** Comsats University Islamabad, Lahore Campus**

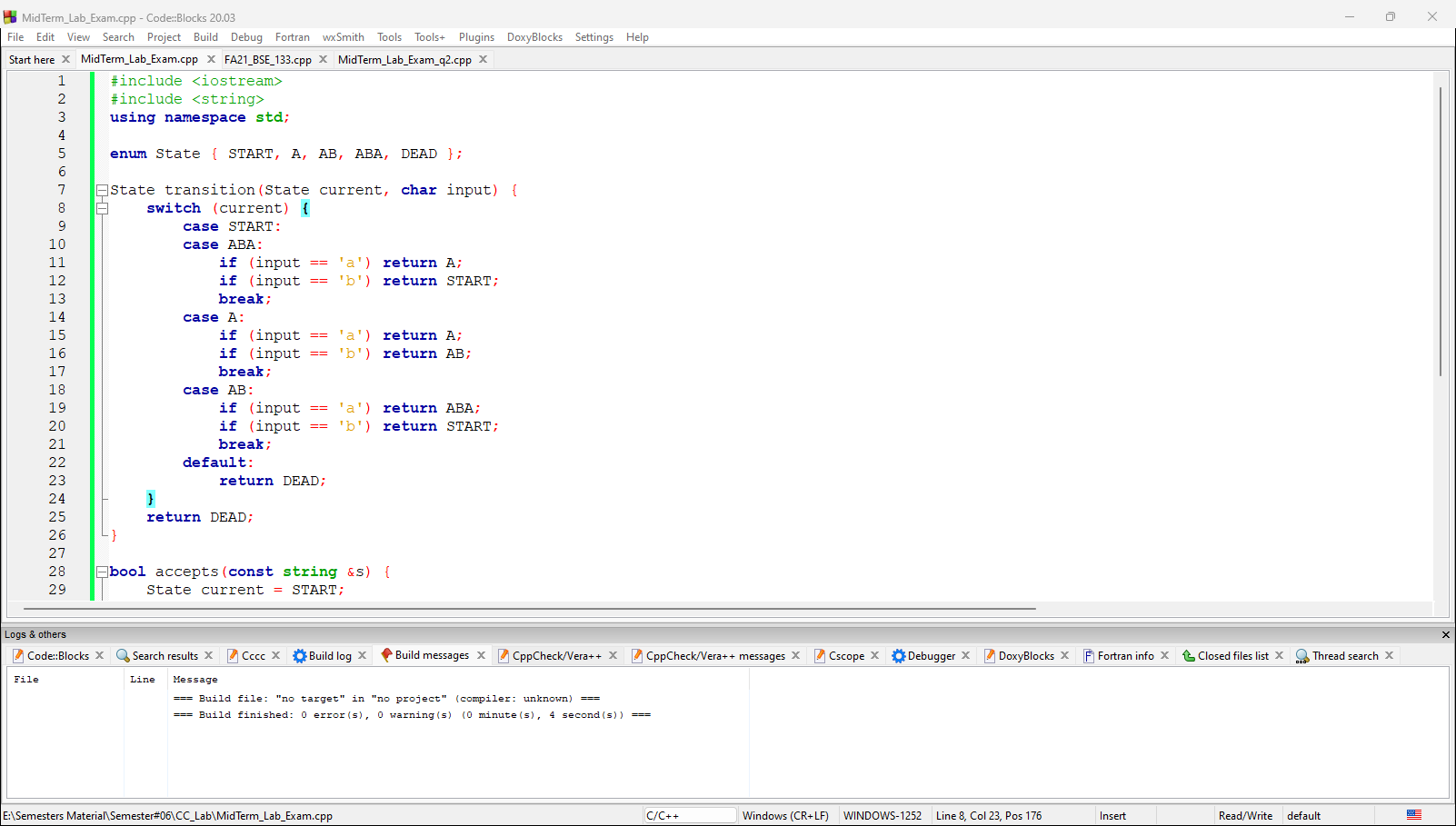
**(Defence Road, Off Raiwind Road, Lahore)**

**Midterm Exam (Lab) – Spring 2024**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Course Title** | Compiler Construction | **Course Code** | | | CSC441 | | **Credit Hours** | | | 3(2,1) |
| **Instructor (s)** | Gul Bano Anwar | **Program Name** | | | BSCS | | | | | |
| **Semester** | 7th | **Batch** | SP21 | **Section** | | B | | **Date** | April 05, 2024 | |
| **Time Allowed** | 90 minutes | **Maximum Marks** | | | 50 | | | | | |
| **Student Name:** | Mr. Aoun Haider | **Reg No.** | | | FA21-BSE-133 | | | | | |
| **Important Instructions/Guidelines:**   1. Submit your solution on time. 2. Internet will only allowed on time of submission. 3. In case of plagiarism direct zero will be marked. 4. Your submission should contain cpp and word file 5. On word file paste your code and place screenshot of output screen. 6. Submit file **SP21\_BCS\_XYZ.cpp.** 7. Return question paper after leaving the lab. | | | | | | | | | | |

**Question no. 1**

**Implement the DFA: (a+b)\*aba(a+b)\***

****

***Code snippet:***

#include <iostream>

#include <string>

using namespace std;

enum State { START, A, AB, ABA, DEAD };

State transition(State current, char input) {

switch (current) {

case START:

case ABA:

if (input == 'a') return A;

if (input == 'b') return START;

break;

case A:

if (input == 'a') return A;

if (input == 'b') return AB;

break;

case AB:

if (input == 'a') return ABA;

if (input == 'b') return START;

break;

default:

return DEAD;

}

return DEAD;

}

bool accepts(const string &s) {

State current = START;

for (char c : s) {

current = transition(current, c);

if (current == DEAD) return false;

}

return current == ABA || current == A || current == START;

}

int main() {

string s;

cout << "Enter a string to check: ";

cin >> s;

if (accepts(s)) {

cout << "The string is accepted by the DFA." << endl;

} else {

cout << "The string is not accepted by the DFA." << endl;

}

return 0;

}

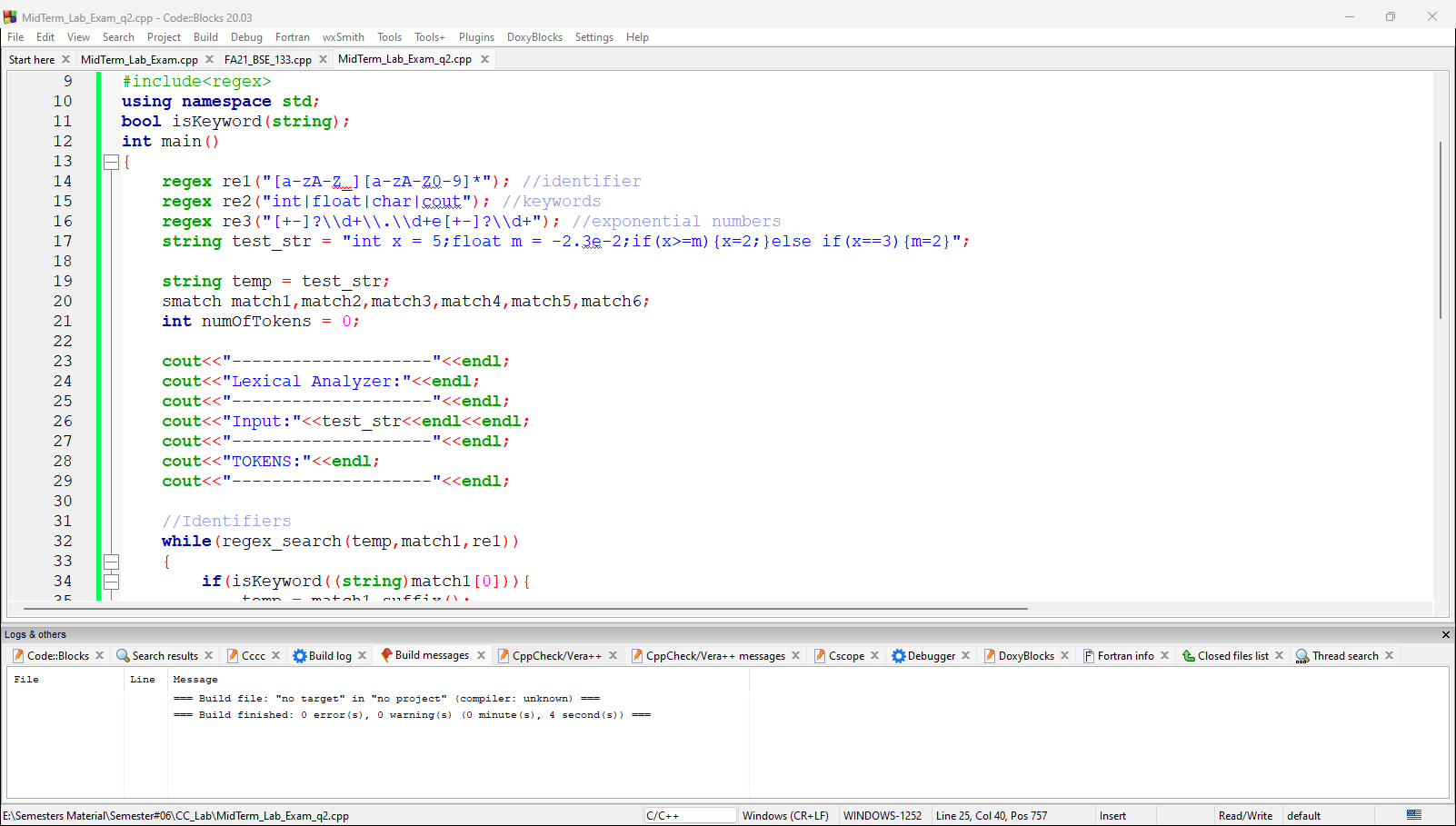
**Question no. 02**

**Implement lexical analyzer that return the tokens for variables/identifiers, exponential and keywords from the input string using the R.E of identifier and following keywords**

**Keywords:** int | float | char | cout

**Output should display in the following order: <Lexem: Tok>**

**Screenshot:**

****

**Code snippet:**

//CSC441 - Compiler Construction

//Midterm Question no.02

//Name: Mr. Aoun-Haider

//ID: FA21-BSE-133

//Submitted to: Mam Gul Bano

//Spring2024

#include<iostream>

#include<string>

#include<regex>

using namespace std;

bool isKeyword(string);

int main()

{

regex re1("[a-zA-Z\_][a-zA-Z0-9]\*"); //identifier

regex re2("int|float|char|cout"); //keywords

regex re3("[+-]?\\d+\\.\\d+e[+-]?\\d+"); //exponential numbers

string test\_str = "int x = 5;float m = -2.3e-2;if(x>=m){x=2;}else if(x==3){m=2}";

string temp = test\_str;

smatch match1,match2,match3,match4,match5,match6;

int numOfTokens = 0;

cout<<"--------------------"<<endl;

cout<<"Lexical Analyzer:"<<endl;

cout<<"--------------------"<<endl;

cout<<"Input:"<<test\_str<<endl<<endl;

cout<<"--------------------"<<endl;

cout<<"TOKENS:"<<endl;

cout<<"--------------------"<<endl;

//Identifiers

while(regex\_search(temp,match1,re1))

{

if(isKeyword((string)match1[0])){

temp = match1.suffix();

continue;

}

cout<<"<identifier:"<<match1[0]<<">"<<endl;

temp = match1.suffix();

numOfTokens++;

}

temp = test\_str;

//Keywords

while(regex\_search(temp,match2,re2))

{

cout<<"<keyword:"<<match2[0]<<">"<<endl;

temp = match2.suffix();

numOfTokens++;

}

temp = test\_str;

//Float numbers

while(regex\_search(temp,match4,re3))

{

cout<<"<exp-number:"<<match4[0]<<">"<<endl;

temp = match4.suffix();

numOfTokens++;

}

temp = test\_str;

cout<<"--------------------"<<endl;

cout<<"Total # of tokens:"<<numOfTokens<<endl;

cout<<"--------------------"<<endl;

return 0;

}

bool isKeyword(string token)

{

if(token == "int" || token == "float" || token == "char"|| token == "cout")

return true;

return false;

}