using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.InteropServices;

using System.Runtime.Serialization;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

namespace ConsoleApp3

{

internal class Program

{

static public void ValidateEmail(string str)

{

string pattern = @"^(?!\.)(""([^""\r\\]|\\[""\r\\])\*""|"

+ @"([-a-z0-9!#$%&'\*+/=?^\_`{|}~]|(?<!\.)\.)\*)(?<!\.)"

+ @"@[a-z0-9][\w\.-]\*[a-z0-9]\.[a-z][a-z\.]\*[a-z]$";

Regex regex = new Regex(pattern, RegexOptions.IgnoreCase);

if (!regex.IsMatch(str))

{

throw new InvalidEmail("Invalid e-mail");

}

}

private static void Main(string[] args)

{

Console.ReadKey();

}

private static void FirstExercise()

{

try

{

int[] arr = { 1, 2, 3, 4 };

Console.WriteLine(arr[4]);

}

catch (IndexOutOfRangeException ex)

{

Console.WriteLine(ex.StackTrace);

}

}

private static void SecondExercise()

{

try

{

int a = 1, b = 0;

Console.WriteLine(a / b);

}

catch

{

Console.WriteLine("No idea of what happened, but it must be something bad");

}

}

private static void ThirdExercise()

{

try

{

DivideByZeroExeptionEmulator();

}

catch (Exception ex)

{

Console.WriteLine("Message\n" + ex.Message + "\nStack\n" + ex.StackTrace);

}

}

private static void FourExercise()

{

try

{

Console.WriteLine("The code should be written here...");

}

finally

{

Console.WriteLine("Useless piece of .... code");

}

try

{

throw new Exception("143");

}

catch (Exception ex)

{

Console.WriteLine($"-Who are you?\n-I`m error, my name is {ex.GetType()}, and i have a message for you - '{ex.Message}'");

}

finally

{

Console.WriteLine("An error has occurred, but 'finally' caused, fantastic");

}

}

private static void FifthExercise()

{

try

{

ValidateEmail("abc@example.com");

}

catch (InvalidEmail ex)

{

Console.WriteLine(ex.Message);

}

finally

{

Console.WriteLine("Useless piece of .... code");

}

}

private static void SixthExercise()

{

try

{

string str = null;

Console.WriteLine(str.Contains(":)"));

ValidateEmail(str);

Nullable<Int32>[] integers = { null, 345, 43 };

var selected = integers.Select(integer => (int)integer);

foreach (var integer in selected)

{

Console.WriteLine(integers);

}

}

catch (InvalidOperationException ex)

{

Console.WriteLine("ghgfh");

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

}

private static void DivideByZeroExeptionEmulator()

{

try

{

int i = 1, b = 0; ;

Console.WriteLine(i / b);

}

catch (DivideByZeroException ex)

{

Console.WriteLine(ex.Message + "\nThe error was occured in -> " + ex.StackTrace);

throw new DivideByZeroException("DivideByZeroExeptionEmulator error", ex);

}

}

}

public class Person

{

}

internal class InvalidEmail : Exception

{

public InvalidEmail(string message)

: base(message)

{

}

public InvalidEmail(string message, Exception innerException)

: base(message, innerException)

{

}

public override string Message => base.Message;

}

}