Class Documentation

**Classes**:

**Parser**:

**Definition**: Parsers are objects that parse CSV files and return parsed values. The Parser class provides a way to efficiently retrieve data from CSV files in order to store and organize the data into a usable form. It contains two different ways to read the CSV files according to the type of file they are.

**Member types:**

int rowCount = the count of the rows starting at -1 on the header row of the CSV file

int colCount = the count of the columns starting at -1 on the header row of the CSV file

const int idCol = the number that corresponds to the ID column in the CSV files

const int ownerIdCol = the number that corresponds to the Owner ID column in the CSV files

const int scoreCol = the number that corresponds to the Score column in the CSV files

const int titleCol = the number that corresponds to the Title column in the CSV files

const int bodyCol = the number that corresponds to the Body column in the CSV files

const int codeCol = the number that corresponds to the Code column in the CSV files

vector<Page> rows = the vector (or hash table or AVL tree) that stores the Page objects that contain the data from the CSV files

vector<Tag> tags = the vector (or hash table or AVL tree) that stores the Tag objects that contain the data from the CSV files

vector<int> idLocations= the vector that stores the integer associated with an ID’s location

**Member Functions:**

Constructor = Constructs Parser object (public member function)

Parser(); = default constructor

1. Constructs a Parser that is able to parse and return data from a CSV file

Parameters: None

readTagFile = Accepts a file, reads it row by row and stores the data to a vector

Parameters: character pointer to the file name

Return Value: none

Example:

Parser p;

p.readTagFile(“tag.csv”);

readFile = Accepts a file, reads it row by row and stores the data to a vector

Parameters: character pointer to the file name

Return Value: An integer to allow for exception handling

Example:

Parser p;

p.readFile(“question.csv”);

csv\_read\_row = Reads a row of the CSV file and returns a vector of strings

Parameters: string reference to the line and a character to be used for delimitation or an

input stringstream reference and a character to used for delimitation

Return Value: A vector of strings containing the line

Example:

char\* file = “tags.csv”;

std::ifstream in(file);

std::vector<std::string> row = csv\_read\_row(in, ‘,’);

parseBodyWords = Parses body column, removes unwanted characters, changes to lower case and saves data to a vector

Parameters: string of the line and integer holding the ID number

Return Value: none

Example:

parseBodyWords(tempBody, endID);

findFile = Finds and returns file location based on index in idLocations vector

Parameters: integer holding the ID of the file to be found

Return Value: An integer representing the location within the idLocations vector

Example:

int position = 0;

position = findFile(2465);

TotalQuestions = returns the total number of questions

Parameters: none

Return Value: An integer representing the size of the idLocations vector

Example:

int total = 0;

total = TotalQuestions();

readTag = returns the tag contained at the index of the tags vector passed into the method

Parameters: An integer representing the index of the tags vector to be accessed

Return Value: A string containing the tag contained at the index

Example:

string s;

s = readTag(52);

readCode = returns the code contained at the index of the rows vector passed into the method

Parameters: An integer representing the index of the rows vector to be accessed

Return Value: A string containing the code contained at the index

Example:

string s;

s = readCode(52);

readBody = returns the body contained at the index of the rows vector passed into the method

Parameters: An integer representing the index of the rows vector to be accessed

Return Value: A string containing the body contained at the index

Example:

string s;

s = readBody(52);

readTitle = returns the title contained at the index of the rows vector passed into the method

Parameters: An integer representing the index of the rows vector to be accessed

Return Value: A string containing the title contained at the index

Example:

string s;

s = readTitle(52);

readTagId = returns the tag ID contained at the index of the tags vector passed into the method

Parameters: An integer representing the index of the tags vector to be accessed

Return Value: An integer containing the tag ID contained at the index

Example:

int I = 0;

I = readTagId(52);

readScore = returns the score contained at the index of the rows vector passed into the method

Parameters: An integer representing the index of the rows vector to be accessed

Return Value: An integer containing the score contained at the index

Example:

int I = 0;

I = readScore(52);

readId = returns the ID contained at the index of the rows vector passed into the method

Parameters: An integer representing the index of the rows vector to be accessed

Return Value: An integer containing the Id contained at the index

Example:

int I = 0;

I = readId(52);

Tag:

Definition: Tags are objects that hold data concerning the tag files and allow you to retrieve it. They store the ID number of the tag as an int and the tag itself as a string.

Member Types:

float tagId = the ID number of the tag

string phrase = the phrase that represents the tag

Member Functions:

Constructor = Constructs Tag object (public member function)

Tag();

Tag(float I, std::string p);