ProCSV

Team Members:
Tabara Nosiba (tn2341), Project Manager, Tester
Tahiya Chowdhury (tc2672), Language Guru, Tester
Tahsina Saosun (ts2931), System Architect, Tester

Introduction

The ProCSV language solves issues pertaining to the manipulation of one or more CSV files. People who regularly analyze, manipulate, and compare data across multiple CSV files for data collection purposes may find it tedious and inefficient in most programming languages. ProCSV will provide users with built-in functions that will make tasks such as cleaning data, comparing, parsing, and visualizing data stored within multiple CSV files. Anyone working with large CSV files will find this language helpful.

Overview

Our language is meant to streamline the parsing through CSV files. Since many institutions' data analysis process requires work to parse and visualize insights from traditionally formatted data collection formats, such as CSV. Simplifying this process would improve developers' productivity significantly and would also save companies millions of dollars in the process.

Data Types

Integers	int x = 5;
Float	float x = 5.0;
String	string s = "hello world";
Boolean	boolean checkUnique = true;
Array	int [] a = array([3, 4, 5, 6]);

Objects

csv	Object that includes data parsed from the CSV file stored in an internal

	hashmap
hashmap	Key-pair values
table	An object with formatted columns and rows containing the parsed data. Will be implemented using DOM manipulation

Data Operators

+	Concatenation (String), Addition(int and float)
==	Equal to (String, int, and float)
/	Division (int and float)
*	Multiplication (int and float)
=	Equals (int and float)
>, <, <=, >=	Greater than, less than, greater than equal to, less than equal to (int and float)
[]	Index (Array)
||	Logical OR
&&	Logical AND
!	Logical NOT
%	Module (int and float)
++	Increment by 1 (int and float)
--	Decrement by 1 (int and float)
+=	Assignment and increment (int and float)

Control Flow

if / else if / else	Conditional statements
for	Conditional for loop statement
While	Conditional while loop statement
#	Single line comment
/* … */	Multi line comment
;	End of statement indicator

Built-in Functions

read_csv()	Reads in a CSV file
parse()	Parses data from the CSV file based on the specified delimiter
print()	Prints all data types
tablify()	Formats the data the user wants to put into a table format
showTable()	Displays table containing data into a DOM doc
merge()	Merges two different CSV files into one
find()	Looks for and returns a specific piece of data based on the argument
sim()	Compares data inside two different CSV files and returns a csv data type of common data

Sample Programs:

```
/*
Takes in two csv files as arguments and returns common data
*/

csv findSimilarData(csv c1, csv c2){
        csv similarData = sim(c1, c1);
        return similarData;
}

//Calling the function inside the main method
funct main(String[] args){
        csv input_csv1 = read('sample1.csv');
        csv input_csv2 = read('sample2.csv');

        csv result_csv = findSimilarData(input_csv1, input_csv2);
}
```

```
/*
Takes in two csv files as arguments, merges them into one csv file, and displays
data in a table
*/

table createTable(csv c1, csv c2){
        csv merged_csv = merge(c1, c2);
        table dataSet = tablify(merged_csv);
}

//Calling the function inside the main method
funct main(String[] args){
```

```
    csv input_csv1 = read('sample1.csv');
    csv input_csv2 = read('sample2.csv');

    table result_table = createTable(input_csv1, input_csv2);
}
```