

**Lab Mid**

Name:

Areeba Eman

Reg. #:

Fa21-BCS-025

Course:

Compiler Construction

Instructors:

Sir Bilal Haider Bukhari

**Question # 02: Create a password generator using regex with the following specs,**

**• Contains 2 digits of your registration number**

**• Contains second letters from your first name and last name**

**• Contains 2 characters from your favourite movie**

**• Contains special characters**

**• Does not contain # sign**

**• Length should be exactly 14**

**Code:**

using System;

using System.Text.RegularExpressions;

public class PasswordGenerator

{

public static string GeneratePassword(string registrationNumber, string firstName, string lastName, string movie)

{

// Step 1: Extract two digits from the registration number

string regDigits = Regex.Match(registrationNumber, @"\d{2}").Value;

// Step 2: Extract the second letters from first and last names

string secondLetterFirstName = firstName.Length >= 2 ? firstName[1].ToString() : "";

string secondLetterLastName = lastName.Length >= 2 ? lastName[1].ToString() : "";

// Step 3: Extract two characters from favorite movie

string movieChars = movie.Length >= 2 ? movie.Substring(0, 2) : "";

// Step 4: Include special characters, avoiding '#'

string specialChars = "@$%&\*";

// Combine elements

string combined = regDigits + secondLetterFirstName + secondLetterLastName + movieChars + specialChars;

// Adjust the password length to 14

if (combined.Length < 14)

{

combined += new string('X', 14 - combined.Length); // Fill with 'X' if needed

}

else if (combined.Length > 14)

{

combined = combined.Substring(0, 14); // Trim to 14 characters

}

// Validate length and lack of '#' character

if (combined.Length == 14 && !combined.Contains("#"))

{

return combined;

}

else

{

throw new Exception("Password generation failed.");

}

}

public static void Main()

{

// Example usage

string registrationNumber = "25"; // Replace with actual registration number

string firstName = "Areeba"; // Replace with actual first name

string lastName = "Eman"; // Replace with actual last name

string movie = "World War Z"; // Replace with actual favorite movie

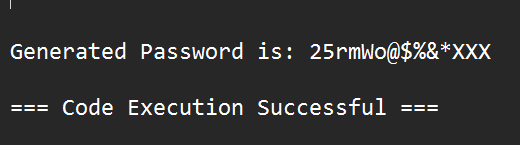
string password = GeneratePassword(registrationNumber, firstName, lastName, movie);

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

}

}

**Output:**

****

**Question # 03: Ziyad bought a toy remote car, the car has a controller and can pass the following commands start, stop accelerate, right turn. Ziyad was able to find that the car does not turn left. Create a context free grammer to help ziyad sort out the issue with his car**

**Non Terminals (Goal Symbol, CMD, Action)**

**Terminal (Start, Stop, Accelerate, Brake, Left, Right)**

**Code:**

using System;

using System.Collections.Generic;

public class RemoteCar

{

private static HashSet<string> validCommands = new HashSet<string>

{

"Start", "Stop", "Accelerate", "Brake", "Right", "Left"

};

public static void Main()

{

Console.WriteLine("Enter commands (separated by commas): ");

string input = Console.ReadLine();

if (input != null)

{

string[] commands = input.Split(',');

foreach (var command in commands)

{

string trimmedCommand = command.Trim();

if (IsValidCommand(trimmedCommand))

{

Console.WriteLine($"Command '{trimmedCommand}' is valid.");

}

else

{

Console.WriteLine($"Command '{trimmedCommand}' is invalid. The car cannot turn left.");

}

}

}

}

private static bool IsValidCommand(string command)

{

// Command can be either a valid command or an action

if (command == "Start" || command == "Stop" || command == "Accelerate" || command == "Brake" || command == "Right")

{

return true; // Valid command

}

else if (command == "Left")

{

return false; // Invalid action as the car does not turn left

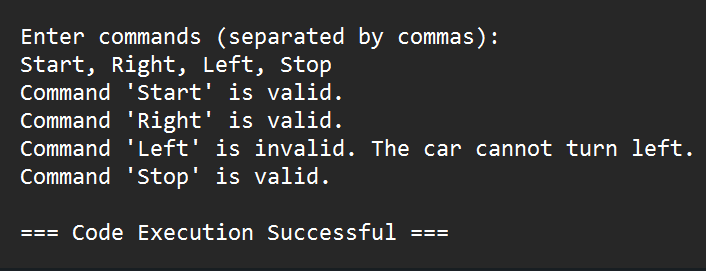
}

return false; // Any other command is also invalid

}

}

**Output:**

****