1. The lexical analyzer should ignore redundant spaces, tabs and new lines. It should also ignore comments. Although the syntax specification states that identifiers can be arbitrarily long, you may restrict the length to some reasonable value. Develop a lexical Analyzer to identify identifiers, constants, operators using C program.

#include <stdio.h>

#include <ctype.h>

#include <string.h>

#define MAX\_IDENTIFIER\_LENGTH 50

int isOperator(char ch) {

char operators[] = "+-\*/%=><&|^~!?:";

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

if (ch == operators[i]) {

return 1;

}

}

return 0;

}

int main() {

char input[1000];

printf("Enter your code:\n");

fgets(input, sizeof(input), stdin);

int i = 0;

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

if (isspace(input[i])) {

i++;

continue;

}

if (input[i] == '/' && input[i+1] == '/') {

break;

}

if (isalpha(input[i])) {

char identifier[MAX\_IDENTIFIER\_LENGTH + 1];

int j = 0;

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

identifier[j++] = input[i++];

}

identifier[j] = '\0';

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

}

else if (isdigit(input[i])) {

char constant[20];

int j = 0;

while (isdigit(input[i]) || input[i] == '.') {

constant[j++] = input[i++];

}

constant[j] = '\0';

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

}

else if (isOperator(input[i])) {

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

}

else {

i++;

}

}

return 0;

}

Output:

