**EX 5:** **Develop a lexical Analyzer to test whether a given identifier is valid or not.**

**AIM:**

Develop a lexical analyzer to test whether a given identifier is valid according to the rules of identifier naming in programming.

**Algorithm**

1. **Start**
2. Read the input string or text containing possible identifiers.
3. Initialize counters for whitespace and newline characters if needed.
4. Split the input text into tokens using spaces or special characters.
5. For each token:
   * Check if it matches the pattern for a valid identifier:
     + Begins with a letter (A-Z, a-z) or an underscore (\_).
     + Followed by letters, digits (0-9), or underscores.
   * If the token matches the pattern, consider it a valid identifier.
6. Display the results:
   * Print the valid identifiers.
   * Optionally, show the counts of whitespace and newline characters.
7. **End**

**CODE:**

#include <stdio.h>

#include <ctype.h>

#include <string.h>

#define MAX\_IDENTIFIER\_LENGTH 100

int isValidIdentifier(const char \*identifier) {

if (!isalpha(identifier[0]) && identifier[0] != '\_') {

return 0;

}

for (int i = 1; i < strlen(identifier); i++) {

if (!isalnum(identifier[i]) && identifier[i] != '\_') {

return 0;

}

}

return 1;

}

int main() {

char identifier[MAX\_IDENTIFIER\_LENGTH];

printf("Enter an identifier: ");

scanf("%s", identifier);

if (isValidIdentifier(identifier)) {

printf("'%s' is a valid identifier.\n", identifier);

} else {

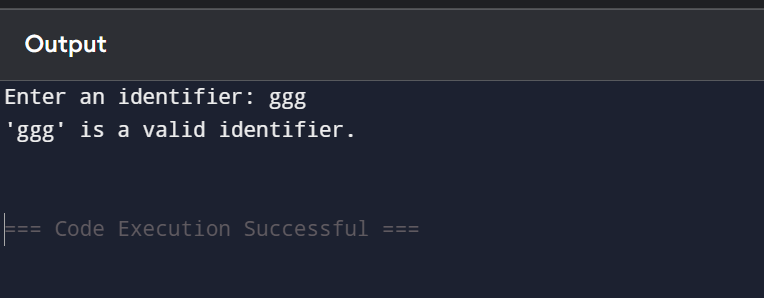
printf("'%s' is not a valid identifier.\n", identifier);

}

return 0;

}

**OUTPUT**

****