Deborah Barndt

ITMD 513 Open Source Programming

Professor Dr. Sam

Hw8

3-26-19

'''

Deborah Barndt

3-25-19

EmailDictionary.py

hw8: Name and Email Dictionary

This program will keep names and email addresses in a dictionary as key-value

pairs. It will display a menu that lets the user look up a person's email

address, add a new name and email address, change an existing email address,

and delete an existing name and email address. The program will bind the

dictionary and save it to a file when the user exits the program. Each time the

program starts, it will retrieve the dictionary from the file and unbind it.

The program will run continuously until the user selects the option exit from

the menu.

Written by Deborah Barndt.

'''

import pickle

# Function to check to for an existing file or dictionary.

def checkExist():

try:

# Open the file contacts.dat in read mode.

contacts = open('contacts.dat', 'rb')

# Unpickle the contacts in the dictionary.

contactDict = pickle.load(contacts)

# Close the contacts.dat file.

contacts.close()

# If no dictionary exists, ask user if they would like to create one.

if (contactDict == {}):

print('There are no contacts in your dictionary.')

create = input('Would you like to add a contact? ')

# If user types yes, go to addContact().

if ('yes' in create.lower()):

addContact()

else:

return

else:

return contactDict

# If there is a reading error or the end of the file has been reached, give response to user.

except EOFError:

print('There are no contacts in the dictionary.')

# Ask the user if they would like to create a contact.

create = input('Would you like to add a contact? ')

# If user types yes, go to addContact().

if ('yes' in create.lower()):

addContact()

else:

return

# If no file exists, give response to user.

except FileNotFoundError:

print('There are no contacts in the dictionary.')

# Ask the user if they would like to create a contact.

create = input('Would you like to add a contact? ')

# If user types yes, go to addContact().

if ('yes' in create.lower()):

addContact()

else:

return

# Function to find a contact in the dictionary.

def findContact():

print('\nFind Contact\n')

# Check to see if the file or dictionary exists.

contactDict = checkExist()

# If the dictionary does not exist, return to main menu.

if (contactDict == None):

return

else:

try:

# Ask user to enter the contact name, all, or cancel.

search = input('Enter the contact name, all, or cancel: ')

# If user types cancel, return to main menu.

if (search.lower() == 'cancel'):

return

elif (search.lower() == 'all'):

# Print an organized list of contacts in the dictionary.

for name, email in sorted(contactDict.items()):

print('Name: ', name, '\nEmail: ', email, '\n')

return

else:

print('Name: ', search, '\nEmail: ', contactDict[search])

return

# If invalid entry, respond to user.

except KeyError:

print('The contact does not exist in the dictionary.')

# Ask the user if they would like to check the list.

checkList = input('Would you like to check the list? ')

# If user types yes, return organized sorted list.

if ('yes' in checkList.lower()):

for name, email in sorted(contactDict.items()):

print('Name: ', name, '\nEmail: ', email, '\n')

else:

return

# Function to add a contact to the dictionary.

def addContact():

print('\nAdd Contact\n')

# Try to open the file.

try:

# Open the file contacts.dat in read mode.

contacts = open('contacts.dat', 'rb')

# Unpickle the contacts in the dictionary.

contactDict = pickle.load(contacts)

# Close the contacts.dat file.

contacts.close()

# Ask user to enter new contact name or cancel.

name = input('Enter new contact name or cancel: ')

# If user types cancel, return to main menu.

if (name.lower() == 'cancel'):

return

else:

# Ask user to enter new contact email address.

email = input('Enter new contact email address: ')

contactDict.update({name : email})

# Open the file contacts.dat in write mode.

contacts = open('contacts.dat', 'wb')

# Pickle the dictionary, and write it to the file.

pickle.dump(contactDict, contacts)

# Close the contacts.dat file.

contacts.close()

# If there is no file, then give response to user.

except:

print('There are no contacts in the dictionary.')

# Ask user to enter new contact name or cancel.

name = input('Enter new contact name or cancel: ')

# If user types cancel, return to main menu.

if (name.lower == 'cancel'):

return

else:

# Ask user to enter new contact email address.

email = input('Enter new contact email address: ')

contactDict = {}

contactDict.update({name : email})

# Open the file contacts.dat in write mode.

contacts = open('contacts.dat', 'wb')

# Pickle the dictionary, and write it to the file.

pickle.dump(contactDict, contacts)

# Close the contacts.dat file.

contacts.close()

return

# Function to edit a contact in the dictionary.

def editContact():

print('\nEdit Contact\n')

# Check to see if the file or dictionary exists.

contactDict = checkExist()

# If the dictionary does not exist, return to main menu.

if (contactDict == None):

return

else:

# Ask the user to enter a contact name to edit or cancel.

edit = input('Enter contact name or cancel: ')

# If user types cancel, return to main menu.

if (edit.lower() == 'cancel'):

return

else:

if (edit not in contactDict):

create = input('The contact is not in the dictionary. Would you like to create this new contact? ')

# If user types yes, ask user to enter new email address.

if ('yes' in create.lower()):

newEmail = input('Enter new email address: ')

contactDict[edit] = newEmail

print('Name: ', edit, '\nEmail: ', contactDict[edit])

# Open the file contacts.dat in write mode.

contacts = open('contacts.dat', 'wb')

# Pickle the dictionary, and write it to the file.

pickle.dump(contactDict, contacts)

# Clost the contacts.dat file.

contacts.close()

return

else:

return

# If contact exists, allow user to edit.

else:

# Ask the user for the new email address for contact.

newEmail = input('Enter new email or cancel: ')

# If user types cancel, return to main menu.

if (newEmail.lower() == 'cancel'):

return

else:

contactDict[edit] = newEmail

print('Contact email updated.\n')

print('Name: ', edit, '\nEmail: ', contactDict[edit])

# Open the file contacts.dat in write mode.

contacts = open('contacts.dat', 'wb')

# Pickle the dictionary, and write it to the file.

pickle.dump(contactDict, contacts)

# Close the contacts.dat file.

contacts.close()

return

# Function to delete a contact in the dictionary.

def deleteContact():

print('\nDelete Contact\n')

# Check to see if the file or dictionary exists.

contactDict = checkExist()

# If the dictionary does not exist, return to main menu.

if (contactDict == None):

return

else:

try:

delete = input('Enter contact name or cancel: ')

# If user types cancel, return to main menu.

if (delete.lower() == 'cancel'):

return

else:

# Delete the contact from the dictionary.

contactDict.pop(delete)

print('The contact ', delete, ' has been deleted from the dictionary.')

# Open the file contacts.dat in write mode.

contacts = open('contacts.dat', 'wb')

# Pickle the dictionary, and write it to the file.

pickle.dump(contactDict, contacts)

# Close the contacts.dat file.

contacts.close()

# If there are no contacts in the dictionary, give response to user.

if (contactDict == None):

print('There are no contacts in the dictionary.')

else:

return

# Exception for invalid input.

except KeyError:

print('The contact does not exist in the dictionary.')

# Ask the user if they would like to check the list.

checkList = input('Would you like to check the list? ')

# If user types yes, return organized sorted list.

if ('yes' in checkList.lower()):

for name, email in sorted(contactDict.items()):

print('Name: ', name, '\nEmail: ', email, '\n')

else:

return

# Function that contains the main menu to the dictionary.

def main():

option = ''

# Loop for the main menu to run continuously.

while option.lower() != 'exit':

print('----------------------------------------')

print(' Email Dictionary Main Menu')

print('----------------------------------------')

option = input('Please select an option from the menu by number or name:\n'

'1. Add New Contact\n'

'2. Find Contact\n'

'3. Edit Contact\n'

'4. Delete Contact\n'

'5. Exit\n\n'

'Select Option: ')

# If statement for each of the options on the menu.

if (option.lower() in '1. add new contact'):

addContact()

elif (option.lower() in '2. find contact'):

findContact()

elif (option.lower() in '3. edit contact'):

editContact()

elif (option.lower() in '4. delete contact'):

deleteContact()

elif (option.lower() in '5. exit'):

break

else:

print('Your input is invalid. Please enter a number or name from the menu.')

# Call main function to begin program.

main()

Output Result:























