COMP 125, Fall 2021
Homework Assignment 3

Due: 23:59, December 22 (Wednesday), 2021

In this homework assignment, you are supposed to implement an office directory that contains phone and room numbers of multiple employees. For each employee, you will store a record, in which you keep

1. The name (string)
2. The phone number (integer), and
3. The room number (string).

This information will be added to the office directory when an employee is hired and remains unchanged until the employee is fired or quits the job. Thus, in your implementation, you **MUST** define a tuple to keep information of each employee and you **MUST** define a list of tuples to keep all employees.

Your implementation must have at least the following four functions whose details are given below:

1. Add an employee
2. Remove an employee
3. Display all employees
4. Show information about an employee

ADD AN EMPLOYEE

This function adds an employee to the office directory. The name, the phone number, and the room number of the employee are specified as parameters. In this office directory, names are unique. The names are also case insensitive; that is, "cigdem gunduz" and "CIGdem guNduz" should be considered as the same. Thus, if the user attempts to add an employee with an already existing name, the function does not add the employee to the directory and displays a warning message. Otherwise, it performs the action. (See the example test program and its corresponding output given below to better understand how this function should work and the format of the warning message.)

For this function, you **MUST** use the following header.

```
# This function takes a list of employees together with the name, phone number,
# and room number of an employee to be added. If the name does not exist in the
# list, the function adds a tuple for the employee to the end of the list.
# Otherwise, if the name already exists in the list, the function does not add
# any tuple to the list and displays a warning message (for the message format,
# see the output example of the test program given below). You may assume that
# the arguments are always valid. That is, the name and the room number are
# always strings and the phone number is always a positive integer.
def add_employee(employee_list, name, phone, room):
    # Write your code here
    # ...
```

REMOVE AN EMPLOYEE

This function removes an employee from the office directory. The employee name is specified as a parameter. If there is no employee with the specified name, the function does not remove any employee from the office directory and displays a warning message. Otherwise, it performs the action. (See the example test program and its corresponding output given below to better understand how this function should work and the format of the warning message.)

For this function, you **MUST** use the following header.

```
# This function takes a list of employees together with the name of an employee
# to be removed. If the name exists in the list, the function removes the
# corresponding tuple from the list. Otherwise, if the name does not exist in
# the list, the function does not remove any tuple from the list and displays
# a warning message (for the message format, see the output example of the test
# program given below). You may assume that the name argument is always a string.
def remove_employee(employee_list, name):
    # Write your code here
    # ...
```

DISPLAY ALL EMPLOYEES

This function lists all employees already found in the office directory. The output should be in the following format. If there are no employees in the directory, this function displays `--EMPTY--` (See the output example of the test program given below to better understand the format.)

```
Name, office number, room number        (for the 1st employee)
Name, office number, room number        (for the 2nd employee)
. . .
```

For this function, you **MUST** use the following header.

```
# This function takes a list of employees and displays all employee tuples
# found in the list according to the required output format. If there exist
# no tuples in the list, this function displays --EMPTY-- (see the output
# example of the test program given below).
def display_all(employee_list):
    # Write your code here
    # ...
```

SHOW AN EMPLOYEE

This function displays all information about an employee whose name is specified as a parameter. The output should be in the following format. If the employee with the specified name does not exist in the office directory, the function displays a warning message. (See the example test program and its corresponding output given below to better understand how this function should work and the format of the warning message.)

```
Name: name of the employee
Office: room number of the employee
Tel: phone number of the employee
```

For this function, you **MUST** use the following header.

```
# This function takes a list of employees together with the name of an employee
# whose information will be showed. If the name exists in the list, the function
# displays the information stored in the corresponding tuple according to the
# required output format. If the name does not exist in the list, it displays
# a warning message (for the message format, see the output example of the test
# program given below). You may assume that the name argument is always a string
def show_employee(employee_list, name):
    # Write your code here
    # ...
```

WHAT TO SUBMIT?

Unlike the previous ones, this homework assignment asks you to submit only one file whose name must be **`office.py`**. This file should contain at least four functions whose details are given above. Remember that the headers of these functions must be

```
def add_employee(employee_list, name, phone, room):
def remove_employee(employee_list, name):
def display_all(employee_list):
def show_employee(employee_list, name):
```

Your file may contain other auxiliary functions, which might be called from the aforementioned four functions. However, it SHOULD NOT contain a function whose name is **`main()`**. Additionally, this file SHOULD NOT have any statements written outside a function definition. We will test your functions writing our own main function, similar to the one given below, and calling this main function to start the program.

*****IMPORTANT: You MUST use the given function headers as they are. We will use these headers to call your functions and test them. You are NOT allowed to modify these function headers. If you modify them, you will get no points from the corresponding part since we cannot call your functions properly, and thus, we cannot test and grade them.**

Additionally, if your file contains a function called `main()` or if it has a statement outside any function definition, you may lose a considerable number of points.

Note that although you will not submit a main function, you should, of course, write a main function in another py file and test your own functions (but do not submit this additional py file).

Like the previous homework assignments, the correctness of your program will affect your grade. However, in addition to this, the following points are important to get full credit.

- You MUST use meaningful names for the variables.
- You MUST write explanatory and clearly understandable comments.
- At the beginning of each file, you MUST write your name, your id, and your section as comments. After these, you MUST provide a very brief description about what the program does as additional comments.

For this assignment, you are allowed to use the codes given in the recommended textbooks and/or our lecture slides. However, you ARE NOT ALLOWED to use any codes from other sources (including the codes given in other textbooks, found on the Internet, belonging to your classmates, etc.). **Do not forget that plagiarism and cheating will be heavily punished. Please do the homework yourself.**

Before submitting your homework, please be sure that you carefully read all these explanations and understood them very well.

Here is an example test program, containing an example main function, and the corresponding output. We will use a similar program to test your functions so make sure that the name of your submitted file is **`office.py`** and you use the functions headers given above.

```
import office

def main():
    L = []
    office.display_all(L)
    print('---------------------------------')

    office.add_employee(L, 'Cigdem Gunduz', 1853, 'ENG108A')
    office.add_employee(L, 'Serkan Cil', 1111, 'GRT 02')
    office.add_employee(L, 'Mehmet SAYAR', 2222, 'ENG118A')
    office.add_employee(L, 'Seher Ozcelik', 3333, 'ENG123')
    office.add_employee(L, 'Ulkem Kasapoglu', 3333, 'ENG123')
    office.display_all(L)
    print('---------------------------------')

    office.add_employee(L, 'CIGdem GuNduz', 4444, 'SNA 52')
    office.add_employee(L, 'Seher Ozcelik', 5555, 'SOS B7')
    print('---------------------------------')
    office.display_all(L)
    print('---------------------------------')

    office.show_employee(L, 'Cigdem GUNDUZ')
    print('---------------------------------')
    office.show_employee(L, 'Ulkem Kasapoglu')
    print('---------------------------------')
    office.show_employee(L, 'Mehmet Sayar')
    print('---------------------------------')
    office.show_employee(L, 'Merve KUNAK')
    print('---------------------------------')

    office.remove_employee(L, 'Mehmet Sayar')
    office.display_all(L)
    print('---------------------------------')
    office.show_employee(L, 'Mehmet Sayar')
    print('---------------------------------')

    office.remove_employee(L, 'Merve KUNAK')
    print('---------------------------------')

    office.remove_employee(L, 'Cigdem Gunduz')
    office.remove_employee(L, 'Seher Ozcelik')
    office.remove_employee(L, 'Ulkem KASapoglu')
    office.display_all(L)
    print('---------------------------------')

    office.remove_employee(L, 'Serkan Cil')
    office.display_all(L)
    print('---------------------------------')


main()
```

The output of this test program should be:

```
--EMPTY--
---------------------------------
Cigdem Gunduz , ENG108A , 1853
Serkan Cil , GRT 02 , 1111
Mehmet SAYAR , ENG118A , 2222
Seher Ozcelik , ENG123 , 3333
Ulkem Kasapoglu , ENG123 , 3333
---------------------------------
(error in add)  CIGdem GuNduz already exists
(error in add)  Seher Ozcelik already exists
---------------------------------
Cigdem Gunduz , ENG108A , 1853
Serkan Cil , GRT 02 , 1111
Mehmet SAYAR , ENG118A , 2222
Seher Ozcelik , ENG123 , 3333
Ulkem Kasapoglu , ENG123 , 3333
---------------------------------
Name: Cigdem Gunduz
Office: ENG108A
Tel: 1853
---------------------------------
Name: Ulkem Kasapoglu
Office: ENG123
Tel: 3333
---------------------------------
Name: Mehmet SAYAR
Office: ENG118A
Tel: 2222
---------------------------------
(error in show)  Merve KUNAK is not found
---------------------------------
Cigdem Gunduz , ENG108A , 1853
Serkan Cil , GRT 02 , 1111
Seher Ozcelik , ENG123 , 3333
Ulkem Kasapoglu , ENG123 , 3333
---------------------------------
(error in show)  Mehmet Sayar is not found
---------------------------------
(error in remove)  Merve KUNAK is not found
---------------------------------
Serkan Cil , GRT 02 , 1111
---------------------------------
--EMPTY--
---------------------------------
```