5 - LEX Analysis

Monday, February 26, 2024 11:17 PM

Ashima Fatima Seik Mugibur Raghman, 21BAI1830

AIM: Write a LEX program to

a. count the number of vowels & constants in a given string

b. count the number of keywords, identifiers & digits in the given string / sample C program.

c. check whether the given string abc@gmail.com is a valid mail ID.

PROCEDURE:

1. Get user input using switch case to determine if user input is a sentence, code or an email.
2. If it is a sentence,
   a. iterate through each character and convert it to lowercase.
   b. check if it is a vowel and if it is, increment the 'vowelCount' variable.
   c. otherwise, increment the '~~constant~~ consonant Count' variable.
   d. print 'vowelCount' and '~~constant~~ consonant Count'.
3. If the user input was code,
   a. tokenize the input string with the delimiters " \n\t" to extract the words.
   b. check if the tokens match any keywords and if so, increment variable 'keywordCount'.
   c. check if the token starts with an alphabet or an underscore and if so increment the 'identifierCount'.
   d. check if the first character of the ~~te~~ token is a digit and if so increment the 'digitCount' variable.
   e. print all the above variables

'digit Count' variable.

e. print all the above variables.

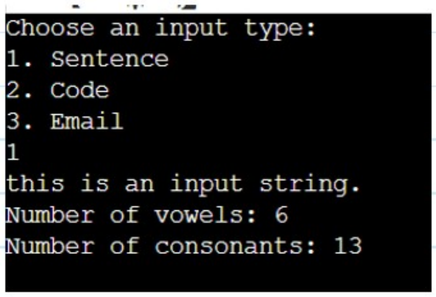
4. If user input is an email,
   a. iterate through each character of the input string
   b. if the '@' symbol is found, set the 'atSymbolFound' variable to 'true'.
   c. if the character '.' is found after the '@' symbol has been found, increment 'dot Count'.
   d. email is valid if 'atSymbolFound' = true and dotCount > 0.
   e. print email validity output.

5. As default case, output "Invalid input type".

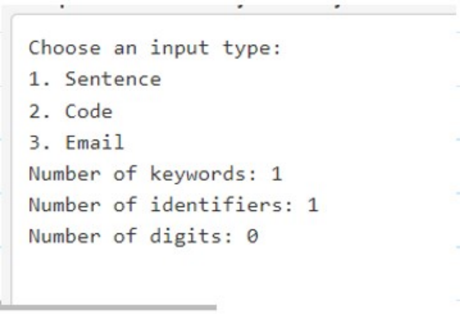
Sample Input :

```
1
This is a sample input string.
2
int main(){
    int x = 10;
    float y = 3.14;
    return 0;
}
3                                  3
abc@gmail.com                      abc@gmailcom
   ↳ valid                              ↳ invalid.
```

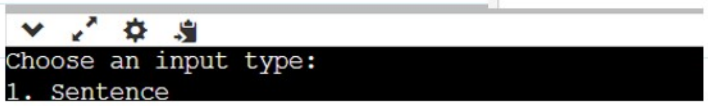
OUTPUT :



```
Choose an input type:
1. Sentence
2. Code
3. Email
1
this is an input string.
Number of vowels: 6
Number of consonants: 13
```

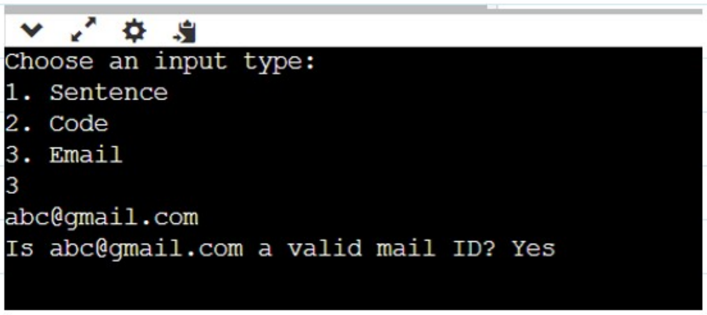


```
Choose an input type:
1. Sentence
2. Code
3. Email
Number of keywords: 1
Number of identifiers: 1
Number of digits: 0
```

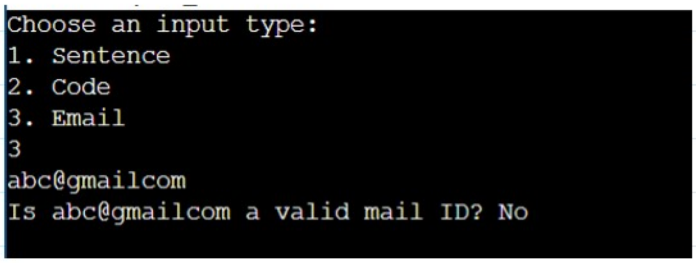


```
Choose an input type:
1. Sentence
```

```
Choose an input type:
1. Sentence
```



```
Choose an input type:
1. Sentence
2. Code
3. Email
3
abc@gmail.com
Is abc@gmail.com a valid mail ID? Yes
```



```
Choose an input type:
1. Sentence
2. Code
3. Email
3
abc@gmailcom
Is abc@gmailcom a valid mail ID? No
```

CODE :

```
#include <iostream>
#include <cstring>
using namespace std;
int vowelCount = 0;
int consonantCount = 0;
int keywordCount = 0;
int identifierCount = 0;
int digitCount = 0;
bool validMailID = false;
int yylex(char* input, int type);
int main() {
    cout << "Choose an input type:\n";
    cout << "1. Sentence\n";
    cout << "2. Code\n";
    cout << "3. Email\n";
    int choice;
    cin >> choice;
    switch (choice) {
        case 1: {
            char input[100];
            cin.ignore(); // Ignore newline character left in buffer
            cin.getline(input, sizeof(input));
            yylex(input, 1);
            break;
        }
        case 2: {
            char input[1000];
            cin.ignore();
            cin.getline(input, sizeof(input));
            yylex(input, 2);
            break;
        }
        case 3: {
            char input[100];
            cin.ignore();
            cin.getline(input, sizeof(input));
            yylex(input, 3);
            break;
        }
        default:
            cout << "Invalid choice";
    }
    return 0;
}
int yylex(char* input, int type) {
    char* token = nullptr;
    bool atSymbolFound = false;
    int dotCount = 0;
    switch (type) {
        case 1: // Sentence
            for (int i = 0; i < strlen(input); ++i) {
                char ch = tolower(input[i]);
                if (isalpha(ch)) {
                    if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch ==
'u') {
                        vowelCount++;
                    } else {
                        consonantCount++;
                    }
                }
            }
            cout << "Number of vowels: " << vowelCount << endl;
            cout << "Number of consonants: " << consonantCount << endl;
            break;
        case 2: // Code
            token = strtok(input, " \n\t");
            while (token != NULL) {
                if (strcmp(token, "int") == 0 || strcmp(token, "float") == 0 ||
strcmp(token, "return") == 0) {
                    keywordCount++;
                } else if (isalpha(token[0]) || token[0] == '_') {
                    identifierCount++;
                } else if (isdigit(token[0])) {
                    digitCount++;
                }
                token = strtok(NULL, " \n\t");
            }
            cout << "Number of keywords: " << keywordCount << endl;
            cout << "Number of identifiers: " << identifierCount << endl;
            cout << "Number of digits: " << digitCount << endl;
            break;
        case 3: // Email
            for (int i = 0; i < strlen(input); ++i) {
                if (input[i] == '@') {
                    atSymbolFound = true;
                }
                if ((atSymbolFound = true) && input[i] == '.') {
                    dotCount++;
                }
            }
            validMailID = atSymbolFound && dotCount > 0;
            cout << "Is " << input << " a valid mail ID? " << (validMailID ?
"Yes" : "No") << endl;
            break;
        default:
            cout << "Invalid input type";
    }
    return 0;
}
```