Project 5: Journal Entry Processing

Overview: A Journal Entry represents the foundation of a financial accounting system. A financial journal contains all of the transactions that impact a company. Prior to computer systems, a journal was a large physical ledger book containing entries for all sales and expenses. One could calculate the overall company profit by adding up all of the sales or revenue transactions and subtracting all of the expense transactions. So a single transaction would be for the sale of an item or purchasing raw materials to build things. One core rule of [double entry accounting system](https://en.wikipedia.org/wiki/Double-entry_bookkeeping_system) is that a Journal Entry (JE) must balance. Instead of just specifying an amount, each transaction needs to have a debit and a credit. The total of the debits and credits must match. A single JE line can have either Debit (Dr) or Credit (Cr), but not both. The goal of this project is to simulate the creation of an accounting general ledger (GL) and produce a simple Income Statement (revenues and expenses).



### Sample Accounting Entries

Here is a photo of a sample accounting entry. Note the key elements of the Date, Detail, Account, Debit and Credit. You will be provided a text file containing data similar to this.



### Chart of Accounts

A [Chart of Accounts](https://en.wikipedia.org/wiki/Chart_of_accounts) is a listing of valid accounts to be used in a business. Each account has a number, name, and account type. For this project, we will focus only on revenue and expense accounts.

### **Project Objective:**

Experience working with multiple data objects in C++ and matching values based upon unique keys. Understand object relationships with an introduction to data structures.

Description: You will develop an object-oriented program that will:

1. Set up a Chart of Accounts structure
2. Define Journal Entry objects
3. Read and parse a text file into the journal entry objects
4. Validate each journal entry based on the rules below.
5. Create a simple income statement report

### Required Class Definitions

* [Chart of Accounts](https://en.wikipedia.org/wiki/Chart_of_accounts)
  + A chart of account is a listing of all the valid accounts. This can be represented as an array of Account objects (defined in main() ).
  + The account class definition consists of the following attributes:
    - Number
    - Name
    - Type – Which will be “Revenue” or “Expense” or “Balance Sheet”
    - Balance – Initialize to zero. This represents the total amount from all of the transactions per account.
  + Create accessors for each of the attributes.
  + Create a mutator for balance.
  + You may hardcode the following account numbers into the array. Create a general constructor and use this to initialize each slot in the array. Insert these in order to make life simpler on the income statement report.

|  |  |  |
| --- | --- | --- |
| Number | Name | Type |
| 1000 | Sales | Revenue |
| 1010 | Services | Revenue |
| 1020 | Warranties | Revenue |
| 1030 | Refunds | Revenue |
| 1040 | Shipping | Revenue |
| 6000 | Travel | Expense |
| 6005 | Meals | Expense |
| 6010 | Salaries | Expense |
| 6020 | Supplies | Expense |
| 6050 | Rent | Expense |
| 6060 | Repairs | Expense |
| 2000 | Accounts Receivable | Balance Sheet |
| 4000 | Accounts Payable | Balance Sheet |

* JE Transaction Header – This represents the overall transaction. A transaction can have many lines
  + Attributes:
    - Date of JE – This can be represented as a string.
    - Transaction ID – Uniquely identifies each journal
    - Memo – Description of the entry
    - Valid – Indicates whether the JE is valid or not
    - Lines – This would be an array or vector (extra credit) of the debits and credits in the JE
    - Line count – Represents the total number of lines in the journal (array size)
  + Methods or Member Functions:
    - Create – Updates the date, transaction ID, and memo fields from the data file.
    - Insert line – Inserts a line from the file into the journal.
    - Validate – This is the key function to write. It will be easiest to read in all of the data from the file without error checking in that function (see below). After all of the data is read in, call this function to determine if a specific JE is valid or not. It will need to receive the chart of accounts array. And it should update the valid property of the class. Check for the following errors:
      * Invalid account number on transaction – match against the list in the Chart of Accounts.
      * Negative Debit or Credit on a line
      * Both debit and credit are non-zero on a line.
      * The sum of the debits and credits are not equal.
    - isValid – Returns whether the specific JE is valid. You only want to add valid JEs to the income statement report.
    - Update Balances – This function should go through every line on the journal and add (debit) or subtract (credit) the line amount from the appropriate account balance. It should only process a valid journal. A debit will increase the balance amount, while a credit will decrease a balance amount. Pass in the Chart of Accounts array to the function.
* JE Transaction Lines – Each line represents an impact to the General Ledger against a specific account. We only need the account number, since the account name is stored in the Chart of Accounts.
  + Attributes from the data file:
    - Line Number
    - Debit
    - Credit
    - Account Number
  + Methods or Member Functions:
    - Insert or set – This function will update each member variable.
    - Getters for each member variable

### Other Functions

Your main program needs to do four things. Ideally, these should be in their own functions, but could all be done in the main function:

* Define Chart of Account (plus initialize) and Journal arrays.
* Read data file – This function should prompt the user for a file name to read from. It will then read the Journal headers followed by the lines. Do not assume that there is fixed number of journals in the file. Don’t worry about error checking at this point
* Validate JEs imported from the file – For each journal, call the validate method.
* Display income statement – For each journal, call the update balance method. Then display each account total in the form of an income statement. See Sample Output.

## Sample Input File

### Journals

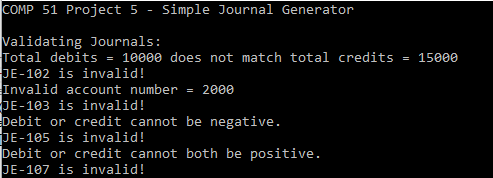
The journal file will contain multiple entries (JEs). Each line in the file will start with a line number. A zero line number indicates start of a new JE.

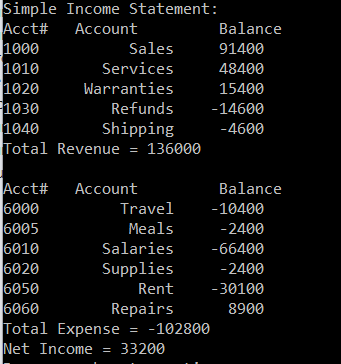
|  |  |  |  |
| --- | --- | --- | --- |
| Line Number | Date / Debit | Memo / Credit | Transaction ID /  Account Number |
| 0 | 8/31/18 | AugustTransactions | JE-100 |
| 1 | 50000 | 0 | 1000 |
| 2 | 0 | 50000 | 2000 |
| 2 | 0 | 25000 | 6010 |
| 3 | 0 | 10000 | 6050 |
| 4 | 0 | 2000 | 6020 |
| 5 | 37000 | 0 | 4000 |
| 0 | 9/30/18 | SeptTransactions | JE-101 |
| 1 | 10000 | 0 | 1010 |
| 2 | 0 | 5000 | 6000 |
| 3 | 0 | 15000 | 6010 |
| 4 | 45000 | 0 | 1000 |
| 5 | 0 | 2000 | 6005 |
| 6 | 0 | 5000 | 6050 |
| 7 | 0 | 3000 | 6060 |
| 8 | 0 | 25000 | 6020 |
| 0 | 9/15/18 | Invalid-NotBalanced | JE-102 |
| 1 | 10000 | 0 | 1000 |
| 2 | 0 | 15000 | 6000 |
| 0 | 8/15/18 | InvalidAccount | JE-103 |
| 1 | 5000 | 0 | 2000 |
| 2 | 0 | 5000 | 6000 |
| 0 | 10/1/18 | InvalidAmount | JE-104 |
| 1 | -100 | 0 | 1000 |
| 2 | 0 | -100 | 6000 |

Note – when you read in data from the JE file, you need to check the line number before doing any other processing. For line 0, you next read in the date, the memo information, and the transaction ID. Keep reading non-zero lines to add new lines to the JE. When an invalid transaction occurs, mark this as invalid (see the valid property above).

Sample Output:

This output shows the transactions which are in error and then the income statement. You may also want to display the valid transactions with the lines to help debug your program.





## Extra Credit (8 points):

So far in this class, we have studied static arrays, where the size of the array is fixed at compile time. C++ supports Vectors which allow you to add elements to an array one slot at a time. Vectors require the standard template library #include <vector>. You will need to research how to use these new data types to implement this in your class definition. Both the journal line array and the Journal array could be converted to Vectors. Make sure to document this in your program as extra credit.

Coding Requirements: Your program should perform user input validation. The user should not be allowed to enter illegal values. If an illegal value is entered, an appropriate message should be given, and the user re-prompted. Your code should be thoroughly commented, and you should follow all of the good programming practices we have discussed in class, e.g. commenting, meaningful variable names, indentation, etc. **You are not allowed to use GLOBAL VARIABLES, goto statements.**

You **must** follow all of the good programming practices discussed in class:

* Comment your code thoroughly.
* Indent your code appropriately.
* Use meaningful variable names.
* No use of global variables.
* Provide the user with understandable prompts and instructions.
* Make sure your name is included in comments at the top of your code.
* You are NOT allowed to use goto statements in this or any other COMP 51 projects.
* …

If this is not done, points will be deducted from your program and it will be impossible to earn an ‘A’.

Submission: Using the Canvas assignment feature, you should upload the source code (.cpp file).  **Make sure you submit your assignment after uploading the file attachment!**