using System;

using System.Collections.Generic;

using System.Text;

using System.Text.RegularExpressions;

namespace Dobro.Text.RegularExpressions

{

class EmailValidator

{

static void Main(string[] args)

{

string checkOutput = default(string);

string[] emails = new string[] {

"shahrukh@123test.com", "123@shjdsssss",

"ab@222202020202","aa.cc.11@alpha1\_ab.com" };

if (args.Length > 0)

{

emails = args;

}

foreach (string email in emails)

{

if (TestEmail.IsEmail(email))

{

checkOutput = "valid :)";

}

else

{

checkOutput = "invalid :(";

}

Console.WriteLine(

String.Format("\nThe Given Email Address \"{0}\" is {1}", email, checkOutput));

}

Wait4User2Exit();

}

private static void Wait4User2Exit()

{

Console.WriteLine("\nPress any key to exit ...");

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Text;

using System.Text.RegularExpressions;

namespace Dobro.Text.RegularExpressions

{

public static class TestEmail

{

public const string MatchEmailPattern =

@"^(([\w-]+\.)+[\w-]+|([a-zA-Z]{1}|[\w-]{2,}))@"

+ @"((([0-1]?[0-9]{1,2}|25[0-5]|2[0-4][0-9])\.([0-1]?

[0-9]{1,2}|25[0-5]|2[0-4][0-9])\."

+ @"([0-1]?[0-9]{1,2}|25[0-5]|2[0-4][0-9])\.([0-1]?

[0-9]{1,2}|25[0-5]|2[0-4][0-9])){1}|"

+ @"([a-zA-Z0-9]+[\w-]+\.)+[a-zA-Z]{1}[a-zA-Z0-9-]{1,23})$";

public static bool IsEmail(string email)

{

if (email != null) return Regex.IsMatch(email, MatchEmailPattern);

else return false;

}

}

}