**LAB TASK 1**using System;

using System.Text.RegularExpressions;

public class Program

{

public static void Main()

{

string password = "Sp22-bcs-014";

// Regular expression pattern for the requirements

string pattern = @"^(?=(.\*\d.\*){2})(?=.\*[A-Z])(?=(.\*[a-z]){4})(?=(.\*[-!@#$%^&\*(),.?\""{}|<>]){2}).{1,12}$";

// Check if the password matches the pattern

if (Regex.IsMatch(password, pattern))

{

Console.WriteLine("Password is valid.");

}

else

{

Console.WriteLine("Password is invalid.");

}

}

}

**LAB TASK 2**

using System;

using System.Text;

using System.Text.RegularExpressions;

class Program

{

static void Main(string[] args)

{

// Sample inputs

Console.WriteLine("Enter your first name: ");

string firstName = Console.ReadLine();

Console.WriteLine("Enter your last name: ");

string lastName = Console.ReadLine();

Console.WriteLine("Enter your registration number: ");

string regNumber = Console.ReadLine();

Console.WriteLine("Enter your favorite food: ");

string food = Console.ReadLine();

Console.WriteLine("Enter your favorite game: ");

string game = Console.ReadLine();

// Generate the password

string password = GeneratePassword(firstName, lastName, regNumber, food, game);

// Display the generated password

Console.WriteLine("Generated Password: " + password);

}

static string GeneratePassword(string firstName, string lastName, string regNumber, string food, string game)

{

// Combine all input values

string combined = firstName + lastName + regNumber + food + game;

// Regular expression to remove any unwanted characters (non-alphanumeric)

string sanitized = Regex.Replace(combined, @"[^a-zA-Z0-9]", "");

// Make the string more complex by adding special characters and digits

string complexPassword = sanitized;

// Add some random numbers and special characters

Random rand = new Random();

string specialChars = "!@#$%^&\*()\_+[]{}|;:,.<>?/~`";

for (int i = 0; i < 4; i++)

{

// Add random number

complexPassword += rand.Next(0, 10).ToString();

// Add random special character

complexPassword += specialChars[rand.Next(specialChars.Length)];

}

// Ensure password length is at least 12 characters

if (complexPassword.Length < 12)

{

complexPassword = complexPassword.PadLeft(12, 'X'); // Add filler 'X' if too short

}

// Randomly shuffle the password to increase complexity

StringBuilder shuffledPassword = new StringBuilder();

while (complexPassword.Length > 0)

{

int index = rand.Next(complexPassword.Length);

shuffledPassword.Append(complexPassword[index]);

complexPassword = complexPassword.Remove(index, 1);

}

return shuffledPassword.ToString();

}

}