DESIGN DOCUMENT

Team: Zero Defects

Name	Student Number
Muhammad Usman Majeed	10086980
Jessica Nahulan	10029341
Johan Cornelissen	10098176

Contents

QBasic Application Usage:

The following provides a brief overview of how to use the QBasic python front end application.
The Front End takes two arguments: the name of a Valid Accounts List file, and the name of a Transaction Summary file. Additionally, the Front End reads input from standard input, which can either be typed in by the user or redirected from a file (when testing).

For the first case (production mode):
./QBasic.py validaccounts.txt transactionsummary.txt

When testing, use redirection to feed in the "user" input, say testinput.txt:
./QBasic.py validaccounts.txt transactionsummary.txt < testinput.txt

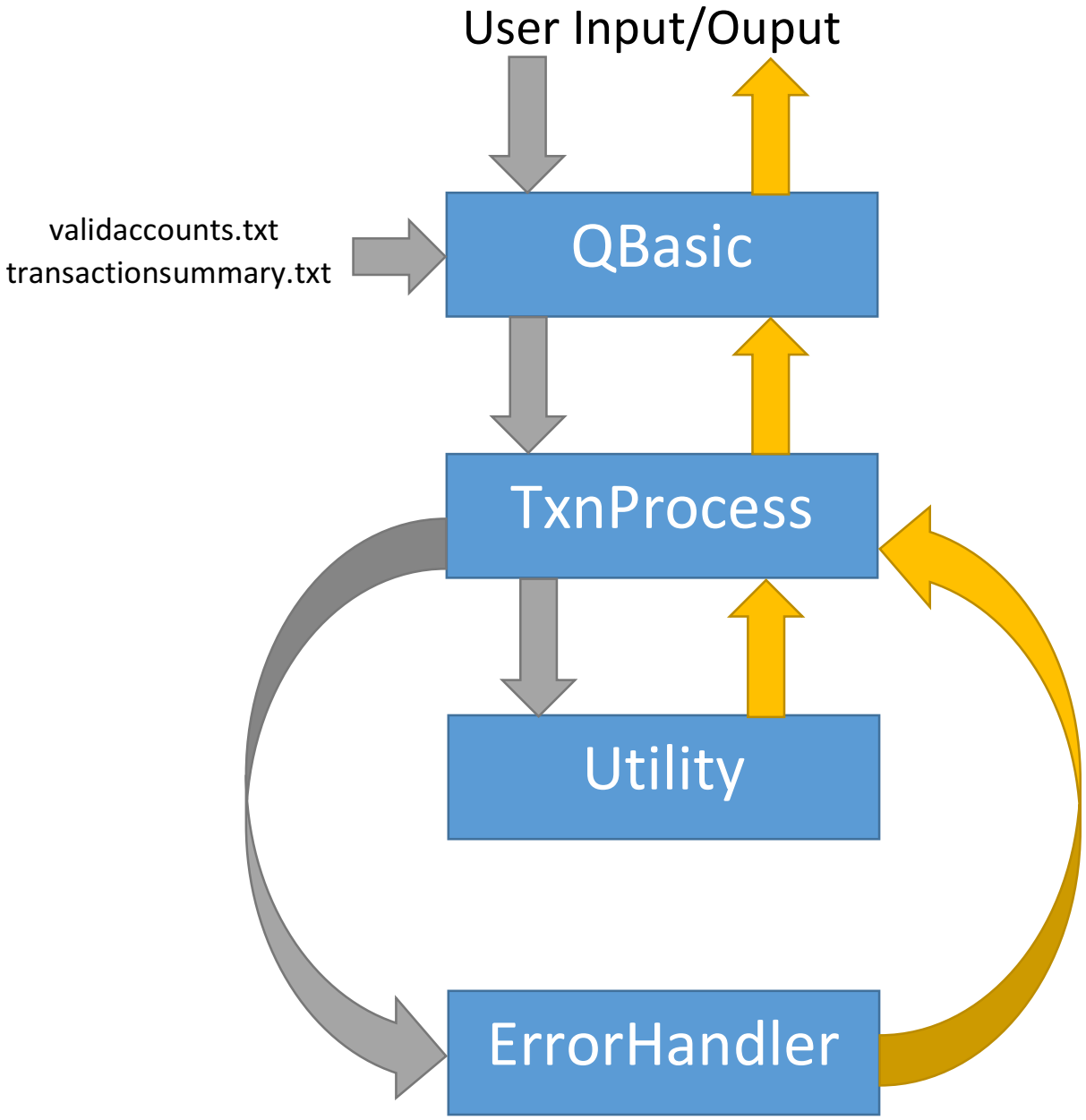
You can also save the terminal output created by the test:
./QBasic.py validaccounts.txt transactionsummary.txt < testinput.txt > testoutput.txt

Architecture Description:

The QBasic front end follows an object-oriented approach made up of 4 classes. These 4 classes consist of QBasic, TxnProcess, Utility, and ErrorHandler classes. The QBasic class is responsible for collecting user input, processing provided input files, and produces output via standard out as well as through the transaction summary file. The implementation of each transaction command is done through the TxnProcess class. Each transaction code corresponds to a TxnProcess method which configures the commands constraints and behavior. Logic that is required in multiple places, like checking what kind of user is logged in, is done through methods in the Utility class. The Utility class members are used by multiple TxnProcess methods and ensures the architecture is modular. Lastly, the ErrorHandler class is used throughout all the classes to allow for clean and concise error reporting. A simple ErrorHandler method can be invoked using a unique error code to allow for a more detailed error message to be displayed to the user. A class relationship diagram is provided below in the “Solution Structure” section to illustrate how the classes interact with one another.

Solution Structure:

Classes:



Methods:

Class	Function	Intentions
QBasic	Main	To prompt user for user input and call correct TxnProcess functions based on the transaction code entered by the user.
TxnProcess	txn_login	Function to process a user login transaction code. Sets user type (machine,agent) and reads the valid accounts file.
	txn_logout	Function to process a user logout transaction code. Logs out the current user and creates transaction summary file.
	txn_deposit	Function to process a user deposit transaction code. Checks if account number and deposit amount are valid.
	txn_createacct	Function to process a user createacct transaction code. Checks if account number and account name are valid before creating account.
	txn_deleteacct	Function to process a user deleteacct transaction code. Checks if account number and account name are valid before deleting account.
	txn_withdraw	Function to process a user withdraw transaction code. Checks if account number and withdraw amount are valid before processing withdrawal.
	txn_transfer	Function to process a user transfer transaction code. Checks if "From" and "To" account number and transfer amount are valid before processing transfer.
Utility	process_account_file	Function to process valid accounts file. Valid account numbers are stored in TxnProcess.valid_acc_list
	initiaze_withdraw_totals	Function to initialize withdrawal amounts for each valid account.
	create_txn_msg	Function to form the transaction message to be added into the transaction summary file.
	create_txn_summary_file	Function to write all cached transaction messages to transaction summary file.

	is_account_valid	Function to check if passed in account number is valid.
	is_account_unique	Function to check if passed in account number is unique (or new).
	is_amount_valid	Function to check if passed in amount is within limits for machine and agent users. Used by deposit, withdraw, and transfer transaction commands.
	is_name_valid	Function to check if passed in account name is valid.
	is_within_withdraw_limit	Function to check if passed in account will surpass daily withdraw limit by completing pending withdrawal. Returns False if limit will be reached with pending withdrawal, true if withdrawal is valid.
ErrorHandler	process_error	Function to send error string to stdout.
	__init__	Initialize error code to error string mapping.