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

Introduction 2
Lexical Elements: 2
Data Elements 3
Functions 3
Control Flow Statements: 4
Sample Programs 5
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:**

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>boolean</td>
<td>float</td>
</tr>
<tr>
<td>if</td>
<td>else</td>
<td>elif</td>
</tr>
<tr>
<td>while</td>
<td>for</td>
<td>string</td>
</tr>
<tr>
<td>doc</td>
<td>print</td>
<td>toLower</td>
</tr>
<tr>
<td>docPlus</td>
<td>import</td>
<td></td>
</tr>
</tbody>
</table>

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**
1. Arithmetic integer operators
2. Numerical integer operators
3. Logical operator
4. 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. **Strings**: Strings will be defined by type string. They will be defined with two double quotes " ". ex( string intro = “Hello World”)
      i. **string.size()** - The number of characters in the string
      ii. **string.append(string a)** - Append another string to the end of the string
      iii. **string.equal(string a)** - Check if both strings are equal
      iv. **string.get(int i)** - get the character at index i

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

4. Functions

1. Built in Functions
## 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.

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>docPlus.getKeywords(int numberOfKeywords)</code></td>
<td>Return the most frequently used words in a document</td>
</tr>
<tr>
<td><code>docPlus.getCount(string word)</code></td>
<td>Returns number of times words occur in doc object</td>
</tr>
<tr>
<td><code>docPlus.compare(doc a, doc b, ....)</code></td>
<td>Compares all documents and returns percent of documents matching other documents.</td>
</tr>
</tbody>
</table>

## 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>
```
6. Sample Programs

```java
import docPlus;
boolean plagiarismCheck(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;
}
```