Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Информационные сети. Основы безопасности.

Отчет по лабораторной работе №1

Шифр Цезаря. Шифр Виженера.

Выполнил: студент гр. 953504 Басенко К. А.

Проверил: Олисейчик В.В.

Минск 2022

Содержание

Введение

Цель выполнения задания:

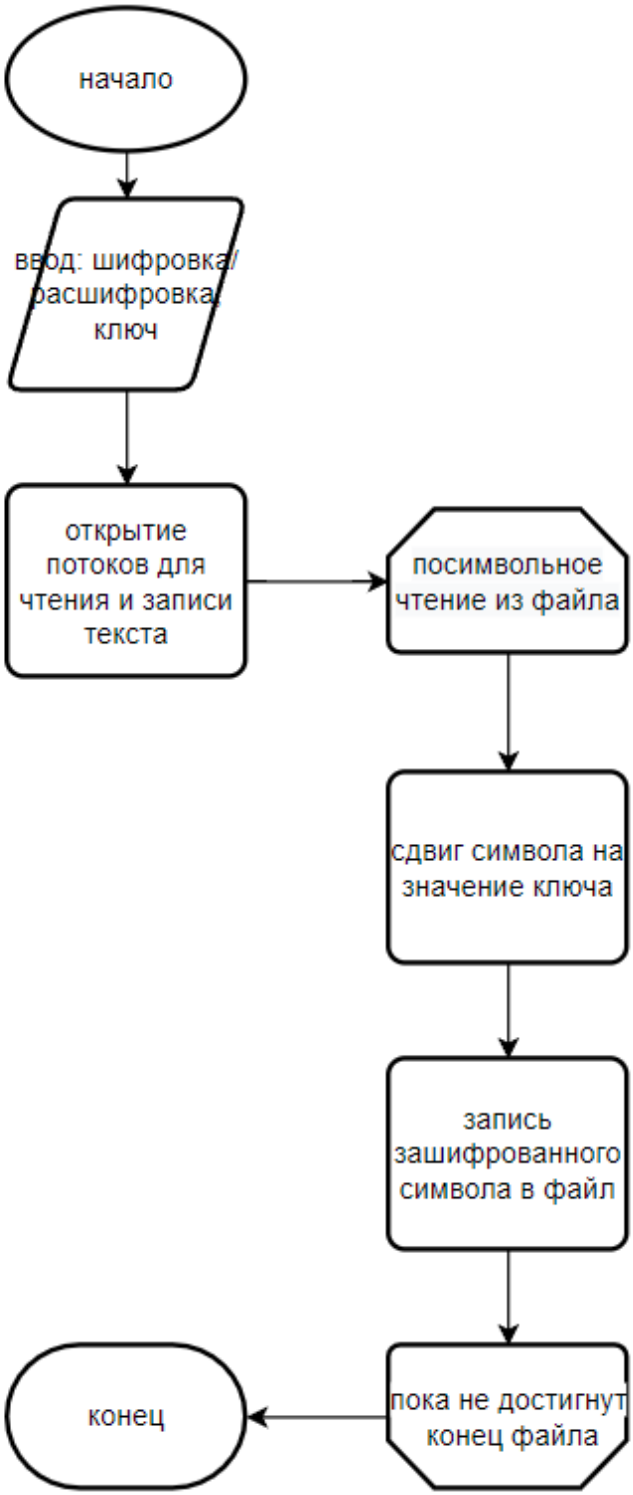
- Изучить шифры Цезаря и Виженера
- Реализовать программные средства шифрования и дешифрования текстовых файлов при помощи Шифра Цезаря и шифра Виженера.
-

Краткие теоретические сведения

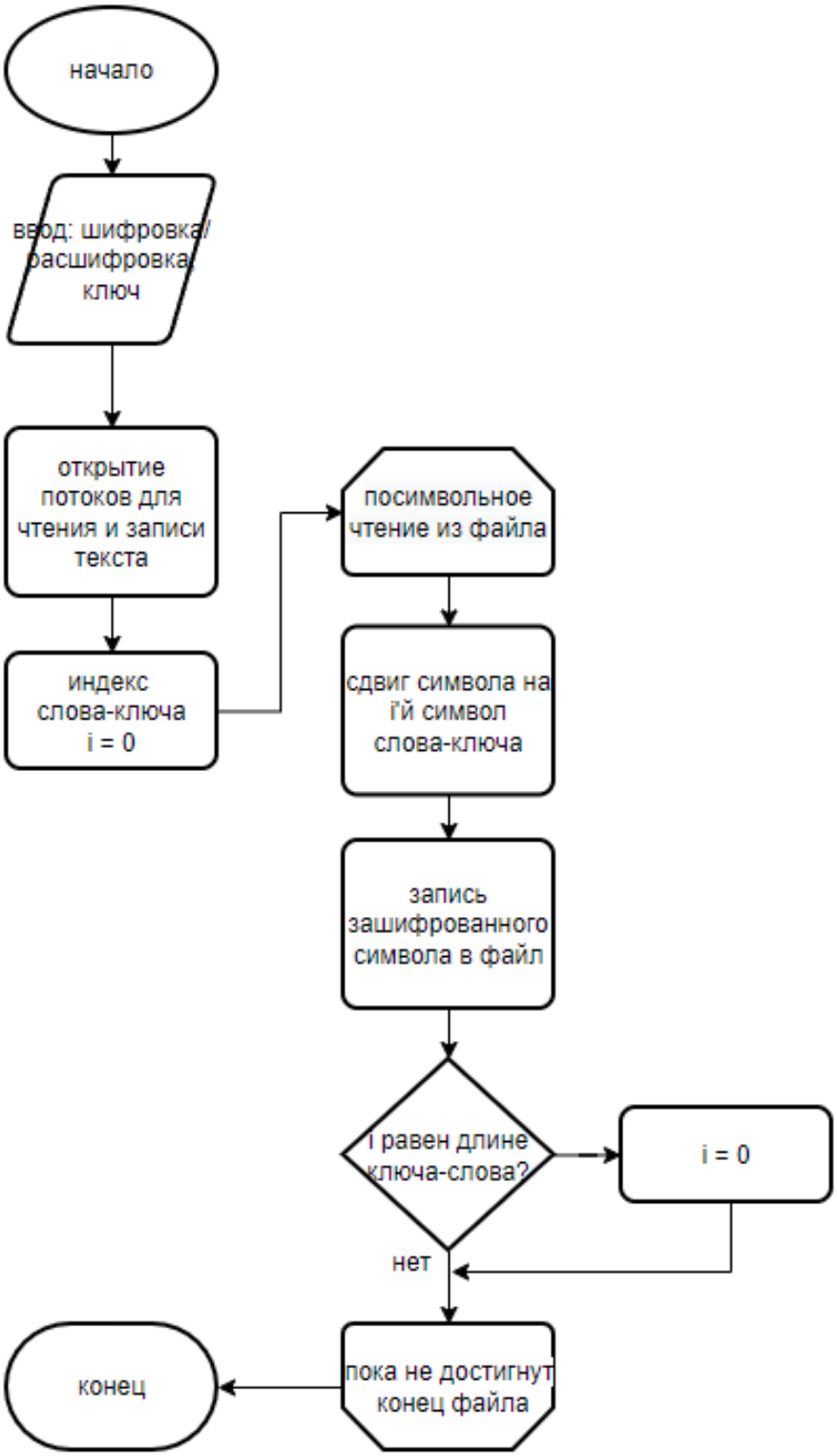
Шифр Цезаря, также известный, как шифр сдвига, код Цезаря или сдвиг Цезаря – один из самых простых и наиболее широко известных методов шифрования. Шифр Цезаря – это вид шифра подстановки, в котором каждый символ в открытом тексте заменяется символом, находящимся на некотором постоянном числе позиций левее или правее него в алфавите. Например, в шифре со сдвигом 4 А была бы заменена на Г, Б станет Д, и так далее.

Шифр Виженера состоит из последовательности нескольких шифров Цезаря с различными значениями сдвига. Для шифрования может использоваться таблица алфавитов, называемая tabula recta или квадрат (таблица) Виженера. Применительно к латинскому алфавиту таблица Виженера составляется из строк по 26 символов, причём каждая следующая строка сдвигается на несколько позиций. Таким образом, в таблице получается 26 различных шифров Цезаря. На каждом этапе шифрования используются различные алфавиты, выбираемые в зависимости от символа ключевого слова.

Блок-схема алгоритма. Шифр Цезаря



Шифр Веженера. Блок-схема.



Результат выполнения:

Шифр Цезаря

Шифровка

Текст в файле:

```
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam fermentum sem
tortor. Aenean malesuada suscipit odio in blandit. Nam facilisis, turpis at
lacinia tempor, felis diam dapibus enim, sit amet interdum mauris mi ut justo.
Quisque euismod erat at lectus egestas, mattis lobortis lacus fringilla. Orci
varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus
mus. Etiam pulvinar augue vitae massa tincidunt gravida. Suspendisse luctus
dolor urna, in sagittis justo porta quis. Suspendisse vitae molestie nisi. Duis
at quam sit amet nibh aliquam interdum. Aenean vel porttitor arcu, a facilisis
est.
```

Запуск программы:

>1.exe caesar encode 1

Текст в файле:

```
Mpsfn jqtvn epmps tju bnfu, dpotfdufuvs bejqjtdjoh fmju. Bmjrvbn gfsnfouvn tfn
upsups. Bfofbo nbmftvbeb tvtdjqju pejp jo cmboeju. Obn gbdjmjtjt, uvsqjt bu
mbdjojb ufnqps, gfmjt ejbn ebqjcvt fojn, tju bnfu joufsevn nbvsjt nj vu kvtup.
Rvjtrvf fvjtnpe fsbu bu mfduvt fhftubt, nbuujt mpcpsujt mbdvt gsjohjmmb. Psdj
wbsjvt obuprvf qfobujcvt fu nbhojt ejt qbsuvsjfou npouft, obtdfuvs sjejdvmvt
nvt. Fujbn qvmwjobs bvhvf wjubf nbttb ujodjevou hsbwjeb. Tvtqfoejttf mvduvt
epmps vsob, jo tbhjuujt kvtup qpsub rvjt. Tvtqfoejttf wjubf npmftujf ojtj. Evjt
bu rvbn tju bnfu ojci bmjrvbn joufsevn. Bfofbo wfm qpsuujups bsdv, b gbdjmjtjt
ftu.
```

Шифр Цезаря

Расшифровка

Текст в файле:

```
Mpsfn jqtvn epmps tju bnfu, dpotfdufuvs bejqjtdjoh fmju. Bmjrvbn gfsnfouvn tfn
upsups. Bfofbo nbmftvbeb tvtdjqju pejp jo cmboeju. Obn gbdjmjtjt, uvsqjt bu
mbdjojb ufnqps, gfmjt ejbn ebqjcvt fojn, tju bnfu joufsevn nbvsjt nj vu kvtup.
Rvjtrvf fvjtnpe fsbu bu mfduvt fhftubt, nbuujt mpcpsujt mbdvt gsjohjmmb. Psdj
wbsjvt obuprvf qfobujcvt fu nbhojt ejt qbsuvsjfou npouft, obtdfuvs sjejdvmvt
nvt. Fujbn qvmwjobs bvhvf wjubf nbttb ujodjevou hsbwjeb. Tvtqfoejttf mvduvt
epmps vsob, jo tbhjuujt kvtup qpsub rvjt. Tvtqfoejttf wjubf npmftujf ojtj. Evjt
bu rvbn tju bnfu ojci bmjrvbn joufsevn. Bfofbo wfm qpsuujups bsdv, b gbdjmjtjt
ftu.
```

Запуск программы:

>l1.exe caesar decode 1

Текст в файле:

```
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam fermentum sem
tortor. Aenean malesuada suscipit odio in blandit. Nam facilisis, turpis at
lacinia tempor, felis diam dapibus enim, sit amet interdum mauris mi ut justo.
Quisque euismod erat at lectus egestas, mattis lobortis lacus fringilla. Orci
varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus
mus. Etiam pulvinar augue vitae massa tincidunt gravida. Suspendisse luctus
dolor urna, in sagittis justo porta quis. Suspendisse vitae molestie nisi. Duis
at quam sit amet nibh aliquam interdum. Aenean vel porttitor arcu, a facilisis
est.
```

Шифр Веженера

Шифровка

Текст в файле:

```
Neque porro quisquam est qui dolorem ipsum quia dolor sit amet, consectetur,
adipisci velit...
There is no one who loves pain itself, who seeks after it and wants to have it,
simply because it is pain...
```

Запуск программы:

>l1.exe vigenere encode abiba

Текст в файле:

```
Nfyve qwsro yvisrcbm fau qvq domwsem qqsun ruib eolpz siu bmeu, cooafctfbvr,
ieipjadi wmmit...
Upfre qt np pne eio mwwes xbin qusemn, wiw sefst agbfr jb ane xanua to pbve qu,
tqnplz cecbcte jb is xbin...
```

Шифр Веженера

Расшифровка

Текст в файле:

```
Nfyve qwsro yvisrcbm fau qvq domwsem qqsun ruib eolpz siu bmeu, cooafctfbvr,
ieipjadi wmmit...
Upfre qt np pne eio mwwes xbin qusemn, wiw sefst agbfr jb ane xanua to pbve qu,
tqnplz cecbcte jb is xbin...
```

Запуск программы:

>l1.exe vigenere decode abiba

Текст в файле:

```
Neque porro quisquam est qui dolorem ipsum quia dolor sit amet, consectetur,
adipisci velit...
There is no one who loves pain itself, who seeks after it and wants to have it,
simply because it is pain...
```

Код программы:

```
using System.Text;

namespace Program
{
   public class Program
   {
      private static readonly string EncodedFilePath = "encoded.txt";
      private static readonly string DecodedFilePath = "decoded.txt";

      private static (string, Action<string[]>)[] commands = new (string, Action<string[]>)[]
      {
         ("help", PrintHelp),
         ("caesar", Caesar),
         ("vigenere", Vigenere),
      };

      private static string[][] helpMessages = new string[][]
      {
         new string[] { "help", "prints the help screen"},
         new string[] { "caesar", "encode/decode caesar cipher with key in range[0, 26]" },
         new string[] { "vigenere", "encode/decode vigenere cipher with key" },
      };

      private static string[] CodingModes = new string[]
      {
         "encode",
         "decode",
      };

      public static void Main(string[] args)
```

```
    {
        if (args.Length != 3 || args[0] == commands[0].Item1)
        {
            PrintHelp(args);
            return;
        }

        const int commandIndex = 0;
        var command = args[commandIndex];

        var index = Array.FindIndex(commands, i => i.Item1.Equals(command,
StringComparison.InvariantCultureIgnoreCase));

        if (index >= 0)
        {
            const int parametersIndex = 1;
            commands[index].Item2(args[parametersIndex..]);
        }
        else
        {
            PrintHelp(args);
            return;
        }
    }

    private static void PrintHelp(string[] args)
    {
        Console.WriteLine("commands:");
        foreach (var helpMessage in Program.helpMessages)
        {
            Console.WriteLine("\t{0,-10}\t- {1}.", helpMessage[0], helpMessage[1]);
        }
    }
```

```
private static void Caesar(string[] parameters)
{
    if (!byte.TryParse(parameters[^1], out var key) || key > 26 || key < 0)
    {
        Console.WriteLine("invalid key.");
    }

    var mode = parameters[0];

    if (!Program.CodingModes.Contains(mode))
    {
        PrintHelp(parameters);
        return;
    }

    string fileToReadPath = Program.DecodedFilePath,
           fileToWritePath = Program.EncodedFilePath;

    if (mode.Equals("decode", StringComparison.InvariantCultureIgnoreCase))
    {
        (fileToReadPath, fileToWritePath) = (fileToWritePath, fileToReadPath);
    }

    using var readingStream = new StreamReader(fileToReadPath, Encoding.UTF8);
    using var writingStram = new StreamWriter(fileToWritePath, false, Encoding.UTF8);

    int readed;
    char symbol;
    var sb = new StringBuilder();
    while ((readed = readingStream.Read()) > 0)
    {
        symbol = Convert.ToChar(readed);
```

```
            symbol = ShiftAlphabetSymbol(symbol, key, mode);
            writingStram.Write(symbol);
        }

        writingStram.Flush();
    }

    private static void Vigenere(string[] parameters)
    {
        var key = parameters[^1];
        var mode = parameters[0];

        if (!Program.CodingModes.Contains(mode))
        {
            PrintHelp(parameters);
            return;
        }

        string fileToReadPath = Program.DecodedFilePath,
               fileToWritePath = Program.EncodedFilePath;

        if (mode.Equals("decode", StringComparison.InvariantCultureIgnoreCase))
        {
            (fileToReadPath, fileToWritePath) = (fileToWritePath, fileToReadPath);
        }

        using var readingStream = new StreamReader(fileToReadPath, Encoding.UTF8);
        using var writingStram = new StreamWriter(fileToWritePath, false, Encoding.UTF8);

        int readed;
        char symbol;
        var sb = new StringBuilder();
        int count = 0;
```

```
    while ((readed = readingStream.Read()) > 0)
    {
        symbol = Convert.ToChar(readed);
        symbol = ShiftAlphabetSymbol(symbol, IndexInAlphabet(key[count]), mode);
        writingStram.Write(symbol);
        count = (count + 1) % key.Length;
    }

    writingStram.Flush();
}

private static byte IndexInAlphabet(char symbol)
{
    symbol = char.ToLower(symbol);
    return (byte)(symbol - 'a');
}

private static char ShiftAlphabetSymbol(char symbol, byte key, string codingMode)
{
    var ranges = new int[][]
    {
        new int[] { 97, 122 },
        new int[] { 65, 90 },
    };

    int code = (int)symbol;

    foreach (var range in ranges)
    {

        if (code >= range[0] && code <= range[1])
        {
            code -= range[0];
```

```
            code += codingMode switch
            {
               "encode" => key,
               "decode" => -key,
               _ => 0,
            };

            var len = range[1] - range[0] + 1;
            if (code < 0)
            {
               code += len;
            }
            code %= len;
            code += range[0];

            break;
         }
      }

      return (char)code;
   }
  }
}
```