The English Language Language:

A Document Analysis System

Language Reference Manual

Emily Bau, eb3029, Project Manager
Nivita Arora, na2464, Language Guru
Michele Lin, ml3706, System Architect
Candace Johnson, crj2121, System Architect
Rabia Akhtar, ra2805, Tester

1. Introduction:

The English Language language solves problems specific to document manipulation and data extrapolation. People who would like to write scripts that analyze multiple documents quickly and can cross compare documents will find it it hard in traditional languages. Our language provides core file manipulation operations and storage structures and allows for libraries that mine statistics and check for plagiarism. This could especially be useful for teaching and publishing related activities.

Our language allows for streamlining of calculating most used words, most popular subject, time a human takes to read this file, and other useful information related to one text file. It will also help streamline comparing lists of files for searching for relevant keywords and other comparison functions.

2. Lexical Elements:

a. **Identifiers**
   i. Identifiers are for naming Docs and other data types. They are defined by at least one lowercase letter and can be followed by any combination of letters, numbers, and underscores.

b. **Reserved KeyWords:**

int	boolean	float
if	else	elif
while	for	string
doc	print	toLower
docPlus	import	

c. Literals
   i. Integer literals
      1. Series of one or more digits from 0-9
   ii. Float literals
      1. Series of digits followed by a '.' character and another series of digits. Must either have digits preceding or following the '.'
   iii. Boolean literals
      1. Value of either 'true' or 'false'
   iv. String literals
      1. Series of one or more characters

d. Operators

+, -, *, /, %,	arithmetic integer operators
==, <, >, <=, >=	numerical integer operators
||, &&, !	logical operator
=	assignment

e. Delimiters

i. Parentheses

1. Used to contain arguments in function calls as well as to ensure precedence in expressions

ii. Semicolon

1. End of statement

iii. Curly braces

1. Used to contain enclose block logic in conditionals and loops as well as to contain code in functions

iv. Commas

1. Used to separate input in function calls

f. Whitespace

i. Whitespace is only used to separate tokens

g. Comments

i. Single line comments are started with // and multiple line comments are started with /* and */

3. Data Elements

1. Primitive Data Types

Integers: Most numbers will be declared as type int (int x = 5)
Floats: Floating point numbers will be declared as type float (ex: float x = 2.3)
Booleans: Boolean values will be declared as type bool and can be True or False

2. Non Primitive Data Types

a. **<u>Strings (string)</u>**

i. Strings will be defined by type string. They will be defined with two double quotes " ". ex(string intro = "Hello World")
ii. **string.size()** - The number of characters in the string
iii. **string.append(string a)** - Append another string to the end of the string
iv. **string.equal(string a)** - Check if both strings are equal
v. **string.get(int i)** - get the character at index i

b. <u>Documents (doc)</u>

i. Document stores the contents of a passage through a string
ii. **doc.content** - retrieves the string that contains the contents of the document

4. Functions

1. Built in Functions

print()	Function to print any data type
toLower()	Changes an uppercase word to lowercase

2. Library Functions
    a. We will have an importable libraries that users can incorporate in their programs. This library will be called docPlus which will take in a Doc in the constructor and will perform analysis functions on the Doc. The following are functions within the docPlus library.

docPlus.getKeywords(int numberOfKeywords)	Return the most frequently used words in a document
docPlus.getCount(string word)	Returns number of times words occur in doc object
docPlus.compare(doc a, doc b, ...)	Compares all documents and returns percent of documents matching other documents.

3. User-Defined Functions
    a. Functions are specified by

```
return_type function_name(args) {
        … return return_type;
}
```

    b. To call this function: function_name(args)

5. Control Flow Statements:

Conditional Statements:

```
If( <bool>) {
        <expr>
} Elif(<bool>){
    <expr>

}
Else{
    <expr>
```

Loops:

```
For(<bool>) {
    <expr>
```

```
        }
    While(<bool>) {
        <expr>
        }
```

6. Sample Programs

```
    import docPlus
    boolean plagarismCheck(doc DocA, doc DocB){
    docPlus EssayA = new docPlus(DocA);
    docPlus EssayB = new docPlus(DocB);
    string [] keyWordsA = EssayA.getKeywords(10);
    string [] keyWordsB = EssayA.getKeywords(10);
    int  totalSimilar = 0;

    for(int i =0; i<9; i++){
        for(j=0; j<9; j++){
            if(keyWordsA[i]==keyWords[j]){
                totalSimilar++;
            }           }
    }

    if(totalSimilar >7){
        return True;
    }
    return False;
}

doc findRelevant (docPlus [] docs, string keyword) {
    doc mostRelevant;
    int keywordCount = 0;
    for (int i = 0; i < docs.length; i++) {
        if (docs[i].getCount(keyword) > keywordCount) {
            mostRelevant = docs[i];
        }
    }
    return mostRelevant;
}
```