

Dokumentimi I detyrave te shtepise.

Kapitulli 13

Introduction to Programming

Prof:Muzafer Shala

Ass:Laberion Zebica Student:Elion Krasniqi

Kampusi:FERIZAJ

**Exercises**

1. Write a program that reads a text file and **prints its odd lines** on the console.

using System;

using System.IO;

class ReadTextFileAndPrintItsOddLines

{

/// <summary>

/// Write a program that reads a text file and

/// prints on the console its odd lines.

/// </summary>

static void Main(string[] args)

{

// Getting Cyrillic encoding

System.Text.Encoding encoding = System.Text.Encoding.GetEncoding(1251);

// Create reader with the Cyrillic encoding

StreamReader streamReader = new StreamReader(

"../../Files/test.001.in.txt", encoding);

// Create writer with the Cyrillic encoding

StreamWriter streamWriter = new StreamWriter(

"../../Files/test.001.out.txt", false, encoding);

using (streamReader)

{

using (streamWriter)

{

int lineCounter = 0;

string oddLineContent = streamReader.ReadLine();

while (oddLineContent != null)

{

lineCounter++;

if (lineCounter % 2 == 1)

{

streamWriter.WriteLine(oddLineContent);

Console.WriteLine(oddLineContent);

}

oddLineContent = streamReader.ReadLine();

}

}

}

}

}

1. Write a program that **joins two text files** and records the results in a third file.

using System;

using System.IO;

using System.Text;

class ConcatenateTwoTextFiles

{

/// <summary>

/// Write a program that concatenates two text files into another text file.

/// </summary>

static void Main(string[] args)

{

// Getting Cyrillic encoding

System.Text.Encoding encoding = System.Text.Encoding.GetEncoding(1251);

// Create two readers with the Cyrillic encoding

StreamReader firstStreamReader = new StreamReader(

"../../Files/test.001.in.1.txt", encoding);

StreamReader secondStreamReader = new StreamReader(

"../../Files/test.001.in.2.txt", encoding);

StringBuilder firstText = new StringBuilder();

using (firstStreamReader)

{

firstText.Append(firstStreamReader.ReadToEnd());

//Console.WriteLine(firstText);

}

StringBuilder secondText = new StringBuilder();

using (firstStreamReader)

{

secondText.Append(secondStreamReader.ReadToEnd());

//Console.WriteLine(secondText);

}

// Create a writer with the Cyrillic encoding

StreamWriter streamWriter = new StreamWriter(

"../../Files/test.001.out.txt", false, encoding);

using (streamWriter)

{

streamWriter.WriteLine(firstText);

streamWriter.WriteLine(secondText);

}

}

}

1. Write a program that reads the contents of a text file and **inserts the line numbers** at the beginning of each line, then rewrites the file contents.

using System;

using System.IO;

using System.Text;

class InsertLineNumbers

{

/// <summary>

/// Write a program that reads a text file and inserts

/// line numbers in front of each of its lines.

/// The result should be written to another text file.

/// </summary>

static void Main(string[] args)

{

// Getting Cyrillic encoding

System.Text.Encoding encoding = System.Text.Encoding.GetEncoding(1251);

string inputFile = string.Empty;

string inputFilePath = "../../Files/test.001.in.txt";

string outputFilePath = "../../Files/test.001.out.txt";

// We convert file to string and use method for adding line numbers

inputFile = FileToString(inputFilePath);

// Create a writer with the Cyrillic encoding

StreamWriter streamWriter = new StreamWriter(

outputFilePath, false, encoding);

streamWriter.Write(inputFile);

streamWriter.Close();

}

public static string FileToString(string filePath)

{

// Getting Cyrillic encoding

System.Text.Encoding encoding = System.Text.Encoding.GetEncoding(1251);

StringBuilder stringBuilder = new StringBuilder();

// Create a reader with the Cyrillic encoding

using (StreamReader streamReader = new StreamReader(

"../../Files/test.001.in.txt", encoding))

{

string line;

int counter = 1;

while ((line = streamReader.ReadLine()) != null)

{

stringBuilder.Append(counter);

stringBuilder.Append(". ");

stringBuilder.Append(line);

if (!streamReader.EndOfStream)

{

stringBuilder.Append(Environment.NewLine);

}

counter++;

}

}

return stringBuilder.ToString();

}

}

1. Write a program that **compares two text files** with the same number of rows line by line and prints the number of equal and the number of different lines.

using System;

using System.IO;

class CompareTextFiles

{

/// <summary>

/// Write a program that compares two text files line by line and prints

/// the number of lines that are the same and the number of lines that

/// are different. Assume the files have equal number of lines.

/// </summary>

static void Main(string[] args)

{

// Create two readers with the Cyrillic encoding

StreamReader firstStreamReader = new StreamReader(

"../../Files/test.020.in.1.txt");

StreamReader secondStreamReader = new StreamReader(

"../../Files/test.020.in.2.txt");

// Create counters for the same and the different text lines

int counterOfEqualLines = 0;

int counterOfDifferentLines = 0;

// Use the two StreamReaders

using (firstStreamReader)

{

using (secondStreamReader)

{

StreamWriter streamWriter = new StreamWriter(

"../../Files/test.020.out.txt", false);

using (streamWriter)

{

// Read the lines of the two text files

string firstFileLines = firstStreamReader.ReadLine();

string secondFileLines = secondStreamReader.ReadLine();

while (secondFileLines != null)

{

// Compare the two text files line by line

if (firstFileLines.CompareTo(secondFileLines) == 0)

{

counterOfEqualLines++;

}

else

{

counterOfDifferentLines++;

}

firstFileLines = firstStreamReader.ReadLine();

secondFileLines = secondStreamReader.ReadLine();

}

streamWriter.WriteLine(counterOfEqualLines);

streamWriter.WriteLine(counterOfDifferentLines);

}

}

}

// Print the number of same and the different lines

Console.WriteLine("The number of lines that are the same is: {0}",

counterOfEqualLines);

Console.WriteLine("The number of lines that are different is: {0}",

counterOfDifferentLines);

}

}

5.     Write a program that reads a square matrix of integers from a file and **finds the sub-matrix with size 2 × 2 that has the maximal sum** and writes this sum to a separate text file. The first line of input file contains the size of the recorded matrix (N). The next N lines contain N integers separated by spaces.

Sample input file:

|  |
| --- |
| 4  2 3 3 4  0 2 3 4  **3 7** 1 2  **4 3** 3 2 |

Sample output: 17.

using System;

using System.IO;

class AreaOfSubMatrixWithMaximalSum

{

/// <summary>

/// Write a program that reads a text file containing a square matrix of numbers

/// and finds in the matrix an area of size 2 x 2 with a maximal sum of its

/// elements. The first line in the input file contains the size of

/// matrix N. Each of the next N lines contain N numbers separated

/// by space. The output should be a single number in a separate

/// text file. Example:

/// 4

/// 2 3 3 4

/// 0 2 3 4 output: 17

/// 3 7 1 2

/// 4 3 3 2

/// </summary>

static void Main(string[] args)

{

// Create reader

StreamReader streamReader = new StreamReader(

"../../Files/test.001.in.txt");

// Create a square matrix of numbers

int[,] squareMatrix;

using (streamReader)

{

// The first line in the input file contains the size of matrix N

string firstLine = streamReader.ReadLine();

int size = int.Parse(firstLine);

// Each of the next N lines contain N numbers separated by space

squareMatrix = new int[size, size];

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

{

firstLine = streamReader.ReadLine();

string[] elements = firstLine.Split(' ');

for (int j = 0; j < size; j++)

{

squareMatrix[i, j] = int.Parse(elements[j]);

}

}

}

// Find a 2 x 2 platform in a matrix with a maximal sum of its elements

int bestSum = int.MinValue;

for (int row = 0; row < squareMatrix.GetLength(0) - 1; row++)

{

for (int col = 0; col < squareMatrix.GetLength(1) - 1; col++)

{

int sum = squareMatrix[row, col] + squareMatrix[row, col + 1] +

squareMatrix[row + 1, col] + squareMatrix[row + 1, col + 1];

if (bestSum < sum)

{

bestSum = sum;

}

}

}

StreamWriter streamWriter = new StreamWriter(

"../../Files/test.001.out.txt", false);

using (streamWriter)

{

// Print the maximal sum of a 2 x 2 platform in a matrix

Console.WriteLine(bestSum);

streamWriter.WriteLine(bestSum);

}

}

}

6.Write a program that **reads a list of names** from a text file, arranges them in **alphabetical order**, and writes them to another file. The lines are written one per row.

using System;

using System.Collections.Generic;

using System.IO;

class SortAndSave

{

/// <summary>

/// Write a program that reads a text file containing a list of strings,

/// sorts them and saves them to another text file. Examle

/// Ivan -> George

/// Peter -> Ivan

/// Maria -> Maria

/// George -> Peter

/// </summary>

static void Main(string[] args)

{

// Getting Cyrillic encoding

System.Text.Encoding encoding = System.Text.Encoding.GetEncoding(1251);

// Save every name into a list

List<string> names = new List<string>();

// Create reader with the Cyrillic encoding

StreamReader streamReader = new StreamReader(

"../../Files/test.001.in.txt", encoding);

using (streamReader)

{

string line = streamReader.ReadLine();

while (line != null)

{

names.Add(line);

line = streamReader.ReadLine();

}

}

names.Sort();

// Create writer with the Cyrillic encoding

StreamWriter streamWriter = new StreamWriter(

"../../Files/test.001.out.txt", false, encoding);

using (streamWriter)

{

for (int i = 0; i < names.Count; i++)

{

streamWriter.WriteLine(names[i]);

Console.WriteLine(names[i]);

}

}

}

}

7.     Write a program that **replaces every occurrence of the substring** "**start**" with "**finish**" in a text file. Can you rewrite the program to replace whole words only? Does the program work for large files (e.g. 800 MB)?

using System;

using System.IO;

using System.Text;

using System.Text.RegularExpressions;

class ReplaceString

{

/// <summary>

/// Write a program that replaces all occurrences of the substring

/// "start" with the substring "finish" in a text file.

/// Ensure it will work with large files (e.g. 100 MB).

/// </summary>

static void Main(string[] args)

{

// Getting Cyrillic encoding

System.Text.Encoding encoding = System.Text.Encoding.GetEncoding(1251);

string filePath = "../../Files/test.001.in.txt";

string fileContent = string.Empty;

fileContent = FileToString(filePath);

fileContent = Regex.Replace(fileContent, "start", "finish");

StreamWriter sw = new StreamWriter(

"../../Files/test.001.out7.txt", false, encoding);

sw.Write(fileContent);

sw.Close();

}

public static string FileToString(string filePath)

{

// Getting Cyrillic encoding

System.Text.Encoding encoding = System.Text.Encoding.GetEncoding(1251);

StringBuilder sb = new StringBuilder();

StreamReader sr = new StreamReader("../../Files/test.001.in.txt", encoding);

using (sr)

{

string line;

while ((line = sr.ReadLine()) != null)

{

sb.Append(line);

if (!sr.EndOfStream)

{

sb.Append(Environment.NewLine);

}

}

}

return sb.ToString();

}

}

8.     Write the previous program so that it changes only the **whole words** (not parts of the word).

using System;

using System.IO;

using System.Text;

using System.Text.RegularExpressions;

class ReplaceString

{

/// <summary>

/// Task7

/// Write a program that replaces all occurrences of the substring

/// "start" with the substring "finish" in a text file.

/// Ensure it will work with large files (e.g. 100 MB).

/// Task8

/// Modify the solution of the previous problem to replace

/// only whole words (not substrings).

/// </summary>

static void Main(string[] args)

{

// Getting Cyrillic encoding

System.Text.Encoding encoding = System.Text.Encoding.GetEncoding(1251);

string filePath = "../../Files/test.001.in.txt";

string fileContent = string.Empty;

fileContent = FileToString(filePath);

fileContent = Regex.Replace(fileContent, @"(\s|^)start(\s)", "$1finish$2");

fileContent = Regex.Replace(fileContent, @"\bstart\b", "finish");

StreamWriter sw = new StreamWriter(

"../../Files/test.001.out8.txt", false, encoding);

sw.Write(fileContent);

sw.Close();

}

public static string FileToString(string filePath)

{

// Getting Cyrillic encoding

System.Text.Encoding encoding = System.Text.Encoding.GetEncoding(1251);

StringBuilder sb = new StringBuilder();

StreamReader sr = new StreamReader("../../Files/test.001.in.txt", encoding);

using (sr)

{

string line;

while ((line = sr.ReadLine()) != null)

{

sb.Append(line);

if (!sr.EndOfStream)

{

sb.Append(Environment.NewLine);

}

}

}

return sb.ToString();

}

}

9.     Write a program that **deletes all the odd lines** of a text file.

using System;

using System.IO;

using System.Text;

class DeleteAllOddLinesInTheSameFile

{

/// <summary>

/// Write a program that deletes from given

/// text file all odd lines. The result should be in the same file.

/// </summary>

static void Main(string[] args)

{

// Getting Cyrillic encoding

System.Text.Encoding encoding = System.Text.Encoding.GetEncoding(1251);

// Create reader with the Cyrillic encoding

StreamReader streamReader = new StreamReader(

"../../Files/test.001.out.txt", encoding);

// Using StringBuilder

StringBuilder text = new StringBuilder();

using (streamReader)

{

int lineCounter = 0;

// Find odd lines of the text file

string oddLineContent = streamReader.ReadLine();

while (oddLineContent != null)

{

lineCounter++;

if (lineCounter % 2 == 0)

{

text.AppendLine(oddLineContent);

}

oddLineContent = streamReader.ReadLine();

}

}

// Create writer with the Cyrillic encoding

StreamWriter streamWriter = new StreamWriter(

"../../Files/test.001.out.txt", false, encoding);

using (streamWriter)

{

streamWriter.Write(text.ToString());

}

}

}

10.   Write a program that extracts from an XML file the **text only** (without the tags). Sample input file:

|  |
| --- |
| <?xml version="1.0"><student><name>**Peter**</name> <age>**21**</age><interests count="3"><interest> **Games**</interest><interest>**C#**</interest><interest> **Java**</interest></interests></student> |

Sample output:

|  |
| --- |
| Peter  21  Games  C#  Java |

using System;

using System.Linq;

using System.Xml;

using System.IO;

using System.Net;

class SimpleXmlTagRemover

{

static void Main(string[] args)

{

try

{

string filePath = "../../Files/test.001.in.xml";

// Create XML Reader

XmlTextReader xmlReader = new XmlTextReader(filePath);

// We check the whole file and

// whenever we have text without tags we write it in the console

while (xmlReader.Read())

{

switch (xmlReader.NodeType)

{

case XmlNodeType.Text:

Console.WriteLine("{0}", xmlReader.Value);

break;

}

}

}

// Catch all possible exceptions

catch (IOException ex)

{

Console.WriteLine(ex.Message);

}

catch (WebException ex)

{

Console.WriteLine(ex.Message);

}

catch (UriFormatException ex)

{

Console.WriteLine(ex.Message);

}

catch (FormatException ex)

{

Console.WriteLine(ex.Message);

}

catch (ArgumentException ex)

{

Console.WriteLine(ex.Message);

}

catch (XmlException ex)

{

Console.WriteLine(ex.Message);

}

}

}

11.   Write a program that **deletes all words** that begin with the word "**test**". The words will contain only the following chars: 0…9, a…z, A…Z.

using System;

using System.IO;

using System.Text.RegularExpressions;

namespace RemoveStartsWith

{

class RemoveWordsWithPrefix

{

/// <summary>

/// Write a program that deletes from a text file

/// all words that start with the prefix "test".

/// Words contain only the symbols 0...9, a...z, A…Z, \_.

/// </summary>

static void Main(string[] args)

{

StreamReader reader = new StreamReader("../../Files/test.001.in.txt");

string file = reader.ReadToEnd();

reader.Close();

file = Regex.Replace(file, @"\btest\w+", "");

file = Regex.Replace(file, @"\btest\b", "");

StreamWriter writer = new StreamWriter("../../Files/test.001.out.txt");

writer.Write(file);

writer.Close();

}

}

}

12.   A text file **words.txt** is given, containing a list of words, one per line. Write a program that **deletes in the file text.txt all the words that occur in the other file**. Catch and handle all possible exceptions.

using System;

using System.IO;

using System.Text;

class RemoveAllListedGivenWords

{

/// <summary>

/// Write a program that removes from a text

/// file all words listed in given another text file.

/// Handle all possible exceptions in your methods.

/// </summary>

static void Main(string[] args)

{

char[] separators = { ' ', ',', '.', '!',

'(', ')', '{', '}', '?',

'/', '\n', '\t', '\r' };

StringBuilder resultText = new StringBuilder();

StringBuilder currentWord = new StringBuilder();

try

{

using (StreamReader textFile = new StreamReader(

"../../Files/test.001.in.txt"))

{

using (StreamReader wordsFile = new StreamReader(

"../../Files/test.001.in.words.txt"))

{

string[] words = wordsFile.ReadToEnd().Split(separators);

char symbol;

while (textFile.Peek() != -1)

{

symbol = (char)textFile.Read();

if (IsIn(symbol, separators))

{

if (IsIn(currentWord.ToString(), words))

{

resultText.Append(symbol);

currentWord.Clear();

}

else

{

resultText.Append(currentWord.ToString());

currentWord.Clear();

resultText.Append(symbol);

}

}

else

{

currentWord.Append(symbol);

}

}

}

}

using (StreamWriter outFile = new StreamWriter(

"../../Files/test.001.out.txt", false))

{

outFile.Write(resultText.ToString());

}

}

catch (FileNotFoundException ex)

{

Console.WriteLine(ex.Message);

}

catch (FileLoadException ex)

{

Console.WriteLine(ex.Message);

}

catch (FieldAccessException ex)

{

Console.WriteLine(ex.Message);

}

catch (IOException ex)

{

Console.WriteLine(ex.Message);

}

catch (OutOfMemoryException ex)

{

Console.WriteLine(ex.Message);

}

catch (ArgumentException ex)

{

Console.WriteLine(ex.Message);

}

}

static bool IsIn<T>(T word, T[] words)

{

foreach (T elem in words)

{

if (elem.Equals(word))

{

return true;

}

}

return false;

}

}

13.   Write a program that **reads a list of words** from a file called **words.txt**, **counts how many times each of these words is found in another file** **text.txt**, and records the results in a third file – **result.txt**, but before that, sorts them by the number of occurrences in descending order. Handle all possible exceptions.

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

class CountWordsAndSortByNumberOfOccurrences

{

/// <summary>

/// Write a program that reads a list of words from a file words.txt

/// and finds how many times each of the words is contained in another

/// file test.txt. The result should be written in the file result.txt

/// and the words should be sorted by the number of their occurrences

/// in descending order. Handle all possible exceptions in your methods.

/// </summary>

static void Main(string[] args)

{

try

{

ReadAndCountWords("../../Files/test.001.in.test.txt",

"../../Files/test.001.in.words.txt",

"../../Files/test.001.out.result.txt");

}

catch (FileNotFoundException exception)

{

Console.WriteLine(exception.Message);

}

}

static void ReadAndCountWords(string inputTextFile,

string listOfWords, string outputTextFile)

{

using (StreamReader textFile = new StreamReader(inputTextFile))

{

using (StreamReader wordsFile = new StreamReader(listOfWords))

{

using (StreamWriter resultFile = new StreamWriter(outputTextFile))

{

string fileContent = textFile.ReadToEnd();

char[] splitSymbols = { ' ', ',', '.', ';', ':',

'?', '!', '-', '(',

')', '\n', '\r', '\t' };

string[] words = fileContent.Split(splitSymbols,

StringSplitOptions.RemoveEmptyEntries);

string[] keyWords = wordsFile.ReadToEnd().Split(splitSymbols,

StringSplitOptions.RemoveEmptyEntries);

Dictionary<string, int> dict = new Dictionary<string, int>();

Dictionary<string, int> result = new Dictionary<string, int>();

foreach (string keyWord in keyWords)

{

dict.Add(keyWord, 0);

}

foreach (var item in dict)

{

int counter = 0;

foreach (string word in words)

{

if (word.Contains(item.Key))

{

counter++;

}

}

result.Add(item.Key, counter);

}

var query = from table in result

orderby table.Value descending

select new { table.Key, table.Value };

foreach (var row in query)

{

resultFile.WriteLine("{0} --> {1} times",

row.Key, row.Value);

}

}

}

}

}

}