Lab Report 4

CSE-3204 (Compiler Design Lab)

Submitted By:

Name: Md Sobuj Mia

Roll: 06

Session: 2019-20

Department: CSE

Submitted To:

Sharad Hasan

Lecture

Dept. Of CSE

Sheikh Hasina University

Netrokona

Department of Computer Science and Engineering

Sheikh Hasina University

Netrokona-2400, Bangladesh

1. Lex Program for Counting Spaces, Lines, Words, and Characters

Objective

Develop a Lex program to analyze text and count spaces, lines, words, and characters.

Procedure

Utilize Lex rules to identify and count spaces, lines, words, and characters.

Results

Display the counts for each category.

Conclusion

The Lex program effectively counts spaces, lines, words, and characters in a given text.

Code

```
%{
#include <stdio.h>
int spaceCount = 0, lineCount = 0, wordCount = 0, charCount = 0;
%}
%%
[ \t]    { spaceCount++; charCount++; }
\n       { spaceCount++; lineCount++; charCount++; }
[a-zA-Z]+ { wordCount++; charCount += yyleng; }
.        { charCount++; }
%%
int main() {
printf("Enter C code (Ctrl+D or Ctrl+Z to end input):\n");
 yylex();
printf("Spaces: %d\nLines: %d\nWords: %d\nCharacters: %d\n", spaceCount, lineCount, wordCount,  charCount);
    return 0;
}
```

2. Lex Program for Identifying and Counting Capital Words

Objective

Create a Lex program to recognize and count capital words.

Procedure

Use Lex patterns to identify capital words and maintain a count.

Results

Display and count the identified capital words.

Conclusion

The Lex program accurately recognizes and counts capital words in the given text.

Code

```
%{
#include <stdio.h>
int capitalWordCount = 0;
%}
%%
[A-Z][a-z]* { printf("Capital Word: %s\n", yytext); capitalWordCount++; }
%%
int main() {
    printf("Enter C code (Ctrl+D or Ctrl+Z to end input):\n");
    yylex();
    printf("Total Capital Words: %d\n", capitalWordCount);
    return 0;
}
```

3. Lex Program for Counting Vowels and Consonants

Objective

Develop a Lex program to count vowels and consonants.

Procedure

Define Lex rules to identify and count vowels and consonants.

Results

Display counts for vowels and consonants separately.

Conclusion

The Lex program efficiently counts vowels and consonants in a given text.

Code

```
%{
#include<stdio.h>
int vowelCount = 0, consonantCount = 0;
%}

%%
[aeiouAEIOU] { vowelCount++; }
[a-zA-Z]     { consonantCount++; }
.            { /* Ignore other characters */ }

%%

int main() {
    printf("Enter C code (Ctrl+D or Ctrl+Z to end input):\n");
    yylex();
    printf("Vowels: %d\nConsonants: %d\n", vowelCount, consonantCount);
    return 0;
}
```

4. Lex Program for Recognizing Valid Arithmetic Expressions

Objective

Design a Lex program to identify valid arithmetic expressions and display operators and operands.

Procedure

Utilize Lex rules to recognize valid expressions and extract operands/operators.

Results

Display identified operators and operands in valid expressions.

Conclusion

The Lex program accurately recognizes valid arithmetic expressions and extracts operators and operands.

Code

```
%{
#include<stdio.h>
int operandCount = 0, operatorCount = 0;
%}

%%
[0-9]+      { operandCount++; }
[-+*/]      { operatorCount++; }
[ \t\n]     { /* Ignore whitespace and newline characters */ }
.           { /* Invalid character, handle accordingly */ }

%%

int main() {
    printf("Enter C code (Ctrl+D or Ctrl+Z to end input):\n");
    yylex();
    printf("Operators: %d\nOperands: %d\n", operatorCount, operandCount);
    return 0;
}
```

5. Lex Program for Counting Comments

Objective

Create a Lex program to count the number of comments in a given text.

Procedure

Define Lex rules to identify and count comments.

Results

Display the count of comments in the input text.

Conclusion

The Lex program effectively counts the number of comments in a given text.

Code

```
%{
int commentCount = 0;
%}
%%
"/*"([^*]|[*]+[^*/])*[*]+"/" { commentCount++; }
"//"                         { commentCount++; }
%%

int main() {
   printf("Enter C code (Ctrl+D or Ctrl+Z to end input):\n");
    yylex();
    printf("Total Comments: %d\n", commentCount);
    return 0;
}
```

6. Lex Program to Determine Whether an Email Address is Valid or Not

Objective

Develop a Lex program to determine whether an email address is valid or not.

Procedure

Implement Lex rules to validate email addresses.

Results

Display validation results for email addresses.

Conclusion

The Lex program accurately determines whether an email address is valid.

Code

```
%{
#include<stdio.h>
%}
%%
[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,} { printf("Valid Email: %s\n", yytext); }
.                                              { /* Ignore invalid characters */ }
%%

int main() {
   printf(“Enter an Email Address :”);
    yylex();
    return 0;
}
```

7. Lex Program to Determine Whether an IP Address is Valid or Not

Objective

Create a Lex program to determine whether an IP address is valid.

Procedure

Define Lex rules to validate IP addresses.

Results

Display validation results for IP addresses.

Conclusion

The Lex program accurately determines whether an IP address is valid.

Code

```
%{
#include<stdio.h>
%}
%%
(\d{1,3}\.){3}\d{1,3} { printf("Valid IP: %s\n", yytext); }
.                     { /* Ignore invalid characters */ }
%%

int main() {
  printf("Enter an IP Adress:\n");
    yylex();
    return 0;
}
```

8. Lex Program to Determine Whether an HTTP/HTTPS Address is Valid or Not

Objective

Develop a Lex program to determine whether an HTTP/HTTPS address is valid.

Procedure

Implement Lex rules to validate HTTP/HTTPS addresses.

Results

Display validation results for HTTP/HTTPS addresses.

Conclusion

The Lex program accurately determines whether an HTTP/HTTPS address is valid.

Code

```
%{
#include<stdio.h>
%}
%%
http://[a-zA-Z0-9.-]+\.[a-zA-Z]{2,} { printf("Valid HTTP Address: %s\n", yytext); }
https://[a-zA-Z0-9.-]+\.[a-zA-Z]{2,} { printf("Valid HTTPS Address: %s\n", yytext); }
.                                   { /* Ignore invalid characters */ }

%%
int main() {
   printf("Enter a HTTP Adress:\n");
    yylex();
    return 0;
}
```

9. Lex Program to Count the Number of Tokens

Objective

Develop a Lex program to count the number of tokens.

Procedure

Implement Lex rules to count tokens.

Results

Display the total count of tokens.

Conclusion

The Lex program accurately counts the number of tokens in a given text.

Code

```
%{
#include<stdio.h>
int tokenCount = 0;
%}
%%
[a-zA-Z0-9]+ { printf("Token: %s\n", yytext); tokenCount++; }
.           { /* Ignore other characters */ }

%%

int main() {
printf("Enter C code (Ctrl+D or Ctrl+Z to end input):\n");
 yylex();
    printf("Total Tokens: %d”, tokenCount);
}
```