CP1850 – Assignment 2

Handed Out: 03 Nov 2023
Due Date: 17 Nov 2023

Objectives

This problem set will build on the skills acquired in the first few chapters and will test you extensively on the usage of collections to form relevant data structure to solve various problems.

Collaboration

You may work with other students. However, each student should write up and hand in his or her assignment separately. Be sure to indicate with whom you have worked in the comments of your submission.

Submission

You will create one python file for each question. You need to ensure that suitable comments and **function definitions** are present in the code. Please ensure you submit your assignment to the drop box that is provided on Brightspace **before the due date.**

Late Submissions:

Penalties for late submissions is as follows:

1. Up to 24 hours after the due date: Flat 25% penalty
2. Beyond 24 hours up to 48 hours: Flat 50% penalty
3. Beyond 48 hours: No points for assignment

If for some reason you are unable to submit the assignment on time, please reach out to me to make alternate arrangements – which will be handled based on the merits of the case. **No extensions will be provided.** Please ensure you stay on top of the requirements and manage your time well.

Problem 1:

Create a program that checks whether a number is a prime number and displays its factors if it is not a prime number.

Console

```
Prime Number Checker

Please enter an integer between 1 and 5000: 5
5 is a prime number.

Try again? (y/n): y

Please enter an integer between 1 and 5000: 6
6 is NOT a prime number.
It has 4 factors: 1 2 3 6

Try again? (y/n): y

Please enter an integer between 1 and 5000: 200
200 is NOT a prime number.
It has 12 factors: 1 2 4 5 8 10 20 25 40 50 100 200

Try again? (y/n): n

Bye!
```

Specifications

- A prime number is divisible by two factors (1 and itself). For example, 7 is a prime number because it is only divisible by 1 and 7.
- If the user enters an integer that's not between 1 and 5000, the program should display an error message.
- If the number is a prime number, the program should display a message to that effect.
- If the number is not a prime number, the program should display a message to that effect. Then, it should display the number of factors for the number and a list of those factors.
- Store the factors for each number in a list.
- Use functions to organize the code for this program.

Problem 2:

Create a program that a user can use to manage the primary email address and phone number for a contact.

Console

```
Contact Manager

COMMAND MENU
list - Display all contacts
view - View a contact
add  - Add a contact
del  - Delete a contact
exit - Exit program

Command: list
1. Guido van Rossum
2. Eric Idle

Command: view
Number: 2
Name: Eric Idle
Email: eric@ericidle.com
Phone: +44 20 7946 0958

Command: add
Name: Mike Murach
Email: mike@murach.com
Phone: 559-123-4567
Mike Murach was added.

Command: del
Number: 1
Guido van Rossum was deleted.

Command: list
1. Eric Idle
2. Mike Murach

Command: exit
Bye!
```

Specifications

- Use a list of lists to store the data for the contacts. Provide starting data for two or more contacts.
- For the view and del commands, display an error message if the user enters an invalid contact number.
- When you exit the program, all changes that you made to the contact list are lost.

Problem 3:

Create a two-player Tic Tac Toe game.

Console

```
Welcome to Tic Tac Toe

+---+---+---+
|   |   |   |
+---+---+---+
|   |   |   |
+---+---+---+
|   |   |   |
+---+---+---+

X's turn
Pick a row (1, 2, 3): 1
Pick a column (1, 2, 3): 1

+---+---+---+
| X |   |   |
+---+---+---+
|   |   |   |
+---+---+---+
|   |   |   |
+---+---+---+

O's turn
Pick a row (1, 2, 3): 1
Pick a column (1, 2, 3): 2

...
...

X's turn
Pick a row (1, 2, 3): 3
Pick a column (1, 2, 3): 3

+---+---+---+
| X | O | O |
+---+---+---+
|   | X |   |
+---+---+---+
|   |   | X |
+---+---+---+

X wins!

Game over!
```

Specifications

- Use a list of lists to store the Tic Tac Toe grid.
- If the user picks an invalid row or column or a cell that's already taken, display an error message.
- If there is a winner, the game should display an appropriate message and end. Otherwise, it should continue until the grid is full and end in a tie.

Problem 4:

Create a program that keeps track of the items that a wizard is carrying.

Console

```
The Wizard Inventory program

COMMAND MENU
walk - Walk down the path
show - Show all items
drop - Drop an item
exit - Exit program

Command: walk
While walking down a path, you see a scroll of uncursing.
Do you want to grab it? (y/n): y
You picked up a scroll of uncursing.

Command: walk
While walking down a path, you see an unknown potion.
Do you want to grab it? (y/n): y
You can't carry any more items. Drop something first.

Command: show
1. a wooden staff
2. a scroll of invisibility
3. a crossbow
4. a scroll of uncursing

Command: drop
Number: 3
You dropped a crossbow.

Command: exit
Bye!
```

Specifications

- Your instructor should provide a text file named wizard_all_items.txt that contains a list of all the items that a wizard can carry.
- You should create another text file named wizard_inventory.txt to store the current items that the wizard is carrying.
- A wizard can only carry four items at a time, and those four items must be different.
- When the user selects the walk command, the program should read the items from the file, create a list of the items that aren't already in the wizard's inventory, randomly pick one of those items, and give the user the option to grab it. To create a list of the items that aren't already in the wizard's inventory, you can use a list comprehension as described in chapter 6.
- Make sure to update the inventory text file you created every time the user grabs or drops an item.
- For the drop command, display an error message if the user enters an invalid number for the item.

Problem 5:

Create a program that a user can use to manage the primary email address and phone number for a contact.

Console

```
Contact Manager

COMMAND MENU
list - Display all contacts
view - View a contact
add  - Add a contact
del  - Delete a contact
exit - Exit program

Command: list
1. Guido van Rossum
2. Eric Idle

Command: view
Number: 2
Name: Eric Idle
Email: eric@ericidle.com
Phone: +44 20 7946 0958

Command: add
Name: Mike Murach
Email: mike@murach.com
Phone: 559-123-4567
Mike Murach was added.

Command: list
1. Guido van Rossum
2. Eric Idle
3. Mike Murach

Command: exit
Bye!
```

Specifications

- Your instructor should provide a CSV file named contacts.csv.
- When the program starts, it should read the contacts from the CSV file.
- For the view and del commands, display an error message if the user enters an invalid contact number.
- When you add or delete a contact, the change should be saved to the CSV file immediately. That way, no changes are lost, even if the program crashes later.