Name: Moca Alina
Computer Science, English

Accounting project documentation

• 1. Overview

The code implements in C a simple accounting application. Users can create, edit, and delete accounts, record transactions, generate financial reports, and do other operations. The application uses console input/output or reads and saves data in CSV files.

• 2. Code Structure

Data Structures:
- Date Structure - represents the date of a transaction with day, month, and year components.
- Transaction Structure - represents a financial transaction with an account ID, a transaction date, amount, transaction type, and eventually an entity name.
- Account Structure – represents a financial account, with an ID, a type, a balance, an accountTransactions array and a number of account transactions.

Global Variables:
- Account accounts[50] – an array to store accounts
- numberOfAccounts – unsigned int to track the number of accounts
- Transaction transactions[500] - an array to store financial transactions.
- numberOfTransactions – undigned int, keeps track of the number of recorded transactions.
- csvAccountsFileName – const char*, stores the name of the csv for account data
- csvTransactionsFileName – const char*, stores the name of the csv for transaction data
- filePointer - file pointer for data storage.

Functions:
- getters for accounts: float getBalance(int nr), int getID(int nr), const char* getType(int nr), void setBalance(int nr, float newBalance).
- float readAmount() and int readID() - get data from the console
- int existingID(int id) – search for the position of the account with the given id and returns it, or -1 if the id is not found
- void createAccount() – allows the user to create a new account, specifying all the needed details
- int editAccount() – allows the user to edit an existing account, by searching it after its id and then modifying the balance
- int deleteAccount() – allows the user to delete an existing account
- Date readAndValidateDate() – reads a date from the console, validates it and then returns it
- int receiveTransfer(float amount) – if a transfer is made to an account, it creates a deposit into the desired account
- void getEntityInformation() – reads the name of the entity to which a payment was made
- int recordTransaction() – records a new transaction, specifying all the needed details
- int calculateAccountBalance() – prints the balance of a certain account
- int compareDates(Date date1, Date date2) – compares two dates, by comparing their years, months and days
- void generateAccountStatement() – prints all the necessary information about a certain account
- void transactionsRegister() – prints all information about all the transactions that were done
- void printBalances() – prints the balances of all existing accounts

- void readFromCSVFile() – gets data about accounts and transactions from CSV files
- void saveDataCSVFile() – prints all the information about accounts and transactions in the desired CSV files

Main Function:
- Presents a menu to the user, a while loop continuously prompting the user for options until the user chooses to exit.

• 3. Usage Instructions

- Option 1: Create a new account: input the type and the initial balance of the account.
- Option 2: Edit an account: enter the id of the account you want to edit, and the enter the new balance.
- Option 3: Delete an account: enter the id of the account you want to delete
- Option 4: Record a new transaction: enter the amount and the date of the transaction, the ID of the account associated with it, the type of the transaction. If the type is "payment" enter the name of the entity who received the payment, if the type is "transfer" enter the ID of the account that receives the transfer.
- Option 5: Calculate the balance of an account: enter the id of the account.
- Option 6: Generate account statement: enter the id of the account.
- Option 7: Print balances of all accounts.
- Option 8: Generate a register containing all transactions.
- Option 9: Read data from a CSV File.
- Option 10: Save data in a CSV File.
- Option 0: Exit the application.

• 4. Notes

- Dates are input day, month, and year separately.
- Account types: savings, credit, checkings.
- Transaction types: deposit, withdrawal, payment, transfer.
- The application supports up to 50 accounts and 500 transactions.

• 5. Conclusion

This accounting application provides basic functionality for recording, managing, and reporting financial accounts and transactions. Further enhancements can be made to improve user experience and add additional features.