EXP-1 Implementation of Lexical analyser for a C program

Name: - Sai Mohit Ambekar

Reg No: - RA1911031010137

Class: - CSE-IT (L2 Section)

Aim: -

To write a program for lexical analyser which takes a C file as the input file and converts the content as count of tokens.

Algorithm: -

1. Read the C program file
2. Create lists of keywords, constants, operators, special symbols.
3. Read each line in the file, split the words in each line.
4. If the word is in any of the above lists, append it to a separate list and repeat this step till the last line of the C program.
5. Print the tokens and their respective counts in the C program.

Code: -

```c
#include <stdbool.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>


int delim = 0;
int op = 0;
int iden = 0;
int key = 0;
int inte= 0;
int realno = 0;
int iv = 0;
```

```
bool isDelimiter(char ch)
{
    if (ch == ' ' || ch == '+' || ch == '-' || ch == '*' ||
        ch == '/' || ch == ',' || ch == ';' || ch == '>' ||
        ch == '<' || ch == '=' || ch == '(' || ch == ')' ||
        ch == '[' || ch == ']' || ch == '{' || ch == '}')
    {
        return (true);
        delim++;
    }
    return (false);
}
bool isOperator(char ch)
{
    if (ch == '+' || ch == '-' || ch == '*' ||
        ch == '/' || ch == '>' || ch == '<' ||
        ch == '=')
    {
        return (true);
        op++;
    }
    return (false);
}
bool validIdentifier(char *str)
{
    if (str[0] == '0' || str[0] == '1' || str[0] == '2' ||
        str[0] == '3' || str[0] == '4' || str[0] == '5' ||
        str[0] == '6' || str[0] == '7' || str[0] == '8' ||
        str[0] == '9' || isDelimiter(str[0]) == true)
        return (false);
    return (true);
}
bool isKeyword(char *str)
{
    if (!strcmp(str, "if") || !strcmp(str, "else") ||
        !strcmp(str, "while") || !strcmp(str, "do") ||
        !strcmp(str, "break") ||
```

```
        !strcmp(str, "continue") || !strcmp(str, "int") || !strcmp(str, "double") || !strcmp(str, "float") || !strcmp(str, "return") ||
!strcmp(str, "char") || !strcmp(str, "case") || !strcmp(str, "char") || !strcmp(str, "sizeof") || !strcmp(str, "long") ||
!strcmp(str, "short") || !strcmp(str, "typedef") || !strcmp(str, "switch") || !strcmp(str, "unsigned") || !strcmp(str, "void") ||
!strcmp(str, "static") || !strcmp(str, "struct") || !strcmp(str, "goto"))
        return (true);
    return (false);
}
bool isInteger(char *str)
{
    int i, len = strlen(str);

    if (len == 0)
        return (false);
    for (i = 0; i < len; i++)
    {
        if (str[i] != '0' && str[i] != '1' && str[i] != '2' && str[i] != '3' && str[i] != '4' && str[i] != '5' && str[i] != '6' && str[i] != '7'
&& str[i] != '8' && str[i] != '9' || (str[i] == '-' && i > 0))
            return (false);
    }
    return (true);
}

bool isRealNumber(char *str)
{
    int i, len = strlen(str);
    bool hasDecimal = false;

    if (len == 0)
        return (false);
    for (i = 0; i < len; i++)
    {
        if (str[i] != '0' && str[i] != '1' && str[i] != '2' && str[i] != '3' && str[i] != '4' && str[i] != '5' && str[i] != '6' && str[i] != '7'
&& str[i] != '8' && str[i] != '9' && str[i] != '.' ||
            (str[i] == '-' && i > 0))
            return (false);
        if (str[i] == '.')
            hasDecimal = true;
```

```
    }
    return (hasDecimal);
}

char *subString(char *str, int left, int right)
{
    int i;
    char *subStr = (char *)malloc(
        sizeof(char) * (right - left + 2));
    for (i = left; i <= right; i++)
        subStr[i - left] = str[i];
    subStr[right - left + 1] = '\0';
    return (subStr);
}
void parse(char *str)
{
    int left = 0, right = 0;
    int len = strlen(str);

    while (right <= len && left <= right)
    {
        if (isDelimiter(str[right]) == false)
            right++;

        if (isDelimiter(str[right]) == true && left == right)
        {
            if (isOperator(str[right]) == true)
            {
                printf("'%c' is an Operator.\n", str[right]);
                op++;
            }

            right++;
            left = right;
        }
        else if (isDelimiter(str[right]) == true && left != right || (right == len && left != right))
        {
```

```
        char *subStr = subString(str, left, right - 1);

        if (isKeyword(subStr) == true)
        {
            printf("'%s' is a Keyword.\n", subStr);
            key++;
        }

        else if (isInteger(subStr) == true)
        {
            printf("'%s' is an Integer.\n", subStr);
            inte++;
        }

        else if (isRealNumber(subStr) == true)
        {
            printf("'%s' is a Real Number.\n", subStr);
            realno++;
        }

        else if (validIdentifier(subStr) == true && isDelimiter(str[right - 1]) == false)
        {
            iden++;
            printf("'%s' is a Valid Identifiier.\n", subStr);
        }

        else if (validIdentifier(subStr) == false && isDelimiter(str[right - 1]) == false)
        {
            printf("'%s' is not a Valid Identifiier.\n", subStr);
            iv++;
        }
        left = right;
    }
  }
  return;
}
```

```c
int main()
{
    char str[100] = "int x = 8 + y; ";
    parse(str);
    printf("Number of Lines = %d\n", delim);
    printf("Number of Operator = %d\n", op);
    printf("Number of Identifier = %d\n", iden);
    printf("Number of Keyword = %d\n", key);
    printf("Number of Integer = %d\n", inte);
    printf("Number of Real Numbers = %d\n", realno);
    printf("Number of Invalid Identifier = %d\n", iv);
    return 0;
}
```

Output: -

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

"/Users/saimohitambekar/Documents/Sai Work/Class/Compiler Design Lab/"Exp_1
'int' is a Keyword.
'x' is a Valid Identifiier.
'=' is an Operator.
'8' is an Integer.
'+' is an Operator.
'y' is a Valid Identifiier.
Number of Lines = 0
Number of Operator = 2
Number of Identifier = 2
Number of Keyword = 1
Number of Integer = 1
Number of Real Numbers = 0
Number of Invalid Identifier = 0
saimohitambekar@Sais-MacBook-Air Compiler Design Lab %
```

Result: -

The given program has been successfully executed.