NAME: SHIVAM DAVE

Reg no: 21BCB0107

Compiler Design Lab

Digital Assignment 1

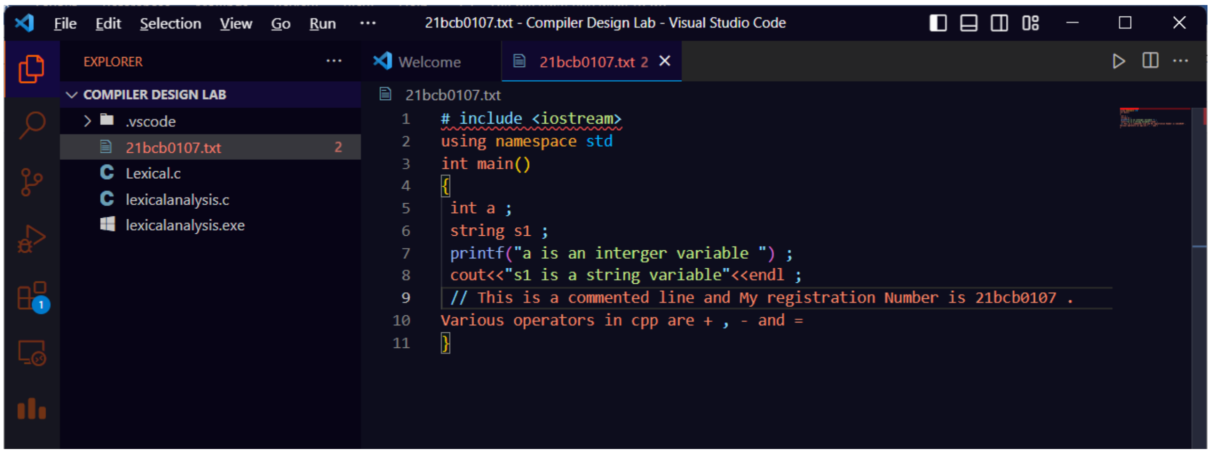
Aim: C program to identify tokens

Code:

```
#include <stdio.h>
#include <string.h>

int KeywordFunc(char* a) {
    char* arr[] = {"void", "using", "namespace", "int", "include", "<iostream>",
                   "std", "main()", "cin", "cout", "return", "float", "double",
                   "string", "endl"};
    int i;
    for (i = 0; i < 14; i++) {
        if (strcmp(arr[i], a) == 0) {
            return 1;
        }
    }
    return 0;
}

int main() {
    int Op = 0;
    int id = 0;
    int key = 0;
    int sym = 0;
    int c = 0;
    char str[100];
    FILE* file;
    char* filename;
    filename = "./21bcb0107.txt";
    file = fopen(filename, "r");
    while (fscanf(file, "%s", str) != EOF) {
        if (strcmp(str, "+") == 0 || strcmp(str, "-") == 0 || strcmp(str, "*") == 0 ||
            strcmp(str, "/") == 0 || strcmp(str, "^") == 0 || strcmp(str, "&&") == 0 ||
            strcmp(str, "||") == 0 || strcmp(str, "=") == 0 || strcmp(str, "==") == 0 ||
            strcmp(str, "&") == 0 || strcmp(str, "|") == 0 || strcmp(str, "%") == 0 ||
            strcmp(str, "++") == 0 || strcmp(str, "--") == 0 || strcmp(str, "+=") == 0 ||
            strcmp(str, "-=") == 0 || strcmp(str, "/=") == 0 || strcmp(str, "=") == 0 ||
            strcmp(str, "%=") == 0) {
            printf("%s is an operator\n", str);
            Op++;
        } else if (KeywordFunc(str)) {
            printf("%s is a keyword\n", str);
            key++;
        } else if (strcmp(str, "(") == 0 || strcmp(str, "{") == 0 ||
                   strcmp(str, "[") == 0 || strcmp(str, ")") == 0 ||
                   strcmp(str, "}") == 0 || strcmp(str, "]") == 0 ||
                   strcmp(str, "<") == 0 || strcmp(str, ">") == 0 ||
                   strcmp(str, "()") == 0 || strcmp(str, ";") == 0 ||
                   strcmp(str, "<<") == 0 || strcmp(str, ">>") == 0 ||
                   strcmp(str, ",") == 0 || strcmp(str, "#") == 0) {
            printf("%s is a symbol\n", str);
            sym++;
        } else if (strcmp(str, "\n") == 0 || strcmp(str, " ") == 0 || strcmp(str, "") == 0) {
            // Skip whitespace and empty strings
        } else if (isdigit(str[0])) {
            int x = 0;
            if (!isdigit(str[x++])) {
                continue;
            } else {
                printf("%s is a constant\n", str);
                c++;
            }
        } else {
            printf("%s is an identifier\n", str);
            id++;
        }
    }
    printf("The number of Keywords is: %d\n", key);
    printf("The number of Symbols is: %d\n", sym);
    printf("The number of constants is: %d\n", c);
    printf("The number of identifiers is: %d\n", id);
    printf("The number of operators is: %d\n", Op);
    return 0;
}
```

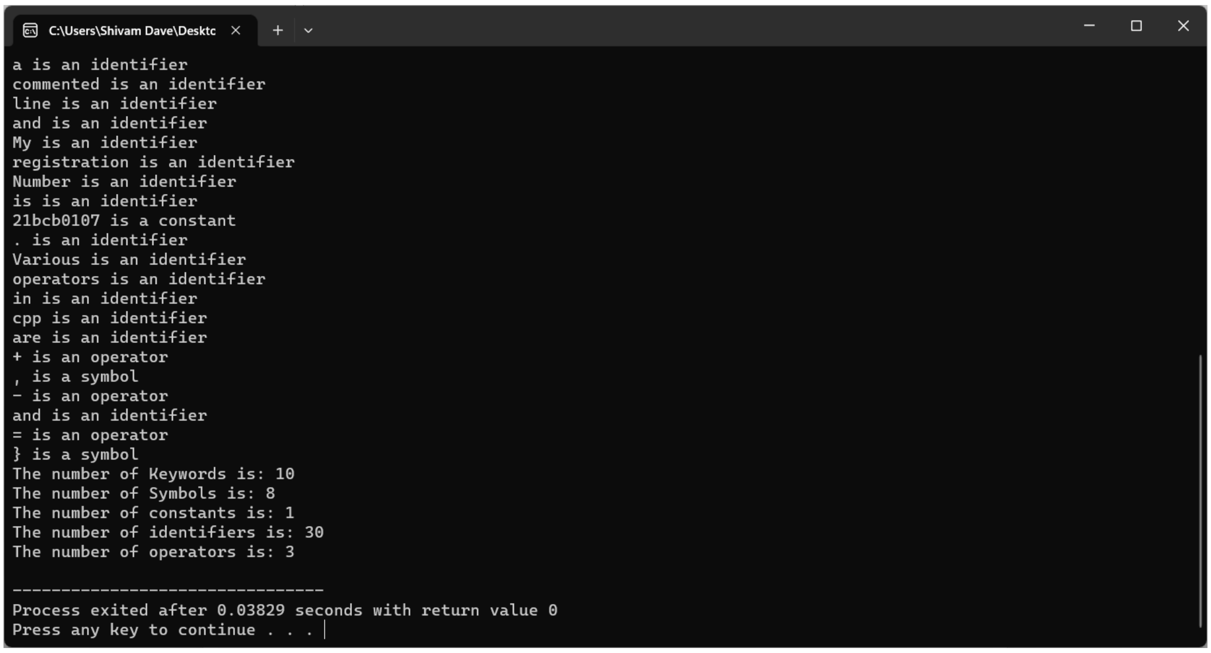
Input file: 21bcb0107.txt

21BCB0107



```
# include <iostream>
using namespace std
int main()
{
 int a ;
 string s1 ;
 printf("a is an interger variable ") ;
 cout<<"s1 is a string variable"<<endl ;
 // This is a commented line and My registration Number is 21bcb0107 .
Various operators in cpp are + , - and =
}
```

Output Screenshot:



```
a is an identifier
commented is an identifier
line is an identifier
and is an identifier
My is an identifier
registration is an identifier
Number is an identifier
is is an identifier
21bcb0107 is a constant
. is an identifier
Various is an identifier
operators is an identifier
in is an identifier
cpp is an identifier
are is an identifier
+ is an operator
, is a symbol
- is an operator
and is an identifier
= is an operator
} is a symbol
The number of Keywords is: 10
The number of Symbols is: 8
The number of constants is: 1
The number of identifiers is: 30
The number of operators is: 3

--------------------------------
Process exited after 0.03829 seconds with return value 0
Press any key to continue . . .
```

Code for a particular user input code instead of a file:

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>

// Token types
typedef enum {
   TOK_IDENTIFIER,
   TOK_NUMBER,
   TOK_OPERATOR,
   TOK_DELIMITER,
   TOK_KEYWORD,
   TOK_UNKNOWN
} TokenType;
```

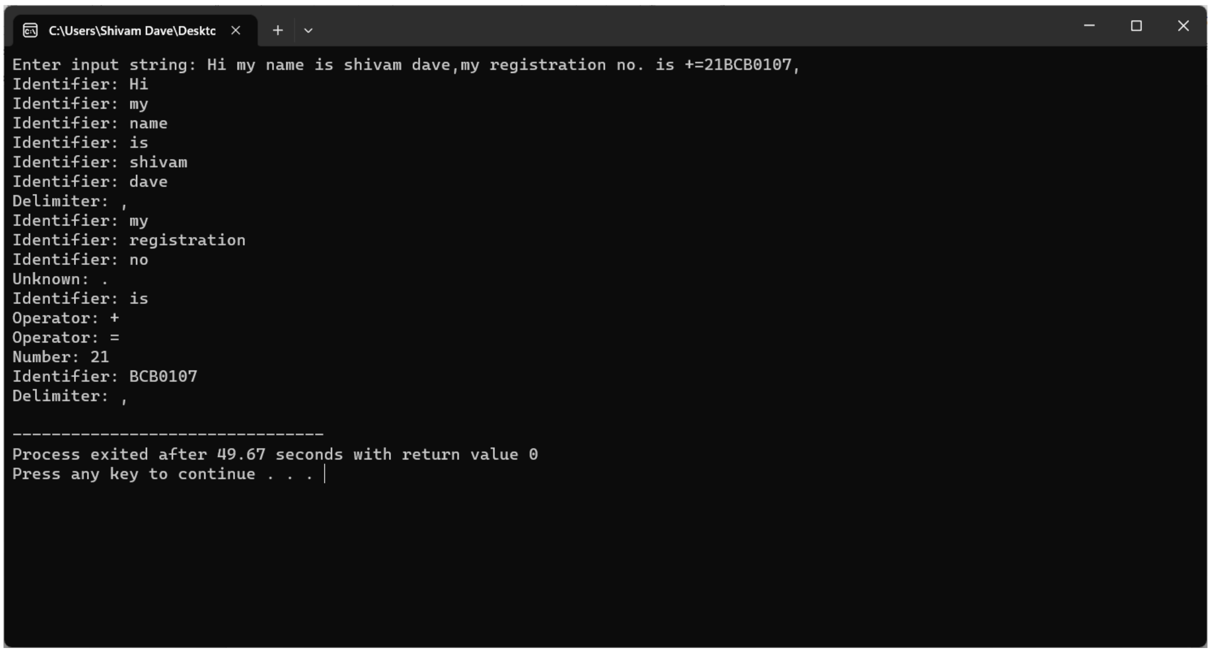
21BCB0107

```c
// Token structure
typedef struct {
   TokenType type;
   char value[50];
} Token;

// Function to check if a character is a delimiter
int isDelimiter(char ch) {
   char delimiters[] = " \t\n,;(){}[]";
   int i;
   for ( i = 0; i < strlen(delimiters); i++) {
      if (ch == delimiters[i])
         return 1;
   }
   return 0;
}

// Function to check if a character is an operator
int isOperator(char ch) {
   char operators[] = "+-*/%=";
   int i;
   for (i = 0; i < strlen(operators); i++) {
      if (ch == operators[i])
         return 1;
   }
   return 0;
}

// Function to check if a string is a keyword
int isKeyword(char* str) {
   char keywords[][10] = {"int", "float", "char", "if", "else", "for", "while", "do", "return"};
   int numKeywords = sizeof(keywords) / sizeof(keywords[0]);
   int i;
   for (i = 0; i < numKeywords; i++) {
      if (strcmp(str, keywords[i]) == 0)
         return 1;
   }
   return 0;
}

// Function to tokenize the input string
void tokenize(char* input) {
   int length = strlen(input);
   int i = 0;

   while (i < length) {
      // Skip whitespace
      if (isspace(input[i])) {
         i++;
         continue;
```

```
    }

    // Handle identifiers and keywords
    if (isalpha(input[i])) {
        int j = 0;
        char identifier[50];

        while (isalnum(input[i])) {
            identifier[j] = input[i];
            i++;
            j++;
        }
        identifier[j] = '\0';

        Token token;
        strcpy(token.value, identifier);

        if (isKeyword(identifier)) {
            token.type = TOK_KEYWORD;
            printf("Keyword: %s\n", token.value);
        } else {
            token.type = TOK_IDENTIFIER;
            printf("Identifier: %s\n", token.value);
        }
        continue;
    }

    // Handle numbers
    if (isdigit(input[i])) {
        int j = 0;
        char number[50];

        while (isdigit(input[i])) {
            number[j] = input[i];
            i++;
            j++;
        }
        number[j] = '\0';

        Token token;
        strcpy(token.value, number);
        token.type = TOK_NUMBER;

        printf("Number: %s\n", token.value);
        continue;
    }

    // Handle operators
    if (isOperator(input[i])) {
        Token token;
        token.value[0] = input[i];
```

```
            token.value[1] = '\0';
            token.type = TOK_OPERATOR;

            printf("Operator: %s\n", token.value);
            i++;
            continue;
        }

        // Handle delimiters
        if (isDelimiter(input[i])) {
            Token token;
            token.value[0] = input[i];
            token.value[1] = '\0';
            token.type = TOK_DELIMITER;

            printf("Delimiter: %s\n", token.value);
            i++;
            continue;
        }

        // Handle unknown characters
                        Token token;
                        token.value[0] = input[i];
                        token.value[1] = '\0';
                        token.type = TOK_UNKNOWN;
                printf("Unknown: %s\n", token.value);
                i++;
}
}

// Main function
int main() {
            char input[100];
            printf("Enter input string: ");
            fgets(input, sizeof(input), stdin);

            // Remove trailing newline character
            input[strcspn(input, "\n")] = '\0';

            // Tokenize the input string
            tokenize(input);

            return 0;
}
```

OUTPUT Screenshot:



```
Enter input string: Hi my name is shivam dave,my registration no. is +=21BCB0107,
Identifier: Hi
Identifier: my
Identifier: name
Identifier: is
Identifier: shivam
Identifier: dave
Delimiter: ,
Identifier: my
Identifier: registration
Identifier: no
Unknown: .
Identifier: is
Operator: +
Operator: =
Number: 21
Identifier: BCB0107
Delimiter: ,

--------------------------------
Process exited after 49.67 seconds with return value 0
Press any key to continue . . .
```