Rrogram.cs

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PZ\_2

{

class Program

{

static void Main(string[] args)

{

bool exit = false;

WorkWhithFile.ChangeFile changeFile = new WorkWhithFile.ChangeFile();

WorkWhithFile.WorkWithData wWD;

string name="";

string fileforcoding = "";

List<string> Tex;

DirectoryInfo dinfo;

FileInfo[] files;

do

{

Console.Clear();

string rl;

Console.WriteLine(" iput W, R ");

rl = Console.ReadLine();

if (rl == "W")

{

Console.Clear();

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

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

Console.WriteLine(" iput name of the folder where will be saved you file");

name = Console.ReadLine();

Console.WriteLine("do you wont to use new code? Y N");

string isnewcoding = Console.ReadLine();

switch(isnewcoding)

{

case"Y":

Tex = new List<string>();

Console.WriteLine("you have next file:");

dinfo = new DirectoryInfo("text");

files = dinfo.GetFiles();

foreach (FileInfo filenames in files)

{

Tex.Add(Convert.ToString(filenames));

}

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

{

Console.WriteLine(Tex[i]);

}

Console.WriteLine("enter name of the file for coding");

fileforcoding = Console.ReadLine();

code = changeFile.ReadFile("Code//Code", true, false);

text = changeFile.ReadFile("text//"+fileforcoding, false, false);

wWD = new WorkWhithFile.WorkWithData(text,code,name,true,null);

wWD.CodingData();

Console.WriteLine(" it`s done");

break;

case"N":

Console.WriteLine("you have next folder:");

DirectoryInfo dir = new DirectoryInfo("Resalt");

foreach (var item in dir.GetDirectories())

{

Console.WriteLine(item.Name);

}

Console.WriteLine(" ");

Console.WriteLine("enter name of the folder whith nessesary coding");

string newcoding = Console.ReadLine();

code = changeFile.ReadFile("Resalt//" + newcoding + "//Key", true, false);

Console.WriteLine(" ");

Console.WriteLine("you have next file:");

Tex = new List<string>();

dinfo = new DirectoryInfo("text");

files = dinfo.GetFiles();

foreach (FileInfo filenames in files)

{

Tex.Add(Convert.ToString(filenames));

}

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

{

Console.WriteLine(Tex[i]);

}

Console.WriteLine(" ");

Console.WriteLine("enter name of the file for coding");

fileforcoding = Console.ReadLine();

text = changeFile.ReadFile("text//"+fileforcoding, false, false);

wWD = new WorkWhithFile.WorkWithData(text, code, name,false,newcoding);

wWD.CodingData();

Console.WriteLine(" it`s done");

break;

default:

break;

}

}

else if (rl == "R")

{

Console.Clear();

Console.WriteLine("you have next folder:");

DirectoryInfo dir = new DirectoryInfo("Resalt");

foreach (var item in dir.GetDirectories())

{

Console.WriteLine(item.Name);

}

Console.WriteLine(" ");

Console.WriteLine(" iput name of the folder whith nessesary file");

name = Console.ReadLine();

Console.WriteLine(" ");

Console.WriteLine("you have next folder:");

DirectoryInfo di = new DirectoryInfo("Resalt");

foreach (var item in di.GetDirectories())

{

Console.WriteLine(item.Name);

}

Console.WriteLine(" ");

Console.WriteLine("enter name of the folder whith nessesary coding");

string newcoding = Console.ReadLine();

Console.WriteLine(" ");

wWD = new WorkWhithFile.WorkWithData(name,newcoding);

wWD.DecodingData();

Console.WriteLine(" it`s done");

}

string l;

Console.WriteLine("exit e");

l = Console.ReadLine();

if (l == "e")

exit = true;

}while(!exit);

}

}

}

ChangeFile.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

namespace PZ\_2.WorkWhithFile

{

class ChangeFile

{

public ChangeFile()

{

}

public IDictionary<string, string> ReadKey(string nameFiles)

{

IDictionary<string,string> rd=new SortedDictionary<string,string>();

string[] lines = File.ReadAllLines(nameFiles + ".txt",Encoding.UTF8);

bool key = true;

string data="";

string dates="";

foreach (string line in lines)

{

foreach (char c in line)

{

if (c != '-' && key)

{

data = Convert.ToString(c);

key = false;

}

else if (c != '-' && !key)

{

dates = Convert.ToString(c);

key = true;

rd.Add(data, dates);

}

}

}

return rd;

}

public List<string> ReadFile(string nameFiles, bool iscode, bool codeData)

{

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

string[] lines = File.ReadAllLines(nameFiles+ ".txt",Encoding.UTF8);

if(iscode)

{

foreach (string line in lines)

{

foreach (char c in line)

{

if(c!=' ')

{

string data = Convert.ToString(c);

DataLists.Add(data);

}

}

}

}

else if (!iscode && codeData)

{

foreach (string line in lines)

{

foreach (char c in line)

{

if (c != ' ' && c!=',')

{

string data = Convert.ToString(c);

DataLists.Add(data);

}

}

}

}

else if (!iscode && !codeData)

{

foreach (string line in lines)

{

foreach (char c in line)

{

string data = Convert.ToString(c);

DataLists.Add(data);

}

}

}

return DataLists;

}

public void Write(List<string> dataList, IDictionary<string, string> usingCode, string namefolder)

{

StreamWriter file;

bool exit = false;

do

{

CreateFolder(namefolder);

file = new StreamWriter("Resalt//" + namefolder + "//CodeData.txt");

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

{

string wr = dataList[i];

file.Write(wr);

}

file.Close();

file = new StreamWriter("Resalt//" + namefolder + "//Key.txt");

foreach (var item in usingCode)

{

string wr = item.Key + "-" + item.Value;

file.WriteLine(wr);

}

file.Close();

exit = true;

} while (!exit);

}

public void Write(List<string> dataList, string namefolder)

{

StreamWriter file;

file = new StreamWriter("Resalt//" + namefolder + "//DeCodedData.txt");

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

{

string wr = dataList[i];

file.Write(wr);

}

file.Close();

}

public void CreateFolder(string namefolder)

{

Directory.CreateDirectory("Resalt//" + namefolder);

}

}

}

WorkWhithData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PZ\_2.WorkWhithFile

{

class WorkWithData

{

IDictionary<string,string> usingCode;

ChangeFile changeFile;

List<string> code;

List<string> dataToCoding;

string namefolderdata;

string namefoldercode;

bool isnewcod;

List<string> toWrite;

public WorkWithData(List<string> dataToCoding, List<string> code, string namefolder, bool isnewcoding,string namefoldercode)

{

this.code=code;

this.dataToCoding=dataToCoding;

changeFile=new ChangeFile();

usingCode = new SortedDictionary<string, string>();

this.namefolderdata = namefolder;

this.isnewcod = isnewcoding;

this.namefoldercode = namefoldercode;

}

public WorkWithData(string namefolderdata,string namefoldercode)

{

this.namefolderdata = namefolderdata;

this.namefoldercode = namefoldercode;

changeFile = new ChangeFile();

}

private string CodingNewSimvol(string simvolToCoding, List<string> massCode)

{

Random rand = new Random();

bool next=false;

string cod;

int i = 0;

int j = 0;

do

{

i = 0;

j = 0;

cod = massCode[rand.Next(0,massCode.Count)];

foreach (var item in usingCode)

{

if (item.Value!= cod)

{

i++;

}

}

foreach (var item in usingCode)

{

if (item.Value != cod)

{

j++;

}

}

if(usingCode.Count==0 || i==usingCode.Count && j==usingCode.Count)

next = true;

}while(!next);

usingCode.Add(simvolToCoding,cod);

return cod;

}

private string CodingExistSimvol(string simvolToCoding)

{

string cod="";

foreach(var item in usingCode)

{

if (item.Key == simvolToCoding)

{

cod = item.Value;

break;

}

}

return cod;

}

public void CodingData()

{

if(!isnewcod)

usingCode = changeFile.ReadKey("Resalt//" + namefoldercode + "//Key");

toWrite = new List<string>();

foreach(var itemSimvol in dataToCoding)

{

bool isExist = false;

foreach(var itemCode in usingCode)

{

if (itemSimvol == itemCode.Key)

{

isExist = true;

break;

}

else

isExist = false;

}

if(isExist)

{

toWrite.Add(CodingExistSimvol(itemSimvol));

}

else

{

toWrite.Add(CodingNewSimvol(itemSimvol,code));

}

Random rand = new Random();

int between = rand.Next(0,10);

if(between<5)

{

toWrite.Add(" ");

}

else

{

toWrite.Add(",");

}

}

try

{

changeFile.Write(toWrite, usingCode, namefolderdata);

}

catch(Exception ex)

{

Console.WriteLine(ex);

}

}

public List<string> DecodingData()

{

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

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

usingCode = new SortedDictionary<string, string>();

decoded = changeFile.ReadFile("Resalt//" + namefolderdata + "//CodeData", false, true);

usingCode = changeFile.ReadKey("Resalt//" + namefoldercode + "//Key");

foreach(var item in decoded)

{

foreach (var itemCod in usingCode)

{

if(item==itemCod.Value)

{

decodedData.Add(itemCod.Key);

}

}

}

changeFile.Write(decodedData,namefolderdata);

return decodedData;

}

}

}