//Randall Cliett -------------Vigenere Cipher Implementation

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Text.RegularExpressions;

namespace VigenereCipherImplementation

{

class Program

{

static void Main(string[] args)

{

string finished = "n";

while (finished != "EXIT") //Loops main process until user is finished.

{

try

{

Implement();

}

catch //writes this if exception occurs--should only be a filenotfound

{

Console.WriteLine("Whoops, something went wrong. Are you sure the filepath names are correct?");

}

//Checks for end condition

Console.WriteLine("Type (EXIT) if you are finished. Otherwise this program will repeat.");

string temp = Console.ReadLine();

finished = temp.ToUpper();

}

//Warns that the program will close

Console.WriteLine("Press any key to close.");

Console.ReadKey();

}

static void Implement()

{

bool cont = false;

var upperprocess = "Q";

while (cont == false) //Loops to garuntee either encryption or decryption is declared

{

Console.WriteLine("Enter (encrypt) to encrypt a message or (decrypt) to decrypt.");

string process = Console.ReadLine();

upperprocess = process.ToUpper();

if (upperprocess == "ENCRYPT" || upperprocess == "DECRYPT")

cont = true;

}

//Gathers key and message location

Console.WriteLine("Please input the filename for the key.");

string keyloc = Console.ReadLine();

Console.WriteLine("Please input the filename for the message.");

string messageloc = Console.ReadLine();

//Retrieves key and message

string key = System.IO.File.ReadAllText(@keyloc);

string message = System.IO.File.ReadAllText(@messageloc);

//makes each uppercase only

string upperkey = key.ToUpper();

string uppermessage = message.ToUpper();

//passes key and message to be encrypted or decrypted, prints out message

//Console.WriteLine("Your output is:");

string result = cypher(upperprocess,upperkey,uppermessage);

//Console.WriteLine(result);

Console.WriteLine("Please input the filename you wish to write the result to.");

string outputloc = Console.ReadLine();

System.IO.File.WriteAllText(@outputloc,result);

}

static string cypher(string process, string key, string message)

{

List<string> cyphermessage = new List<string>();////////////////////////////////////////////////////////////////////////////////////////

//cleans up the key and message

string temp = message.Replace(" ", "").Replace("\n", "").Replace("\t","");

string cleanmessage = Regex.Replace(temp, @"[^A-Za-z]+", "");

temp = key.Replace(" ", "").Replace("\n", "").Replace("\t", "");

string cleankey = Regex.Replace(temp, @"[^A-Za-z]+", "");

//sets copies of key and message for use

string remainingkey = cleankey;

string remainingmessage = cleanmessage;

while (remainingmessage.Length != 0)//Loops while there is still a message to work on.

{

while (remainingkey.Length != 0)//loops while the key is being used, stops when the key needs to be reused.

{

if (remainingmessage.Length == 0)//breaks loop if message runs out before key does

break;

string currentkeychar = remainingkey.Substring(0, 1);//takes the first character of the key for use

remainingkey = remainingkey.Substring(1, remainingkey.Length - 1);//keeps track of which char is next for key use

//changes key char into int representation

char cha = currentkeychar[0];

int unicode = (int) cha;

unicode = unicode - 65;//gathers how much the message needs to be adjusted by the key

string currentmessagechar = remainingmessage.Substring(0, 1);//takes the first char of message for use

remainingmessage = remainingmessage.Substring(1,remainingmessage.Length - 1);//keeps track of which char for message is next

//changes message char into int representation

char messchar = currentmessagechar[0];

int messcode = (int) messchar;

if (process == "ENCRYPT")//Encrypting

{

messcode = messcode + unicode;//adjusts message by key value

if (messcode > 90)

{

messcode = messcode - 26;//keeps message in bounds for a-z

}

}

else//Otherwise Decrypting

{

messcode = messcode - unicode;

if (messcode < 65)

{

messcode = messcode + 26;

}

}

//changes message int into string representation

char cypherchar = (char) messcode;

string cyphermess = cypherchar.ToString();

//gathers all the strings of the message together

cyphermessage.Add(cyphermess);/////////////////////////////////////////////////////////////////////////////////////////////

}

//refreshes the key when it runs out

remainingkey = cleankey;

}

//Joins the message strings into one complete string and returns it

string completetext = string.Join("", cyphermessage.ToArray());/////////////////////////////////////////////////////////////////////

return completetext;

}

}

}