Факультет компьютерных технологий

Лабораторная работа №10

«Разработка программ с использованием управляющих инструкций»

Вариант №10

Выполнил: студент гр. 981063 Ефименко Павел Викторович

Проверил: Яковлев А.В.

Минск 2021

Задание №1 Синхронные потоки

Условие:

Шифр «двойной квадрат» Уинстона

Код:

using System;

using System.Linq;

namespace Laba\_2

{

    class WheatstoneСipher

    {

        private const int M = 13;

        private const int N = 2;

        private char[] \_specSymbolArr = { ' ' };

        private static char[,] \_firstMatrix = new char[M, N];

        private static char[,] \_secondMatrix = new char[M, N];

        public string Encrypt(string text)

        {

            var enAlphabet = Enumerable.Range('a', 'z' - 'a' + 1).Select(i => (char)i).ToList();

            //var ruAlphabet = Enumerable.Range('А', 'я' - 'А').Select(i => (char)i).ToList();

            var alphabet = enAlphabet.ToArray();

            \_firstMatrix = alphabet.Shuffle().ConvertToMatrix(M, N);

            \_secondMatrix = alphabet.Shuffle().ConvertToMatrix(M, N);

            text = text.ToLower();

            string result\_text = "";

            if (text.Length % 2 != 0)

            {

                text += ' ';

            }

            int length = text.Length / 2;

            int k = 0;

            char[,] bigram = new char[length, 2];

            char[,] kripto\_bigram = new char[length, 2];

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

            {

                for (int j = 0; j < 2; j++)

                {

                    bigram[i, j] = text[k];

                    k++;

                }

            }

            int step = 0;

            while (step < length)

            {

                var cortege1 = FindIndexes(bigram[step, 0], \_firstMatrix);

                var cortege2 = FindIndexes(bigram[step, 1], \_secondMatrix);

                kripto\_bigram[step, 0] = \_secondMatrix[(int)cortege1.i, (int)cortege2.j];

                kripto\_bigram[step, 1] = \_firstMatrix[(int)cortege2.i, (int)cortege1.j];

                step++;

            }

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

            {

                for (int j = 0; j < 2; j++)

                {

                    result\_text += kripto\_bigram[i, j].ToString();

                }

            }

            return result\_text;

        }

        public string Decrypt(string text)

        {

            string result\_text = "";

            int length = text.Length / 2;

            int k = 0;

            char[,] bigram = new char[length, 2];

            char[,] kripto\_bigram = new char[length, 2];

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

            {

                for (int j = 0; j < 2; j++)

                {

                    bigram[i, j] = text[k];

                    k++;

                }

            }

            int step = 0;

            while (step < length)

            {

                var cortege1 = FindIndexes(bigram[step, 0], \_secondMatrix);

                var cortege2 = FindIndexes(bigram[step, 1], \_firstMatrix);

                kripto\_bigram[step, 0] = \_firstMatrix[(int)cortege1.i, (int)cortege2.j];

                kripto\_bigram[step, 1] = \_secondMatrix[(int)cortege2.i, (int)cortege1.j];

                step++;

            }

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

            {

                for (int j = 0; j < 2; j++)

                {

                    result\_text += kripto\_bigram[i, j].ToString();

                }

            }

            return result\_text;

        }

        private (int? i, int? j) FindIndexes(char symbol, char[,] matrix)

        {

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

            {

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

                {

                    if (symbol == matrix[i, j])

                    {

                        return (i, j);

                    }

                }

            }

            return (i: null, j: null);

        }

    }

}

using System;

using System.Diagnostics;

using System.Threading;

namespace Laba\_2

{

    class Program

    {

        static void Main(string[] args)

        {

            var wheatstoneСipher = new WheatstoneСipher();

            var thread = new Thread(() => { Cr(wheatstoneСipher, 0 ); });

            var thread2 = new Thread(() => { Cr(wheatstoneСipher, 1000); });

            thread.Start();

            thread2.Start();

            Console.WriteLine("[1]:decrypt or encrypt\n[2]:Your string");

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

            Console.ReadKey();

        }

        private static void Cr(WheatstoneСipher wheatstoneСipher, int sleepTime)

        {

            var userString = "hellol";

            var result = wheatstoneСipher.Encrypt(userString);

            Console.WriteLine(result);

            Thread.Sleep(sleepTime);

            result = wheatstoneСipher.Decrypt(result);

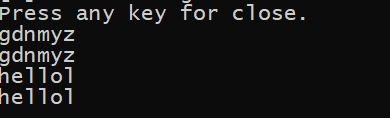
            Console.WriteLine(result);

        }

    }

}

Результат:



Вывод

В ходе лабораторной работы были отработаны навыки создания исключительных ситуаций и их обработка, также отработаны навыки многопоточного программирования с асинхронными потоками.