LAB1

BASIC LEX PROGRAMS

1. Write a Lex program to Recognize and Print Integer Numbers.

Code:

```
%{
#include <stdio.h>
%}

%%
[0-9]+      { printf("Found a number: %s\n", yytext); }
.           { /* ignore everything else */ }
%%

int main() {
    yylex();
    return 0;
}
```

Output:

```
asecomputerlab@hp-desktop:~/Desktop/22075$ gcc lex.yy.c -o 1 -lfl
asecomputerlab@hp-desktop:~/Desktop/22075$ ./1
123 abc 456
Found a number: 123
Found a number: 456
```

2. Lex program to count the number of vowels and consonants in a given string.

Code:

```
%{
#include <stdio.h>
int vowels = 0;
int cons = 0;
%}

%%
[aeiouAEIOU] { vowels++; }
[a-zA-Z]    { cons++; }
%%

int yywrap() {
    return 1;
}

int main() {
    printf("Enter the string.. at end press ^d\n");
    yylex();
    // consonants include vowels currently, subtract vowels
    cons = cons - vowels;
    printf("No of vowels=%d\nNo of consonants=%d\n",
vowels, cons);
    return 0;
}
```

Output:

```
asecomputerlab@hp-desktop:~/Desktop/22075$ ./count
Enter the string.. at end press ^d
Bikrant Pandit

No of vowels=4
No of consonants=5
```

3. Program to count the number of characters, words, spaces, end of lines in a given input file.

Code:

```
%{
#include <stdio.h>

int char_count = 0;
int word_count = 0;
int space_count = 0;
int newline_count = 0;
int in_word = 0;
%}

%%
[ \t]          { space_count++; char_count += yyleng;
in_word = 0; }
\n             { newline_count++; char_count++; in_word =
0; }
[a-zA-Z0-9]+   { word_count++; char_count += yyleng;
in_word = 1; }
.              { char_count++; in_word = 0; }
%%

int yywrap() {
    return 1;
}

int main() {
    printf("Enter input (Ctrl+D to end):\n");
    yylex();
    printf("\nCharacters: %d\nWords: %d\nSpaces: %d\nNew
lines: %d\n",
           char_count, word_count, space_count,
newline_count);
    return 0;
}
```

Output:

```
asecomputerlab@hp-desktop:~/Desktop/22075$ ./count_stats
Enter input (Ctrl+D to end):
Hello Bikrant!

Characters: 15
Words: 2
Spaces: 1
New lines: 1
```

4. Lex Program to Recognize and Print Keywords.

Code:

```
%{
#include <stdio.h>
%}

%%
"if"        { printf("Keyword: if\n"); }
"else"      { printf("Keyword: else\n"); }
"while"     { printf("Keyword: while\n"); }
[a-zA-Z]+   { printf("Identifier: %s\n", yytext); }
[ \t\n]+    ;  // Skip whitespace
.           { printf("Unknown character: %s\n", yytext); }
%%

int main() {
    yylex();
    return 0;
}
```

Output:

```
asecomputerlab@hp-desktop:~/Desktop/22075$ flex 2.l
asecomputerlab@hp-desktop:~/Desktop/22075$ gcc lex.yy.c -o 2 -lfl
asecomputerlab@hp-desktop:~/Desktop/22075$ ./2
123 abc 456
Unknown character: 1
Unknown character: 2
Unknown character: 3
Identifier: abc
Unknown character: 4
Unknown character: 5
Unknown character: 6
```

5. Write a lex program to tokenize a simple arithmetic expression with numbers and operators

Code:

```
%{
#include <stdio.h>
%}

%%
[0-9]+          { printf("NUMBER(%s) ", yytext); }
"+"             { printf("PLUS "); }
"-"             { printf("MINUS "); }
"*"             { printf("MULTIPLY "); }
"/"             { printf("DIVIDE "); }
"("             { printf("LPAREN "); }
")"             { printf("RPAREN "); }
[ \t\n]+        ;  // skip whitespace
.               { printf("UNKNOWN(%s) ", yytext); }
%%

int main() {
    yylex();
    printf("\n");
    return 0;
}
```

Output:

```
asecomputerlab@hp-desktop:~/Desktop/22075$ gcc lex.yy.c -o 4 -lfl
asecomputerlab@hp-desktop:~/Desktop/22075$ ./4
123+34
NUMBER(123) PLUS NUMBER(34)

+23-1*3(2+2)
PLUS NUMBER(23) MINUS NUMBER(1) MULTIPLY NUMBER(3) LPAREN NUMBER(2) PLUS NUMBER(2) RPAREN
```