Project 2

Python Banking Application Project

## Objective

Create a bank teller console system where a single teller can manage multiple customers’ accounts. This Python program will simulate a banking application, allowing a teller to perform deposit, withdrawal, and balance checks for different customers. Key elements include basic account transactions and a navigation menu where it reinforces dictionary access, string format specs, and function logic.

## Detailed Requirements

1. Data Setup and Account Information
   * The account dictionary can be found on the last page, which should be included in your project code. **Do not edit the dictionary;** just copy!
   * Functional Details: Implement a function named get\_account that prompts the teller user to enter a specific account ID.
   * Technical Details: From the teller input, validate if the account ID exists in the bank\_info dictionary. If the account exists, it returns the user’s account dictionary; otherwise, it informs the teller the account was not found and returns an empty dictionary.
2. Account Handling Teller Functions
   * Functional Details: Implement two functions named deposit and withdraw, which take in two arguments and print the updated balance.
   * Technical Details: Both functions should accept a floating-point number as an argument parameter and the account dictionary as the second argument parameter, alter the account dictionary for the deposit or withdrawal, and print a formatted summary of the transaction. Make sure when withdrawing the amount from the balance, it is not a negative number; otherwise, print an error message.
3. User Interface Menu System
   * Functional Details: Implement a function named menu that displays the menu, which is shown to the teller to choose an action, which is 1) Deposit Money, 2) Withdraw Money, 3) Print Balance, or 4) Exit Program.
   * Technical Details: Valid inputs are integer values corresponding to the actions that can be made, which return the same value to the main function to call the function. When exiting the program, the main function uses the return integer choice to break out of the while loop.
4. Output String Formatting
   * Line Width: All printed output should be limited to **50 characters per line** to maintain readability in console environments.
   * Monetary Values: Displayed with **two decimal places** and **right-aligned** within the output line to emphasize numerical precision and alignment.
   * Transaction Summary Format: Each transaction (deposit or withdrawal) and print balance should include:
     + Account ID
     + Full Name (first and last)
     + Phone Number
     + Transaction Type (Deposit, Withdraw, Print)
     + Balance
   * Hint: Make the transaction summary into a helper function called print\_balance that takes in two parameters: account dictionary and transaction type.
   * Use this Python line for a line break: print(f"{'\*':\*^50}")
   * Hint: Make this into a function for ease of use.
5. Documentation
   * Code Docstring: For each function have a docstring that documents the description, input parameters, and returns for the function.
   * CodeComments: Your code should be well-documented, including a brief description of the program, annotations for each function and major block of code, and explanations for the purpose of variables

## Project Constraints

* Adhere strictly to best practices and concepts discussed in class.
* **No external libraries** or Python 3 features, which are off-limits for this course.
* User Input Validation: Ensure your application functions correctly across a variety of test cases for choices in deposits, withdrawals, and account IDs.
* **Ignore all invalid input types**: Assume all user input() will give the correct input type. If the correct value types are inputted, it must handle incorrect values, e.g., be able to handle incorrect user IDs, insufficient funds for withdrawal, and invalid menu choices.
* **Correct Print Formatting**: Print with the correct number of characters, alignments, and precision control.
  + Use f-strings or .format for formatting to maintain readability and performance.
  + Avoid hardcoding spaces; use alignment specifiers (<, >, ^) within format strings. Can use newline whitespace (‘\n’).
  + Ensure all outputs are wrapped or truncated to fit within the 50-character constraint.
* Name your source code file project2.py.

## Deliverables

* The Python source code file (project2.py). Must include these 5 functions: main, menu, deposit, withdraw, and get\_account. Optional helper function to print balance.
* A PDF document detailing your algorithm and design approach. (Optional 20 points)
* Note: This project is an individual effort; ensure all submitted work is entirely your own. Your submission will be scrutinized for originality to maintain academic integrity; this includes your understanding of your code mentioned in the documentation.

## Sample Output

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ACCOUNT MENU \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. Deposit to Account

2. Withdraw from Account

3. Print Account Balance

4. Exit Program

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Enter menu choice > 3

Enter account ID > acc001

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Account ID: ACC001

Name: Alice Smith

Phone: +1-555-111-0001

Account Type: Checking

Transaction: Balance Check

Account Type: Checking

Current Balance: $ 3,200.50

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ACCOUNT MENU \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. Deposit to Account

2. Withdraw from Account

3. Print Account Balance

4. Exit Program

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Enter menu choice > 2

Enter account ID > acc001

Enter withdrawal amount > $ 200.49

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Account ID: ACC001

Name: Alice Smith

Phone: +1-555-111-0001

Account Type: Checking

Transaction: Withdrawal

Amount: $ 200.49

New Balance: $ 3,000.01

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ACCOUNT MENU \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. Deposit to Account

2. Withdraw from Account

3. Print Account Balance

4. Exit Program

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Enter menu choice > 1

Enter account ID > acc001

Enter deposit amount > $ 10000.99

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Account ID: ACC001

Name: Alice Smith

Phone: +1-555-111-0001

Account Type: Checking

Transaction: Deposit

Amount: $ 10,000.99

New Balance: $ 13,001.00

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ACCOUNT MENU \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. Deposit to Account

2. Withdraw from Account

3. Print Account Balance

4. Exit Program

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Enter menu choice > 4

Thank you for using the banking system. Goodbye!

### ERROR HANDLING

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ACCOUNT MENU \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. Deposit to Account

2. Withdraw from Account

3. Print Account Balance

4. Exit Program

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Enter menu choice > 2

Enter account ID > acc001

Enter withdrawal amount > $ 20000.00

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ERROR: INSUFFICIENT FUNDS FOR WITHDRAWAL.

Current Balance: $ 13,001.00

Requested: $ 20,000.00

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ACCOUNT MENU \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. Deposit to Account

2. Withdraw from Account

3. Print Account Balance

4. Exit Program

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Enter menu choice > 1

Enter account ID > acc001

Enter deposit amount > $ -1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ERROR: INVALID AMOUNT FOR DEPOSIT.

Requested: $ -1.00

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ACCOUNT MENU \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. Deposit to Account

2. Withdraw from Account

3. Print Account Balance

4. Exit Program

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Enter menu choice > 3

Enter account ID > acc

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ERROR: ACCOUNT ACC NOT FOUND.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ACCOUNT MENU \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. Deposit to Account

2. Withdraw from Account

3. Print Account Balance

4. Exit Program

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Enter menu choice > -1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ERROR: INVALID CHOICE

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ACCOUNT MENU \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. Deposit to Account

2. Withdraw from Account

3. Print Account Balance

4. Exit Program

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Enter menu choice > 6

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ERROR: INVALID CHOICE

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## Accounts

Here is the dictionary that you can use in your program, which can be declared at the beginning of your program.

bank\_info = {

'ACC001': {

'account\_id': 'ACC001',

'first\_name': 'Alice',

'last\_name': 'Smith',

'phone': '+1-555-111-0001',

'account\_details': {

'account\_type': 'Checking',

'balance': 3200.50

}

},

'ACC002': {

'account\_id':'ACC002',

'first\_name': 'Bob',

'last\_name': 'Johnson',

'phone': '+1-555-111-0002',

'account\_details': {

'account\_type': 'Savings',

'balance': 8750.00

}

},

'ACC003': {

'account\_id':'ACC003',

'first\_name': 'Carol',

'last\_name': 'Davis',

'phone': '+1-555-111-0003',

'account\_details': {

'account\_type': 'Checking',

'balance': 150.75

}

},

}