CS-UY 1114 LAB 12, Spring 2018

In this lab, you will learn about a data structure called Dictionary, specifically, how to:

- Create a dictionary
- Add key and value to dictionary
- Delete a key from dictionary
- Check if a key exist in the dictionary

Part 1: Pen and Paper Problem

```
def f1 (my_dict):
      temp = 0
      for key in my_dict:
            temp = temp + my_dict[key]
      return temp

def f2 (my_dict):
      temp = ‘’
      for key in my_dict:
            if temp < key:
                  temp = key
      return temp

def f3 (my_dict, k, v):
      if k in my_dict:
            my_dict[k] = v

def main():
      a_dict = {‘bill’: 1, ‘rich’: 2, ‘fred’: 10, ‘walter’: 20}

      print(f1(a_dict)) # Line 1
      print(f2(a_dict)) # Line 2
      print(None == f3(a_dict, ‘bill’, -1)) # Line 3
      print(a_dict) # Line 4

main()
```

a. What output is produced by Line 1 of the program?
b. What output is produced by Line 2 of the program?
c. What output is produced by Line 3 of the program?
d. What output is produced by Line 4 of the program?

Part 2: Programming Problem

Problem 1: Phone Book

Implement simple phonebook operations. Phonebook is a dictionary with names as keys and phone numbers as values. A valid phone number is a string with 10 digits.

For example: ‘2014567890’ is a valid phone number. But ‘201a45b789’ or ‘20145678’ are both invalid.

Your program should have the following functions:

- A function add_entry(phonebook, name, phone number) that adds an entry to a phonebook. This function should take a dictionary as phonebook, a string name and a string phone_number as arguments. It then adds an entry with key name and value phone_number to phonebook given that the following is true:

    - An entry in phonebook with key name does not exist
    - Phone_number is a valid phone number

  Your function should display an error message if the entry cannot be added.
  You might want to write a helper function to check if phone_number is valid

- A function lookup(phonebook, name) looks up phonebook. This function should return the phone number associated with name in phonebook. If the input name is not found in phonebook, display an error message that indicates so.

- A function print_all(phonebook) that prints all the entries in phonebook. Your function should print both the name and the phone number.

- A main function that opens ‘Lab 13 - phonebook.txt’ in which each line has name and phone number. The names are in the form ‘Last, Fist’ (Note: some of the phone numbers could be in a bad format and some of the names could be duplicated). Use your function add_entry(phonebook, name, phone number) to construct a phonebook from entries in ‘Lab 13 - phonebook.txt’

Example Interaction:

```
>>> phonebook = main()
21244445325 is not a valid phone number
40813263463 is not a valid phone number
Evan Gallagher already exist in the phone book
John Sterling already exist in the phone book
>>> add_entry(phonebook, 'Preston Moore', '64699736000')
64699736000 is not a valid phone number
>>> add_entry(phonebook, 'Preston Moore', '6469973600')
>>> add_entry(phonebook, 'Preston Moore', '6469973600')
Preston Moore already exist in the phone book
>>> lookup(phonebook, 'Larry Page')
'6508397347'
>>> lookup(phonebook, 'Linda Sellie')
Linda Sellie's phone number is not an recorded in the phonebook
>>> print_all(phonebook)
Preston Moore: 6469973600
Ken Thompson: 6462326463
Evan Gallagher: 7188482476
Itay Tal: 6501111234
Edsger Dijkstra: 7184447323
Sergey Brin: 6508397348
Gordon Moore: 4081367358
Yukihiro Matsumoto: 3476207830
Donald Knuth: 4085627456
John McCarthy: 4081857358
Dennis Ritchie: 4082468568
John Sterling: 6461137548
Larry Page: 6508397347
David Packard: 4159034382
Alan Turing: 1234567890
Phyllis Frankl: 4151234567
Stephen Wolfram: 3107482247
James Gosling: 8185362353
William Hewlett: 4151235346
Larry Wall: 2462467347
Guido van Rossum: 4157391366
```

Problem 2: Bank accounts

Design a class BankAcount

BankAccount should have the following components:

- Account Number
- Balance

BankAccount should have the following functionalities:

- Deposit
- Withdraw
- Show balance (Represent the bank account in the form of a string)

Example Interaction:

```
>>> my_account = BankAccount('BOA123', 1000)
>>> print(my_account)
Current balance is: $1000
>>> my_account.deposit(100)
>>> print(my_account)
Current balance is: $1100
>>> my_account.withdraw(500)
>>> print(my_account)
Current balance is: $600
>>> my_account.withdraw(999999)
Insufficient balance. The current balance remained as $600
```

Problem 3: Library

Design a class Library and class Patron

Patron should have the following characteristics:

- Name
- Library account number, composed of 5 digits. Must be random and unique. Library account number cannot be the same as other patrons. Patron will not have a library account number until the library adds the patron into its system.
- List of books that the patron borrowed

Library should have the following characteristics:

- Name
- Location
- List of patrons
- List of available books

The library should be able to:

- Add a patron to the library. When the patron is added, a library account number is assigned to the patron. You might want to delegate the task of creating a library account number to another function.
- Lend out books. Make sure the person to lend to is a patron of the library. Also make sure the book is available. Once the book is lent to the patron, the book is not available in the library's book list.
- Add books to its list of available books

Example Interaction:

```
>>> bk_library = Library("Brooklyn Public Library", "6 Metrotech Center")
>>> bk_library.add_book("Ender's Game")
>>> bk_library.add_book("Ender's Shadow")
>>> bk_library.add_book("Shadow of the Hegemon")
>>> bk_library.add_book("Ready Player One")
>>> bk_library.add_book("The Outsiders")
>>> bk_library.add_book("Flowers for Algernon")
>>> bob = Patron('Bob')
>>> print(bob.library_account_number)
None
>>> bk_library.add_patron(bob)
>>> print(bob.library_account_number)
85399
>>> print(bk_library)
Name: Brooklyn Public Library
Location: 6 Metrotech Center
```

```
Available Books:
Ender's Game
Ender's Shadow
Shadow of the Hegemon
Ready Player One
The Outsiders
Flowers for Algernon

Library Patron Information:
Name: Bob
Library Account Number: 85399

Borrowed Books:

>>> shuyu = Patron('Shuyu')
>>> bk_library.lend(shuyu, "Flowers for Algernon")
Shuyu is not a patron of the Brooklyn Public Library
>>> bk_library.add_patron(shuyu)
>>> bk_library.lend(shuyu, "Old Man’s War")
Old Man’s War is not available
>>> bk_library.lend(shuyu, "Ender's Shadow")
>>> bk_library.lend(bob, "Ender's Shadow")
Ender's Shadow is not available
>>> print(bk_library)
Name: Brooklyn Public Library
Location: 6 Metrotech Center

Available Books:
Ender's Game
Shadow of the Hegemon
Ready Player One
The Outsiders
Flowers for Algernon

Library Patron Information:
Name: Bob
Library Account Number: 85399

Borrowed Books:

Name: Shuyu
Library Account Number: 29752
```

Borrowed Books:
Ender's Shadow