# Chapter 12

## Gerti Gonxhi

7.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace CheckInputData

{

class CheckInputData

{

static void Main(string[] args)

{

uint number = 0;

try

{

number = uint.Parse(Console.ReadLine());

double result = Math.Sqrt(number);

Console.WriteLine(result);

}

catch (OverflowException)

{

Console.WriteLine("Invalid number!");

}

catch (FormatException)

{

Console.WriteLine("Invalid number!");

}

finally

{

Console.WriteLine("Good bye!");

}

}

}

}

8.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.IO;

namespace ReadNumbers

{

class ReadNumbers

{

static void Main(string[] args)

{

int start = 1;

int end = 100;

byte count = 0;

int previousNumber = 0;

while (count < 10)

{

try

{

int nexNumber = ReadNumber(start, end);

if (previousNumber >= nexNumber)

{

throw new ArgumentOutOfRangeException();

}

previousNumber = nexNumber;

count++;

}

catch (ArgumentOutOfRangeException smallNumber)

{

Console.WriteLine(smallNumber.Message);

}

}

}

static int ReadNumber(int start, int end)

{

int number = 0;

try

{

string input = Console.ReadLine();

number = int.Parse(input);

if ((number < start) || (number > end))

{

throw new ArgumentOutOfRangeException();

}

}

catch (OutOfMemoryException outOfMemory)

{

Console.WriteLine(outOfMemory.Message);

}

catch (ArgumentOutOfRangeException outOfRange)

{

Console.WriteLine(outOfRange.Message);

}

catch (ArgumentNullException nullExeption)

{

Console.WriteLine(nullExeption.Message);

}

catch (FormatException formatEx)

{

Console.WriteLine(formatEx.Message);

}

catch (OverflowException overflow)

{

Console.WriteLine(overflow.Message);

}

catch (IOException IO)

{

Console.WriteLine(IO.Message);

}

catch (Exception general)

{

Console.WriteLine(general.Message);

}

return number;

}

}

}

9.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.IO;

using System.Security;

namespace FileContentToString

{

class FileContentToString

{

static void Main(string[] args)

{

string errorMessage = null;

string filePath = Console.ReadLine();

try

{

Console.WriteLine(FileToString(filePath));

}

catch (ArgumentNullException ane)

{

errorMessage = ane.Message;

}

catch (ArgumentException ae)

{

errorMessage = ae.Message;

}

catch (PathTooLongException ptle)

{

errorMessage = ptle.Message;

}

catch (DirectoryNotFoundException dnfe)

{

errorMessage = dnfe.Message;

}

catch (FileNotFoundException fnfe)

{

errorMessage = fnfe.Message;

}

catch (IOException ioe)

{

errorMessage = ioe.Message;

}

catch (UnauthorizedAccessException uae)

{

errorMessage = uae.Message;

}

catch (NotSupportedException nse)

{

errorMessage = nse.Message;

}

catch (SecurityException se)

{

errorMessage = se.Message;

}

catch (Exception e)

{

errorMessage = e.Message;

}

finally

{

if (errorMessage != null)

{

Console.WriteLine(errorMessage);

}

}

}

private static string FileToString(string filePath)

{

string toString = File.ReadAllText(filePath, Encoding.UTF8);

return toString;

}

}

}

10.

using System;

using System.IO;

using System.Security;

using System.Text;

namespace CompareFiles

{

class CompareFiles

{

static void Main(string[] args)

{

string inputFile = Console.ReadLine();

string outputFile = Console.ReadLine();

string fileContent = ReadFile(inputFile);

byte[] byteArray = BinaryToByteArray(fileContent);

WriteFile(outputFile, byteArray);

}

static string ReadFile(string fileName)

{

string errorMessage = null;

string binaryStr = null;

try

{

binaryStr = File.ReadAllText(fileName, Encoding.UTF8);

}

catch (ArgumentNullException ane)

{

errorMessage = ane.Message;

}

catch (ArgumentException ae)

{

errorMessage = ae.Message;

}

catch (PathTooLongException ptle)

{

errorMessage = ptle.Message;

}

catch (DirectoryNotFoundException dnfe)

{

errorMessage = dnfe.Message;

}

catch (FileNotFoundException fnfe)

{

errorMessage = fnfe.Message;

}

catch (IOException ioe)

{

errorMessage = ioe.Message;

}

catch (UnauthorizedAccessException uae)

{

errorMessage = uae.Message;

}

catch (NotSupportedException nse)

{

errorMessage = nse.Message;

}

catch (SecurityException se)

{

errorMessage = se.Message;

}

catch (Exception e)

{

errorMessage = e.Message;

}

finally

{

if (errorMessage != null) Console.WriteLine(errorMessage);

}

return binaryStr;

}

static byte[] BinaryToByteArray(string fileContent)

{

int bytesCount = fileContent.Length / 8;

byte[] bytes = new byte[bytesCount];

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

bytes[i] = Convert.ToByte(fileContent.Substring(8 \* i, 8), 2);

return bytes;

}

static void WriteFile(string fileName, byte[] fileContent)

{

string errorMessage = null;

try

{

File.WriteAllBytes(fileName, fileContent);

}

catch (ArgumentNullException ane)

{

errorMessage = ane.Message;

}

catch (ArgumentException ae)

{

errorMessage = ae.Message;

}

catch (PathTooLongException ptle)

{

errorMessage = ptle.Message;

}

catch (DirectoryNotFoundException dnfe)

{

errorMessage = dnfe.Message;

}

catch (FileNotFoundException fnfe)

{

errorMessage = fnfe.Message;

}

catch (IOException ioe)

{

errorMessage = ioe.Message;

}

catch (UnauthorizedAccessException uae)

{

errorMessage = uae.Message;

}

catch (NotSupportedException nse)

{

errorMessage = nse.Message;

}

catch (SecurityException se)

{

errorMessage = se.Message;

}

catch (Exception e)

{

errorMessage = e.Message;

}

finally

{

if (errorMessage != null) Console.WriteLine(errorMessage);

}

}

}

}

11.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Diagnostics;

using System.IO;

namespace CustomException

{

class CustomException

{

static void Main(string[] args)

{

string filePath = Console.ReadLine();

string fileName = filePath.Substring(filePath.LastIndexOf('\\') + 1);

try

{

string text = File.ReadAllText(filePath, Encoding.UTF8);

string[] fileContent = text.Split('\n');

long length = fileContent.Length;

for (int line = 0; line < length; line++)

{

ParseLine(fileName, fileContent[line].Trim(), line + 1);

}

}

catch (FileParseException fpe)

{

Console.WriteLine(fpe.Message);

Console.WriteLine(fpe.fileName);

Console.WriteLine(fpe.lineNumber);

}

}

static void ParseLine(string fileName, string lineOfFile, long lineNumber)

{

bool containesNumber = true;

long length = lineOfFile.Length;

for (int symbol = 0; symbol < length; symbol++)

{

if ((lineOfFile[symbol] > 47) && (lineOfFile[symbol] < 58))

{

containesNumber = false;

break;

}

}

if (containesNumber)

{

throw new FileParseException(fileName, lineNumber);

}

}

}

class FileParseException : Exception

{

public string fileName { get; private set; }

public long lineNumber { get; private set; }

public FileParseException()

{

}

public FileParseException(string fileName, long lineNumber)

: base(string.Format("Error while parsing file {0} on line {1}.", fileName, lineNumber))

{

this.fileName = fileName;

this.lineNumber = lineNumber;

}

}

}

12.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.IO;

using System.Security;

namespace FileContentToString

{

class FileContentToString

{

static void Main(string[] args)

{

string errorMessage = null;

string filePath = Console.ReadLine();

try

{

Console.WriteLine(FileToString(filePath));

}

catch (ArgumentNullException ane)

{

errorMessage = ane.Message;

}

catch (ArgumentException ae)

{

errorMessage = ae.Message;

}

catch (PathTooLongException ptle)

{

errorMessage = ptle.Message;

}

catch (DirectoryNotFoundException dnfe)

{

errorMessage = dnfe.Message;

}

catch (FileNotFoundException fnfe)

{

errorMessage = fnfe.Message;

}

catch (IOException ioe)

{

errorMessage = ioe.Message;

}

catch (UnauthorizedAccessException uae)

{

errorMessage = uae.Message;

}

catch (NotSupportedException nse)

{

errorMessage = nse.Message;

}

catch (SecurityException se)

{

errorMessage = se.Message;

}

catch (Exception e)

{

errorMessage = e.Message;

}

finally

{

if (errorMessage != null)

{

Console.WriteLine(errorMessage);

}

}

}

private static string FileToString(string filePath)

{

string toString = File.ReadAllText(filePath, Encoding.UTF8);

return toString;

}

}

}

13.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Net;

namespace DownloadFileFromInternet

{

class DownloadFileFromInternet

{

static void Main(string[] args)

{

WebClient wc = null;

string errorMessage = null;

string url = Console.ReadLine();

try

{

wc = new WebClient();

// Download a file from i-net to the current directory.

wc.DownloadFile(url, @".\google.png");

// Download a file from i-net to parent directory of parent directory.

wc.DownloadFile(url, @"..\..\google.png");

}

catch (WebException we)

{

errorMessage = we.Message;

}

catch (NotSupportedException nse)

{

errorMessage = nse.Message;

}

catch (Exception e)

{

errorMessage = e.Message;

}

finally

{

if (errorMessage != null) Console.WriteLine(errorMessage);

}

}

}

}