System Software

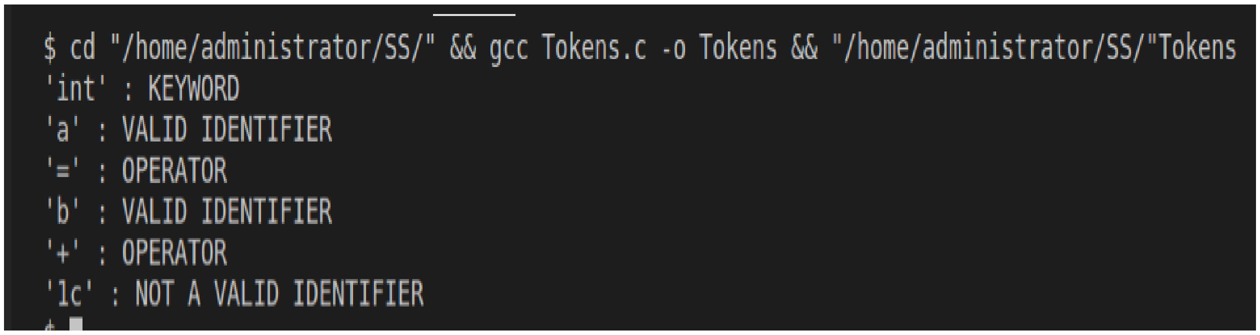
Nehal Jhajharia
Lab Assignment 2

Write a program to detect tokens in c program.

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

// Returns 'true' if the character is a DELIMITER.
bool isDelimiter(char ch) {
if (ch == ' ' || ch == '+' || ch == '-' || ch == '*' ||
        ch == '/' || ch == ',' || ch == ';' || ch == '>' ||
        ch == '<' || ch == '=' || ch == '(' || ch == ')' ||
        ch == '[' || ch == ']' || ch == '{' || ch == '}')
        return (true);
return (false);
}

// Returns 'true' if the character is an OPERATOR.
bool isOperator(char ch) {
if (ch == '+' || ch == '-' || ch == '*' ||
        ch == '/' || ch == '>' || ch == '<' ||
        ch == '=')
        return (true);
return (false);
}

// Returns 'true' if the string is a VALID IDENTIFIER.
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);
}

// Returns 'true' if the string is a KEYWORD.
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);
}

// Returns 'true' if the string is an INTEGER.
bool isInteger(char* str) {
int i = 0;
   int 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);
}

// Returns 'true' if the string is a REAL NUMBER.
bool isRealNumber(char* str) {
int i = 0;
   int 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);
}

// Extracts the SUBSTRING.
char* subString(char* str, int left, int right) {
int i = 0;
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);
}

// Parsing the input string.
void parse(char* str) {
int left = 0;
   int right = 0;
int len = strlen(str);

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

          if (isDelimiter(str[right]) && left == right) {
                    if (isOperator(str[right]) == true)
                              printf("'%c' : OPERATOR\n", str[right]);
                    right++;
                    left = right;
          } else if (isDelimiter(str[right]) && left != right
                              || (right == len && left != right)) {
                    char* subStr = subString(str, left, right - 1);

                    if (isKeyword(subStr))
                              printf("'%s' : KEYWORD\n", subStr);

                    else if (isInteger(subStr))
                              printf("'%s' : INTEGER\n", subStr);

                    else if (isRealNumber(subStr))
                              printf("'%s' : REAL NUMBER\n", subStr);

                    else if (validIdentifier(subStr)
                                        && !isDelimiter(str[right - 1]))
                              printf("'%s' : VALID IDENTIFIER\n", subStr);
                    else if (!validIdentifier(subStr)
                                        && !isDelimiter(str[right - 1]))
                              printf("'%s' : NOT A VALID IDENTIFIER\n", subStr);
                    left = right;
          }
}
return;
}

int main() {
char str[100] = "int a = b + 1c; ";

parse(str);

return (0);
}
```



```
$ cd "/home/administrator/SS/" && gcc Tokens.c -o Tokens && "/home/administrator/SS/"Tokens
'int' : KEYWORD
'a' : VALID IDENTIFIER
'=' : OPERATOR
'b' : VALID IDENTIFIER
'+' : OPERATOR
'1c' : NOT A VALID IDENTIFIER
$
```