Coursework 02 Specification: Personal Finance Tracker (Using Dictionaries)

Overview

This assignment involves developing a Personal Finance Tracker using Python, focusing on fundamental programming principles such as dictionaries, loops, functions, input/output, and input validation. Unlike the original specification, which utilized lists, your application will now use dictionaries to manage financial transactions, with keys representing the type of expenses and values being lists of expenses themselves. This change aims to enhance your understanding of data handling, program design, and testing in a practical context, with an added focus on file I/O for bulk data processing.

Objectives

- Apply essential Python programming constructs.
- Manage data using dictionary manipulations for more efficient data retrieval.
- Implement CRUD operations on financial transactions, leveraging JSON serialization for data storage.
- Introduce file operations for bulk reading of transactions from a text file, enhancing the program's usability for larger data sets.
- Conduct thorough testing and validation to ensure program robustness.

Detailed Steps and Marking Scheme (Total Marks: 100)

1. Design (25 Marks):

- Outline your approach using pseudo code, detailing the dictionary-based data structure and the application's feature flow, including the new bulk reading function.
- Illustrate using pseudo code how transactions are stored, accessed, and manipulated using dictionaries.

2. Implementation and Documentation (60 Marks):

- Develop the application with readability, modular code, and integration of JSON for data handling.
- Implement core features as specified, with each evaluated for correctness and efficiency.

- Include a function to read transactions in bulk from a text file, parsing each entry and adding it to the appropriate dictionary key based on the type of expense.

3. Test (15 Marks):

- Develop a detailed test plan, execute it, and provide a summary of your findings, covering various scenarios including bulk file reading.

Submission Guidelines

Submit all program files, including Python script(s), JSON file(s), documentation, and test plan summary, through your learning platform. Ensure your submission is well-organized and includes clear instructions for setting up and running your application, with particular attention to the new bulk reading functionality.

Sample JSON File Format

Your JSON file will now contain a dictionary with types of expenses as keys and lists of expenses as values.

Example:

```
{
 "Groceries": [
   {"amount": 150, "date": "2024-02-03"},
   {"amount": 75, "date": "2024-02-15"}
 ],
 "Salary": [
   {"amount": 1000, "date": "2024-02-01"}
 ]
}
```

Code Template for Assignment 02

```
import json

# Global dictionary to store transactions
transactions = {}

# File handling functions
def load_transactions():
    pass

def save_transactions():
    pass

def read_bulk_transactions_from_file(filename):
    # Open and read the file, then parse each line to add to the transactions dictionary
    pass

# Feature implementations
def add_transaction():
    pass

def view_transactions():
    pass

def update_transaction():
    pass

def delete_transaction():
    pass

def display_summary():
    pass
```

```
def main_menu():
    pass

if __name__ == "__main__":
    main_menu()
```