Doni Saputra Sirait

8020190320

Caesar Cipher

using MaterialSkin.Controls;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using MaterialSkin;

namespace CaesarCipher

{

public partial class CaesarCipher : MaterialForm

{

private string output;

public CaesarCipher()

{

InitializeComponent();

var materialSkinManager = MaterialSkinManager.Instance;

materialSkinManager.AddFormToManage(this);

materialSkinManager.Theme = MaterialSkinManager.Themes.LIGHT;

materialSkinManager.ColorScheme = new ColorScheme(Primary.Pink800, Primary.Pink900,

Primary.Pink500, Accent.LightBlue200, TextShade.WHITE);

ClearControlsEncrypt();

ClearControlsDecrypt();

}

public void ClearControlsEncrypt()

{

txtEncryptedText.Text = "";

txtKeyToEncrypt.Text = "";

txtToEncrypt.Text = "";

}

public void ClearControlsDecrypt()

{

txtDecryptedText.Text = "";

txtKeyToDecrypt.Text = "";

txtTextToDecrypt.Text = "";

}

private void btnClearControls\_Click(object sender, EventArgs e)

{

ClearControlsEncrypt();

}

private void btnClearControlsDecrypt\_Click(object sender, EventArgs e)

{

ClearControlsDecrypt();

}

private void btnCipher\_Click(object sender, EventArgs e)

{

if (!string.IsNullOrEmpty(txtKeyToEncrypt.Text) && !string.IsNullOrEmpty(txtToEncrypt.Text))

{

output = Cipher.Encrypt(txtToEncrypt.Text, int.Parse(txtKeyToEncrypt.Text));

txtEncryptedText.Text = output;

}

}

private void btnDecipher\_Click(object sender, EventArgs e)

{

if (!string.IsNullOrEmpty(txtKeyToDecrypt.Text) && !string.IsNullOrEmpty(txtTextToDecrypt.Text))

{

output = Cipher.Decrypt(txtTextToDecrypt.Text, int.Parse(txtKeyToDecrypt.Text));

txtDecryptedText.Text = output;

}

}

}

}

using System;

namespace CaesarCipher

{

public class Cipher

{

public static string Encrypt(string textToEncrypt, int key)

{

if (textToEncrypt.Length == 0) return String.Empty;

char chr = textToEncrypt[0].ToString().ToLower()[0];

var code = IsBasicLetter(chr) ? (char)(((chr - 'a' + key) % 26) + 'a') : chr;

return code + Encrypt(textToEncrypt.Substring(1), key);

}

public static string Decrypt(string textToDecrypt, int key)

{

if (textToDecrypt.Length == 0) return String.Empty;

char chr = textToDecrypt[0].ToString().ToLower()[0];

var code = IsBasicLetter(chr) ? (char)('z' - (('z' - chr + key) % 26)) : chr;

return code + Decrypt(textToDecrypt.Substring(1), key);

}

private static bool IsBasicLetter(char c)

{

return (c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z');

}

}

}

