

Programming Assignment:

Contacts Manager

Problem Statement:

Create a program that helps user manage contacts including listing all the contacts, looking up a contact by name, viewing given field for all the contacts, deleting and adding a contact. See the sample runs below.

The program should use a dictionary of dictionaries to store the contacts info where the key of the outer dictionary is the contact name and the key of the nested directory is the field name. Use the starter file `contactsManager-starter.py` where the dictionary of dictionaries is created for you at the beginning of the main function.

Complete the starter file by implementing the contacts manager. Specifically, write a program that

1. Welcomes the user
2. Prints the available commands:

COMMAND MENU

list - Display all contacts (*Displays an alphabetically sorted list of the names of all the players.*)

view - View a contact (*this will first prompt the user for the name of the contact user wants to view, then check if the name exists in the contacts, if so, prints all the fields for that contact in alphabetically sorted fashion. If the name is not found in the dictionary, prints "No contact found for that name." Make this command case insensitive. See sample runs below.*)

add - Add a contact (*this will prompt the user for the contact name, phone, company and address fields and create a new entry in the dictionary for that name. See sample runs below.*)

del - Delete a contact (*this will prompt the user for the name of the contact to be deleted and then delete the corresponding contact from the dictionary. Make this command case insensitive. See sample runs below*)

field – View field for all (*this will prompt the user for the name of the field to be viewed. Check that there is at least one contact with that field. If so, prints the field for all the contacts with that field specified. Otherwise prints "Field not*

found in any contact” No need to make this command case insensitive. Assume user enters the field name exactly as found in the data. See sample runs below.)

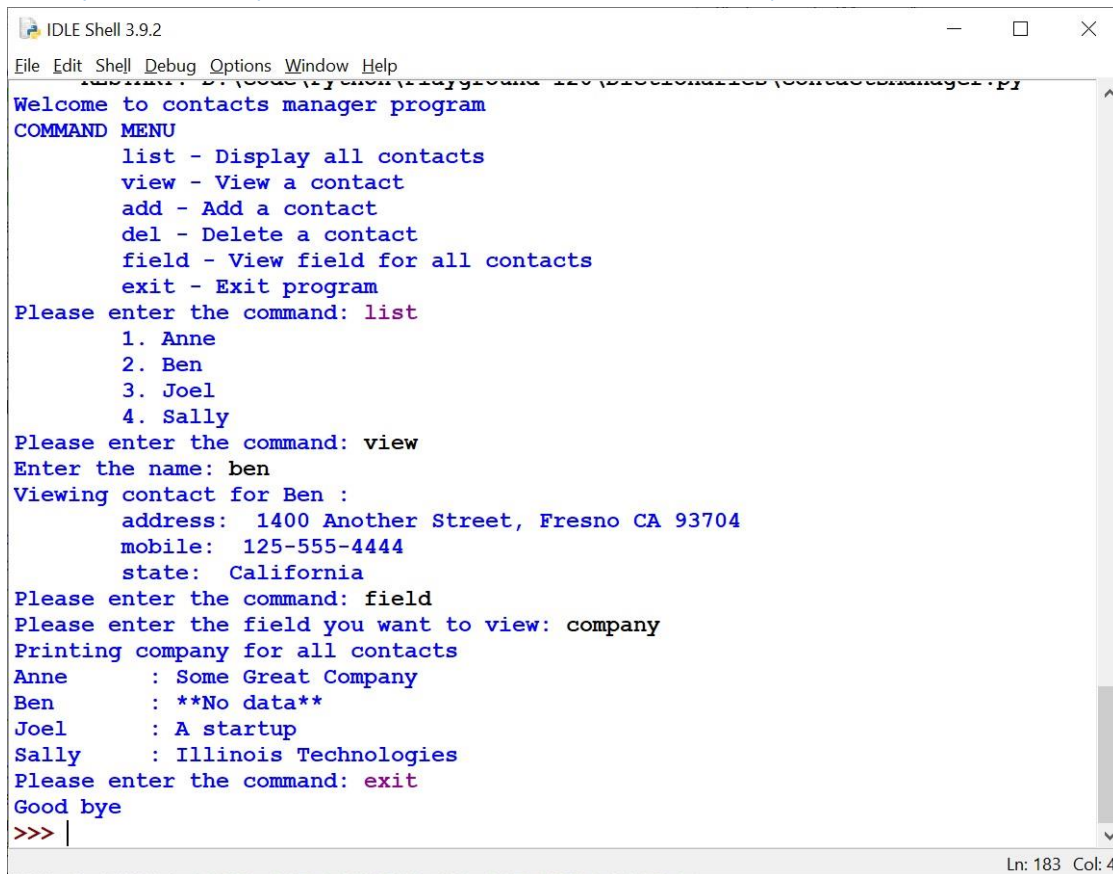
exit - Exit program

3. Then repeatedly prompts the user to enter the command and acts on the command as per the specification above. Program handles the case of user entering an invalid command.
4. When user enters “exit” command, program ends.

See the sample runs below. Submit the program in a file with a name of the form first_last_contactManager.py. For example, Elvis Presley will name the file elvis_presley_contactManager.py. Make sure to add a block-comment at the start of the file that lists assignment title, class name, date, your name, and assignment description. E.g.

```
#Assignment: contactManager.py (name of assignment)
#Class: PROG 120
#Date: (current date)
#Author: Elvis Presley (use your name)
#Description: Program to manage a list of contacts (Purpose of the
assignment)
```

Sample Run 1 (list, view and field commands):



```
IDLE Shell 3.9.2
File Edit Shell Debug Options Window Help
Welcome to contacts manager program
COMMAND MENU
    list - Display all contacts
    view - View a contact
    add - Add a contact
    del - Delete a contact
    field - View field for all contacts
    exit - Exit program
Please enter the command: list
    1. Anne
    2. Ben
    3. Joel
    4. Sally
Please enter the command: view
Enter the name: ben
Viewing contact for Ben :
    address: 1400 Another Street, Fresno CA 93704
    mobile: 125-555-4444
    state: California
Please enter the command: field
Please enter the field you want to view: company
Printing company for all contacts
Anne      : Some Great Company
Ben       : **No data**
Joel      : A startup
Sally     : Illinois Technologies
Please enter the command: exit
Good bye
>>> |
```

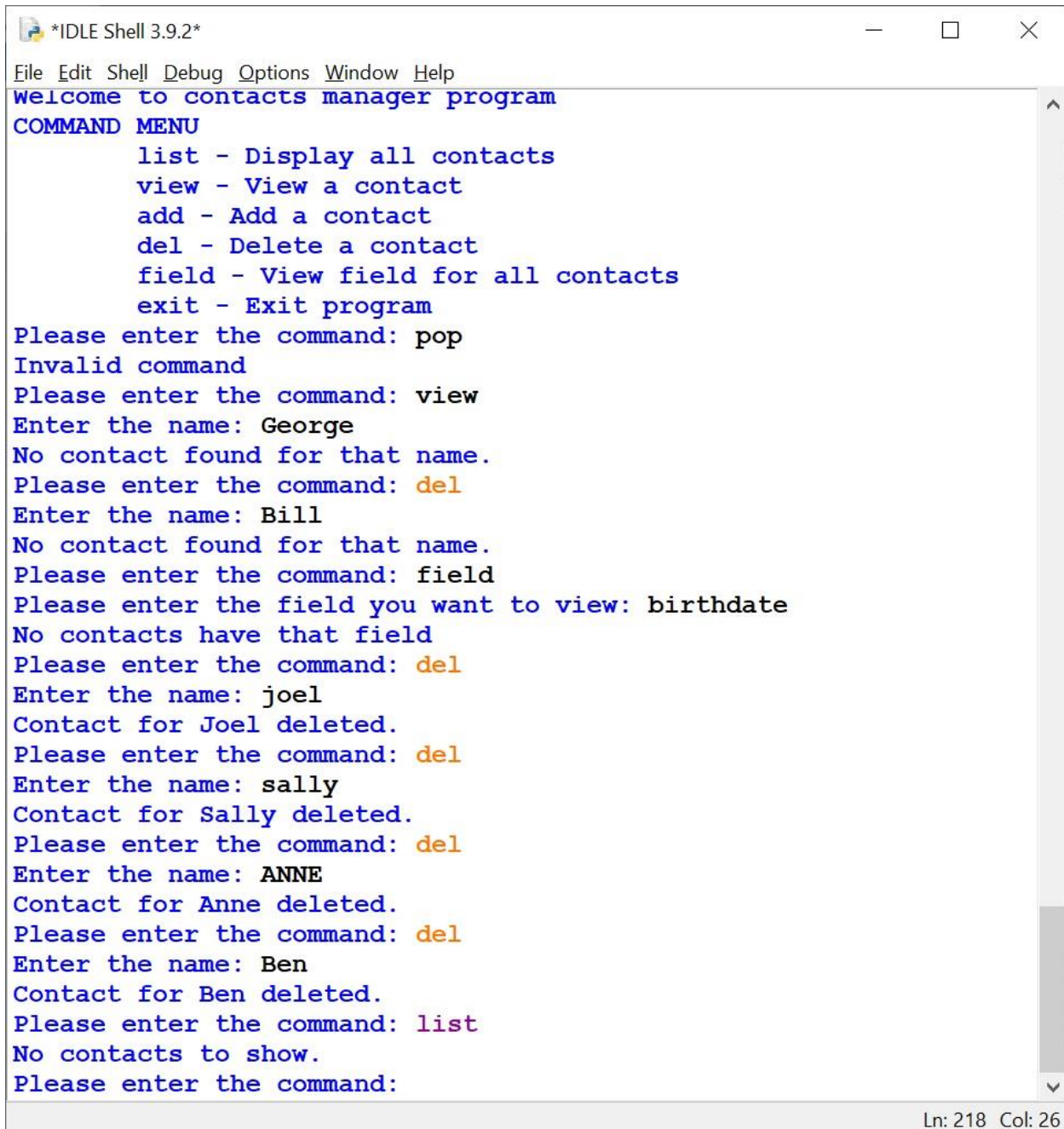
Ln: 183 Col: 4

Sample Run 2 (add and delete commands):

```
Python 3.8.1 Shell
File Edit Shell Debug Options Window Help
RESTART: D:\code\python\Playground_120\dictonaries\contactsmanger.py
Welcome to contacts manager program
COMMAND MENU
    list - Display all contacts
    view - View a contact
    add - Add a contact
    del - Delete a contact
    field - View field for all contacts
    exit - Exit program
Please enter the command: del
Enter the name: Sally
Contact for Sally deleted.
Please enter the command: add
Enter the name for the new contact: Harry
Enter the address for the new contact: 4 Privet Drive, Whinging, Surrey
Enter the mobile number for the new contact: 123-456-7890
Enter the company for the new contact: Hogwarts School
Please enter the command: list
    1. Anne
    2. Ben
    3. Harry
    4. Joel
Please enter the command: view
Enter the name: Harry
Viewing contact for Harry :
    address: 4 Privet Drive, Whinging, Surrey
    company: Hogwarts School
    mobile: 123-456-7890
Please enter the command: exit
Good bye
>>> |
```

Ln: 267 Col: 4

Sample Run 3 (boundary cases):



```
*IDLE Shell 3.9.2*
File Edit Shell Debug Options Window Help
welcome to contacts manager program
COMMAND MENU
    list - Display all contacts
    view - View a contact
    add - Add a contact
    del - Delete a contact
    field - View field for all contacts
    exit - Exit program
Please enter the command: pop
Invalid command
Please enter the command: view
Enter the name: George
No contact found for that name.
Please enter the command: del
Enter the name: Bill
No contact found for that name.
Please enter the command: field
Please enter the field you want to view: birthdate
No contacts have that field
Please enter the command: del
Enter the name: joel
Contact for Joel deleted.
Please enter the command: del
Enter the name: sally
Contact for Sally deleted.
Please enter the command: del
Enter the name: ANNE
Contact for Anne deleted.
Please enter the command: del
Enter the name: Ben
Contact for Ben deleted.
Please enter the command: list
No contacts to show.
Please enter the command:
```

Ln: 218 Col: 26

Program Requirements:

1. The program shouldn't assume that the dictionary was created with alphabetically sorted key values or alphabetically sorted fields. The program should work with any data correctly.
2. Define functions for each of the commands (except for the "exit" command) that takes a dictionary of dictionaries containing the contacts data as an argument and acts on the command correctly. Create additional helper functions to reduce code duplication.

3. Make sure you handle boundary cases:
 - a. Invalid command: print "Invalid command" and prompt again for the next command
 - b. Contact data is empty: list command should print "No contacts to show." If the contacts data is empty.
 - c. Make both the view and the del commands case-insensitive for the name user enters. E.g. when the dictionary data has a contact with the name "Anne", both the view and del commands should recognize user input of "anne", "ANNE" or "Anne" (or any variation) and act correctly.
 - d. Name not found in the contacts data: both the view and del commands should print "No contact found for that name."
 - e. Field not found: field command should print "No contact found with that field" if none of the contacts in the data has that field specified.
 - f. For all the commands (other than the add command), make your code work with generic fields and not with just some hard-coded ones.