파이썬 프로그래밍및실습

Developing a food recommendationfunction

Progress Report : 2

Date : 23/12/10
Name : 이나경
ID : 231135

1. Introduction

1) Background

Food is something you have to eat every day, but each time you choose a menu,it takes a lot of time to choose a satisfactory menu. This leads to a decrease inbreakfast preparation time, lunch breaks, and work hours while thinking aboutmeals. We need a program that automatically recommends food to solve theproblem of wasting time through eating concerns and to select a more satisfyingmenu.

2) Project goal

The goal is to recommend food menus based on food-specific ratings given bycustomers, reducing the time to think about food during meal times and allowingthem to select more satisfying menus.

3) Differences from existing programs

Existing programs recommend food lists that are already finished and cannot beedited. It is highly likely that foods that do not suit individual preferences willoften be recommended. We have a difference from the existing program in thatby applying a customization method in which customers directly add menus,words they want to leave, allergies, etc., they store only the information theyneed to make customized recommendations for that customer.

2. Functional Requirement

1) Function 1 Menu Recommendation Function

- Explanation: We recommend the food based on the rating given by thecustomer by the food.

(1) Detailed function 1: Select Category

- Explanation: Allows you to select the category in which you would like to berecommended for food.

(2) Detailed function 2: Save Memo

- Explanation: Save the words you want to leave on the recommended food, suchas recipes, precautions, and allergies.

(3) Detailed function 3: Save Preferences

- Explanation: Leave a preference for the recommended food and allow it to bereflected in the new food recommendations.

2) Function 2: Menu editing function

- Explanation: Ability to edit items or information on recommended food lists

(1) Detailed function 1: Add Menu

- Explanation: Add food and leave notes such as preferences.

(2) Detailed function 2: Delete Menu

- Explanation: Delete foods you don't want to be recommended again.

3) Function 3: File Storage Function

- Explanation: Upload the newly entered information, such as preferences andnotes, to the file.

3. Progress

1) Implementation of features

(1) Add Menu

- Input; menus: a dictionary that stores categories and menus, ratings: adictionary that stores menus and menu-specific evaluations, memos: a 2D dictionary that stores menus and memos.

Output; The menu you entered is added to the selected category. In addition, menus and ratings are added to the ratings, and menus and notes are added to the memos and returned.

- Description: First, you get a category input, and if that category is not in the keys of menus, you get it back in. If a category exists, enter the menu and ratingyou want to add and save additional menus and ratings in menus and ratings,respectively.

- What you've learned: Repeat statement, condition question, function, module

- Code Screenshot

```
def add_menu(menus,ratings,memos):
    while True: # 알맞은 카테고리만 입력받기
        add_cate = input("메뉴 추가를 원하는 카테고리를 선택해 주세요. ")
        if add_cate not in menus.keys():
            print("입력한 카테고리가 존재하지 않습니다. ")
        else:
            break

    add_menu = input("추가를 원하는 메뉴를 입력해 주세요. ")
    add_rat = input("추가한 메뉴에 별점을 남겨 주세요. ")
    input_memo = input("메모를 남기시겠습니까? (y/n)")
    memo = " "
    if (input_memo == 'y'):
        memo = input("메모를 입력해주세요. ")

    menus[add_cate].add(add_menu)
    ratings[add_menu] = add_rat
    memos[add_menu] = memo
    print("메뉴 추가가 완료되었습니다. ")
```

(2) Delete Menu

- Input; menus: a dictionary that stores categories and menus, ratings: adictionary that stores menus and menu-specific evaluations, memos: a 2D dictionary that stores menus and memos.

Output: Menus, where the selected menu is deleted, the ratings where the menuand rating are missing. Memo from the selected menu also disappears.

- Description: Enter the category with the menu you want to delete first, andproceed with the deletion only if the category exists in the keys of the menu.Similarly, configure the deletion to proceed only when an incoming menu exists.

- What you've learned: Repeat statement, conditional statement, function, module

- Code Screenshot

```
def del_menu(menus,ratings,memos):
    while True: # 카테고리 존재하지 않는 경우 삭제.
        del_cate = input("삭제를 원하는 메뉴가 있는 카테고리를 선택해 주세요. ")
        if del_cate not in menus.keys():
            print("선택한 카테고리가 존재하지 않습니다. 다시 선택해 주세요. ")
            continue
        else:
            break

    print(*menus[del_cate])
    while True:
        del_menu = input("삭제를 원하는 메뉴를 선택해 주세요. ")
        if del_menu not in menus[del_cate]:
            print("해당 카테고리에 존재하지 않는 음식입니다. ")
            continue
        else:
            break

    menus[del_cate].remove(del_menu)
    del ratings[del_menu]
    del memos[del_menu]
    print("삭제가 완료되었습니다. ")
```

(3) File Storage Function

- Input; menus: a dictionary that stores categories and menus, ratings: adictionary that stores menus and menu-specific evaluations, memos: a 2D dictionary that stores menus and memos.

Output: Save to a file reflecting the facts you have added or deleted. Returns the file where the results are stored.

- Description: Convert menus to a list of tubules, and then to DataFrame with columns Category and Menu. Convert each element in the list to a separate row in the expode, and set drop = True to delete the previous index value.

Other dictriaries also proceed to DataFrame in a similar way, merge them into merges and write them to the csv file as pandas.

- What you've learned: pandas, dictionary

- Code Screenshot

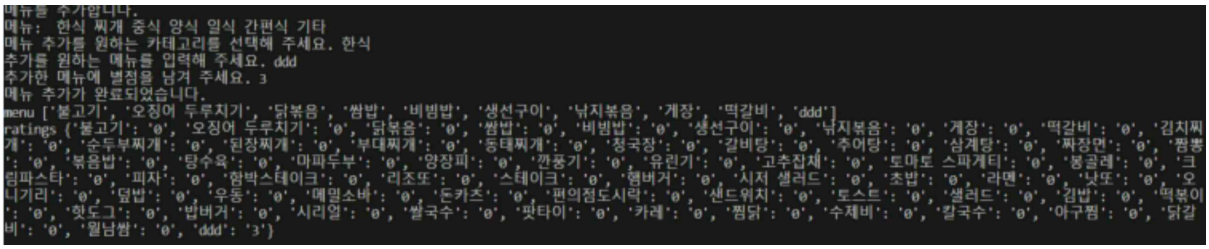
```
print("종료합니다. ")

# menus, ratings, memos를 DataFrame으로 변환
menus_df = pd.DataFrame(list(menus.items()), columns=['Category', 'Menu']).explode('Menu').reset_index(drop=True)
ratings_df = pd.DataFrame(list(ratings.items()), columns=['Menu', 'Ratings'])
memos_df = pd.DataFrame(list(memos.items()), columns=['Menu', 'Memo'])

# menus, ratings, memos를 병합
result_df = pd.merge(menus_df, ratings_df, on='Menu')
result_df = pd.merge(result_df, memos_df, on='Menu')

# 결과를 csv 파일로 저장
result_df.to_csv('menus.csv', index=False)
```

2) Test Results

(1) Add Menu

- Description: Test to see if the menu and horoscope have been added as normal.

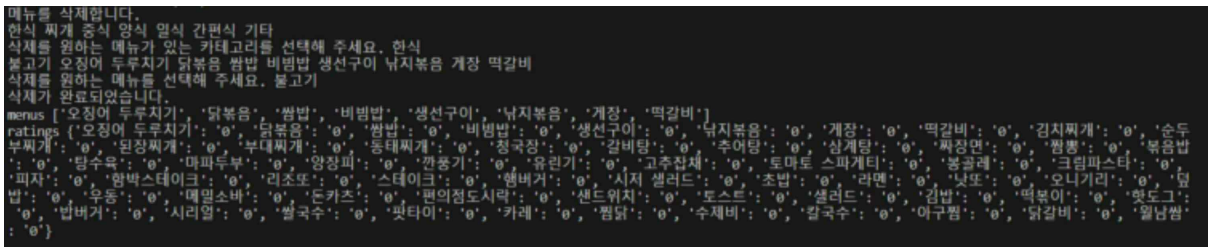
- Test Results Screenshot



(2) Delete Menu

- Description: Test to see if the menu and horoscope have been successfullydeleted.

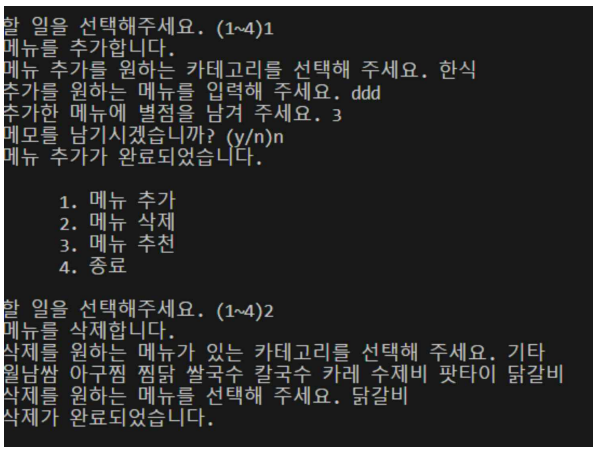
- Test Results Screenshot

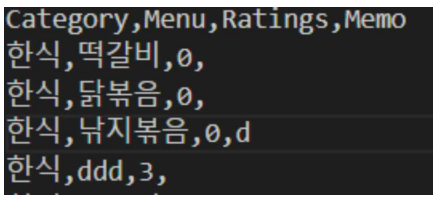


(3) File Storage Function

- Description: Test to ensure that the changes are stored in the file properly.

- Test Results Screenshot





```
간편식,편의점도시락,0,
기타,월남쌈,0,
기타,아구찜,0,
기타,찜닭,0,
기타,쌀국수,0,
기타,칼국수,0,
기타,카레,0,
기타,수제비,0,
기타,팟타이,0,
```

4. Changes in Comparison to the Plan

No changes made

5. Schedule

- 진행 상황 표기

<table>
<tr><th colspan="2">Work</th><th>11/10</th><th>11/17</th><th>11/24</th><th>12/1</th><th>12/8</th><th>12/15</th><th>12/22</th></tr>
<tr><td rowspan="2">Menu Recommendation</td><td>Select category</td><td></td><td></td><td></td><td></td><td>-----></td><td></td><td></td></tr>
<tr><td>Save Memo</td><td></td><td></td><td></td><td colspan="2">Ongoing</td><td></td><td></td></tr>
<tr><td rowspan="3">Menu editing Function</td><td>Save Preferences</td><td></td><td></td><td></td><td></td><td></td><td colspan="2">----------></td></tr>
<tr><td>Add Menu</td><td colspan="2">Completion</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Delete Menu</td><td></td><td colspan="2">Completion</td><td></td><td></td><td></td><td></td></tr>
<tr><td>File Storage Function</td><td></td><td></td><td></td><td></td><td colspan="2">Completion</td><td></td><td></td></tr>
</table>