str(prompt),
    861             self._parent_ident,
    862             self._parent_header,

~\anaconda3\lib\site-packages\ipykernel\kernelbase.py in _input_request(self, prompt, id
ent, parent, password)
    902             except KeyboardInterrupt:
    903                 # re-raise KeyboardInterrupt, to truncate traceback
--> 904                 raise KeyboardInterrupt("Interrupted by user") from None
    905             except Exception as e:
    906                 self.log.warning("Invalid Message:", exc_info=True)

KeyboardInterrupt: Interrupted by user
```

In [ ]:
```
"""
Exercise 2:Output all the ingredients from the following list within
the “ingredients” key using a for loop:
>>> pizza = {
>>> 'ingredients': ['cheese', 'sausage', 'peppers']
>>> }

"""
pizza = { 'ingredients': ['cheese', 'sausage', 'peppers']  }
for i in pizza['ingredients']:
    print(i)
```

Working with Dictionaries

In [ ]:
```
# adding new key/value pair to a dictionary
car = {"year":2018}
car["color"] = "Blue"
print("Year: {} \t Color: {}".format(car["year"],car["color"]))
```

In [ ]:
```
# updating a value for a key/value pair that already exists
```

```
car = {"year":2018,"color":"blue"}
car["color"] = "Red"
print("Year: {} \t Color: {}".format(car["year"],car["color"]))
```

In [ ]:
```
# deleting a key/value pair from a dictionary
car ={"year":2020}
try:
    del car["year"]
    print(car)
except:
    print("That key does not exist")
```

In [ ]:
```
# looping over a dictionary via the keys
person = {"name": "James Bond","age":30}
for key in person.keys():
    print(key)
    print(person[key])
```

In [ ]:
```
# looping over a dictionary via the values
person = {"name": "James Bond","age":30}
for value in person.values( ):
    print(value)
```

In [ ]:
```
# looping over a dicitionary via the key/value pair
person = {"name": "James Bond","age":30}
for key, values in person.items():
    print(f"{key} {values}")
```

In [ ]:
```
"""
Exercise 3: Declare an empty dictionary. Ask the user for their name, address,
and number. Add that information to the dictionary and iterate over it to show
the user.
"""
empty = {}
name=input("Enter Name: ")
address = input("Enter Address")
number= input ("Enter mobile number: ")
empty["name"]= name
empty["address"]= address
empty["number"]= number
print(empty)
for k,v in empty.items():
    print(f"{k} {v}")
```

Tuples, Sets and Frozensets

In [ ]:
```
# declaring a tuple
t1 = ("hello",2,"hello")  # tuples are like list just immutable
t2 = True, 1          # can be declared with or without parenthesis
print(type(t1),type(t2))
```

In [ ]:
```
#declaring set
s1 =set([1,2,3,4,1]) # uses the set keyword and square brackets
s2 = {4,4,5}  # uses curcly brackets like dictionary but without key
print(type(s1),type(s2))
s1.add(5) #using the add method to add new items to a set
```

```
s1.remove(1)  #using the remove method to get rid of the value 1
print(s1)  #drops the second '1'  as set is unique
```

In [ ]:
```
#declaring a frozenset
fset = frozenset([1,2,3,4])
print(type(fset))
```

In [ ]:
```
"""
Exercise :Ask the user to input as many bank account numbers as they’d
like, and store them within a list initially. Once the user is done entering
information, convert the list to a frozenset and print it out.
"""
accounts = []
done = False

while not done:
    ans = input('Enter an account number or quit: ').lower()

    if ans == 'quit':
        done = True

        accounts = frozenset(accounts)
        for acc in accounts:
            print("Account Number: {}".format(acc))

    else:
        accounts.append(ans)
```

In [ ]:
```
"""
Exercise :Convert the following list into a set of unique values. Print it out
after to check there are no duplicates:
>>> nums = [3, 4, 3, 7, 10]
"""
nums = [3, 4, 3, 7, 10]
nums = set(nums)
print(nums)
```

Reading & Writing files

In [ ]:
```
# opening/creating and writing to a test file
f = open("test.txt","w+")  #open file in reading and writing mode
f.write("This is test")
f.close()
#reading from a text file
f = open("test.txt","r")
data = f.read()
f.close()
print(data)
```

In [ ]:
```
# opening/creating and writing to a csv file
import csv
with open("test.csv",mode="w",newline="")as f:
    writer = csv.writer(f,delimiter=",")
    writer.writerow(["Name","City"])
    writer.writerow(["Bames Jond","NYC"])
```

In [ ]:
```
# reading from csv file
```

```
with open("test.csv",mode='r') as f:
    reader = csv.reader(f,delimiter=",")
    for row in reader:
        print(row)
```

In [ ]:

```
"""
Exercise: Ask a user for their favorite number, and save it to a text file.
"""
ask =input("Enter the Number: ")
f = open("test.txt","w+")
f.write(ask)
f.close()
f = open("test.txt","r")
data = f.read()
f.close()
print(f"Your Number is: {data}")
```

In [ ]:

```
"""
Exercise: Using the dictionary of following data, save the
information to a csv file with the keys as the headers and the
values as the rows of data:
>>> Using the dictionary of following data, save the
information to a csv file with the keys as the headers and the
values as the rows of data:
>>> data = {
'name' : ['Dave', 'Dennis', 'Peter', 'Jess'],
'language': ['Python', 'C', 'Java', 'Python']
}
"""

data = {
    'name' : ['Dave', 'Dennis', 'Peter', "Jess"],
    'language' : ['Python', 'C', 'Java', 'Python']
}

import csv

with open('data.csv', mode='w', newline='') as f:
    writer = csv.writer(f, delimiter=',')
    writer.writerow(data.keys())

    for i in range(len(data['name'])):
        writer.writerow([data['name'][i], data['language'][i]])
```

Creating a User Database with CSV Files

In [ ]:

```
"""
1. Check to see if user is logged in.
a. If logged in, ask if they would like to log out/quit.
i. Either quit or log out user and restart.
b. Else, ask if they would like to log in/register/quit.
i. If log in, ask user for e-mail/password.
1. If correct, log user in and restart.
2. Else, display error and restart.
ii. If register, ask for e-mail/password/password2.
1. If passwords match, save user and restart.
2. Else, display error and restart.
iii. If quit, say thank you and exit program.
```

```
"""
```

In [15]:
```
# import all necessary packages to be used
import csv
from IPython.display import clear_output

# handle user changing password
def changePassword():
    '''
        This function must confirm email and pass, then read data and save to local lis
        because you cannot change a single value, you must save all data, then overwrit
        the entire file all together.
    '''

    email = input('Please confirm your e-mail: ')
    password = input('Please confirm your current password: ')

    emails = []
    passwords = []
    found = False

    with open('users.csv', mode='r') as f:
        reader = csv.reader(f, delimiter=',')

        for row in reader:
            if row == [email, password]:
                found = True
            elif row:
                emails.append(row[0])
                passwords.append(row[1])

    if found:
        new_pass = input('What would you like to change your password to? ')

        emails.append(email)
        passwords.append(new_pass)

        with open('users.csv', mode='w') as f:
            writer = csv.writer(f, delimiter=',')

            for i in range(len(emails)):
                writer.writerow([emails[i], passwords[i]])
    else:
        print('Sorry those credentials were incorrect.')

# handle user registration and writing to csv
def registerUser():
    with open('users.csv', mode='a', newline='') as f:
        writer = csv.writer(f, delimiter=',')

        print('To register, please enter your info:')
        email = input('E-mail: ')
        password = input('Password: ')
        password2 = input('Re-type password: ')

        clear_output()

        if password == password2:
            writer.writerow([email, password])
```

```python
            print('You are now registered!')
        else:
            print('Something went wrong. Try again.')

# ask for user info and return true to login
def loginUser():
    print('To login, please enter your info:')
    email = input('E-mail: ')
    password = input('Password: ')

    clear_output()

    with open('users.csv', mode='r') as f:
        reader = csv.reader(f, delimiter=',')

        for row in reader:
            if row == [email, password]:
                print('You are now logged in!')
                return True

    print('Something went wrong, try again.')
    return False

# variables for main loop
active = True
logged_in = False

# main loop
while active:
    if logged_in:
        print('1. Logout\n2. Change Password\n3. Quit')
    else:
        print('1. Login\n2. Register\n3. Quit')

    choice = input('What would you like to do? ').lower()

    clear_output()

    if choice == 'register' and logged_in == False:
        registerUser()
    elif choice == 'login' and logged_in == False:
        logged_in = loginUser()
    elif choice == 'quit':
        active = False
        print('Thanks for using our software!')
    elif choice == 'logout' and logged_in == True:
        logged_in = False
        print('You are now logged out.')
    elif choice == 'change password' and logged_in == True:
        changePassword()
    else:
        print('Sorry, please try again!')
```

```
Thanks for using our software!
```

In [17]:

```python
"""
Write a new program that will ask users what their favorite food is. Save the answers t

Favorite Food?              # of Votes
Turkey                           5
Salad                             3
```

```
"""
def saveFood():
    ans = input('What is your favorite food? ')

    with open('favorite_food.csv', mode='a', newline='') as f:
        writer = csv.writer(f, delimiter=',')

        writer.writerow([ans])

    print('Food added!')

def countFood():
    food_count = {}

    with open('favorite_food.csv', mode='r') as f:
        reader = csv.reader(f, delimiter=',')

        for row in reader:
            if row[0].lower() in food_count:
                food_count[row[0].lower()] += 1
            else:
                food_count[row[0].lower()] = 1

    return food_count

def main():
    while input('Would you like to add more? ').lower() != 'no':
        saveFood()

    clear_output()

    print('Here are the results so far...')

    food_count = countFood()

    for k, v in food_count.items():
        print("{}: {}".format(k, v))

main()
```

```
Here are the results so far...
pizza: 1
```