**LAB (7) TASK (1)**

**Code:**

using System;

using System.Collections.Generic;

class Program {

static Queue<string> tokens = new Queue<string>();

static void Main() {

Console.WriteLine("Enter input (e.g., int x = 10;):");

string input = Console.ReadLine();

Tokenize(input);

bool result = S();

if (result && tokens.Count == 0)

Console.WriteLine("Input is valid according to grammar.");

else

Console.WriteLine("Invalid input according to grammar."); }

static void Tokenize(string input) {

string[] parts = input.Split(new char[] { ' ', '\t' },

StringSplitOptions.RemoveEmptyEntries);

foreach (string part in parts) {

if (part == "int")

tokens.Enqueue("int");

else if (part == "=")

tokens.Enqueue("=");

else if (part == ";")

tokens.Enqueue(";");

else if (int.TryParse(part, out \_))

tokens.Enqueue("number");

else

tokens.Enqueue("id");

} }

static bool S() {

return A() && B(); }

static bool A() {

if (tokens.Count >= 2 && tokens.Peek() == "int") {

tokens.Dequeue();

if (tokens.Peek() == "id") {

tokens.Dequeue();

return true;

} }

return false; }

static bool B() {

if (tokens.Count >= 3 && tokens.Peek() == "=") {

tokens.Dequeue(); // consume '='

if (C()) {

if (tokens.Peek() == ";") {

tokens.Dequeue();

return true;

} } }

return false;

}

static bool C() {

if (tokens.Count > 0 && (tokens.Peek() == "number" || tokens.Peek() == "id")) {

tokens.Dequeue();

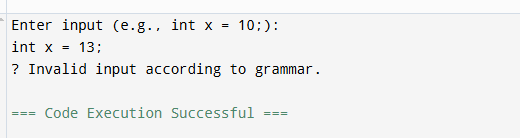
return true;

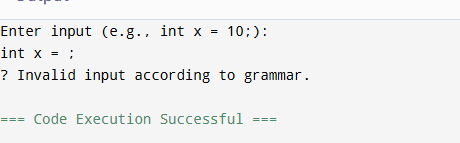
}

return false;

} }

**Output:**

****

****