**COMPILER DESIGN LAB PROGRAMS**

1. Develop a lexical Analyzer to identify identifiers, constants, operators using C program.

**PROGRAM:**

#include <stdio.h>

#include <ctype.h>

#include <string.h>

#define MAX\_TOKEN\_LENGTH 100

int isIdentifier(char\* str) {

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

return 0;

}

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

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

return 0;

}

}

return 1;

}

int isConstant(char\* str) {

for (int i = 0; i < strlen(str); i++) {

if (!isdigit(str[i])) {

return 0;

}

}

return 1;

}

int isOperator(char c) {

return (c == '+' || c == '-' || c == '\*' || c == '/' || c == '=' || c == '>' || c == '<');

}

void lexicalAnalyzer(char\* input) {

int i = 0;

char token[MAX\_TOKEN\_LENGTH];

int tokenIndex = 0;

while (input[i] != '\0') {

if (isspace(input[i])) {

i++;

continue;

}

if (isOperator(input[i])) {

printf("Operator: %c\n", input[i]);

i++;

continue;

}

if (isdigit(input[i])) {

tokenIndex = 0;

while (isdigit(input[i])) {

token[tokenIndex++] = input[i++];

}

token[tokenIndex] = '\0';

if (isConstant(token)) {

printf("Constant: %s\n", token);

}

continue;

}

if (isalpha(input[i]) || input[i] == '\_') {

tokenIndex = 0;

while (isalnum(input[i]) || input[i] == '\_') {

token[tokenIndex++] = input[i++];

}

token[tokenIndex] = '\0';

if (isIdentifier(token)) {

printf("Identifier: %s\n", token);

}

continue;

}

i++;

}

}

int main() {

char input[] = "int main() { int x = 10 + 5; }";

printf("Lexical Analysis Output:\n");

lexicalAnalyzer(input);

return 0;

}

**OUTPUT**