Министерство образования Республики Беларусь

Учреждение образования

“Белорусский государственный университет информатики и радиоэлектроники”

Кафедра инженерной психологии и эргономики

Практическая работа №1

«Криптоанализ классических шифров»

Проверил: Выполнила:

Давыдович К.И. ст. гр.910903 Неверова А.П.

Минск 2022

**Цель работы:** выполнить задания №1,2.

**Задание №1:**

Ниже два шифртекста одного и того же сообщения, зашифрованные с

помощью классических шифров:

**a.** Цезарь - Шифртекст 1.

Srobdoskdehwlf vxevwlwxwlrq flskhuv

**b.** простой замены - Шифртекст 2.

KjgyVgkcVWZqdX nsWnqdqsqdji XdkcZmn

Напишите программу дешифрования, используя любой известный вам

язык программирования:

1) найдите соответствующий открытый текст, вскрыв шифр Цезаря,

2) а затем найдите ключ шифра простой замены, используя для

дешифрования известный открытый текст.

**Задание №2:**

Напишите программу, используя любой известный вам язык

программирования:

* зашифруйте свою фамилию, имя отчество,
* дешифруйте полученный текст,
* сравните с исходным текстом.

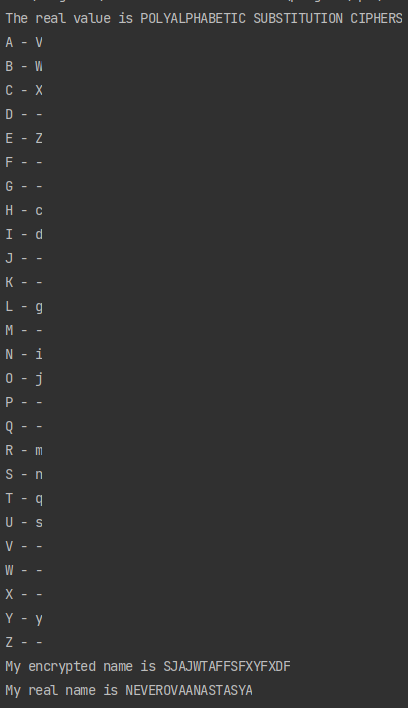
**Код программы:**

namespace CaesarsSipher;  
  
public static class Caesars  
{  
 private const string **Alphabet** = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";  
 private const string **CaesarsSipherText** = "Srobdoskdehwlf vxevwlwxwlrq flskhuv";  
 private const string **SipherText** = "KjgyVgkcVWZqdX nsWnqdqsqdji XdkcZmn";  
 private const string **MyName** = "NeverovaAnastasya";  
  
 private static string[] MostCommonBigrams { get; } =  
 {  
 "th", "he", "in", "en", "nt", "re", "er", "an", "ti", "es", "on", "at", "se", "nd", "or", "ar", "al", "te",  
 "co", "de", "to", "ra", "et", "ed", "it", "sa", "em", "ro"  
 };  
  
 private static string[] MostCommonTrigrams { get; } =  
 {  
 "the", "and", "tha", "ent", "ing", "ion", "tio", "for", "nde", "has", "nce", "edt", "tis", "oft", "sth", "men"  
 };  
  
 private static Dictionary<int, string> DecryptionProcess(string value)  
 {  
 var indexes = value.ToUpper().Select(u => **Alphabet**.IndexOf(u)).ToArray();  
 var results = new Dictionary<int, string>();  
  
 for (var step = 0; step < **Alphabet**.Length; step++)  
 {  
 var values = indexes.Select(index => index == -1 ? ' ' : **Alphabet**[(index + step) % **Alphabet**.Length])  
 .ToList();  
  
 results.Add(**Alphabet**.Length - step, $"{string.Join(String.Empty, values)}");  
 }  
  
 return results;  
 }  
  
 private static string GeTheMostLikelyValue(Dictionary<int, string> results)  
 {  
 var real = new Dictionary<int, int>();  
  
 foreach (var @case in results)  
 {  
 int count = 0;  
 foreach (var bigram in MostCommonBigrams)  
 {  
 if (@case.Value.IndexOf(bigram.ToUpper(), StringComparison.**Ordinal**) > 0)  
 {  
 count++;  
 }  
 }  
  
 foreach (var trigram in MostCommonTrigrams)  
 {  
 if (@case.Value.IndexOf(trigram.ToUpper(), StringComparison.**Ordinal**) > 0)  
 {  
 count++;  
 }  
 }  
  
 real.Add(@case.Key, count);  
 }  
  
 var indexOfRealValue = real.Aggregate((l, r) => l.Value > r.Value ? l : r).Key;  
 return results[indexOfRealValue];  
 }  
  
 private static Dictionary<char, char> GetKey(string realString, string value)  
 {  
 var dictionary = new Dictionary<char, char>();  
  
 foreach (var t in **Alphabet**)  
 {  
 var index = realString.IndexOf(t);  
 dictionary.Add(t, index > 0 ? value[index] : '-');  
 }  
  
 return dictionary;  
 }  
  
 private static void ShowDictionary(Dictionary<char, char> dictionary)  
 {  
 foreach (var couple in dictionary)  
 {  
 Console.WriteLine($"{couple.Key} - {couple.Value}");  
 }  
 }  
  
 private static string HideMyName(string value, int step)  
 {  
 var results = new List<char>();  
 foreach (var c in value.ToUpper())  
 {  
 var index = **Alphabet**.IndexOf(c);  
 results.Add(**Alphabet**[(index + step) % **Alphabet**.Length]);  
 }  
  
 return string.Join(String.Empty, results);  
 }  
  
 public static void ShowWork()  
 {  
 *// 1st task* var allPossibleResults = DecryptionProcess(**CaesarsSipherText**);  
 var mostLikelyRealValue = GeTheMostLikelyValue(allPossibleResults);  
 Console.WriteLine($"The real value is {mostLikelyRealValue}");  
 *//2nd task* var key = GetKey(mostLikelyRealValue, **SipherText**);  
 ShowDictionary(key);  
 *//3rd task* var hideName = HideMyName(**MyName**, 5);  
 Console.WriteLine($"My encrypted name is {hideName}");  
 var allPossibleNames = DecryptionProcess(hideName);  
 var mostLikelyRealName = GeTheMostLikelyValue(allPossibleNames);  
 Console.WriteLine($"My real name is {mostLikelyRealName}");  
 }  
}

**Результаты работы программы:**

Расшифрованный текст: POLYALPHABETIC SUBSTITUTION CIPHERS. Он был зашифрован с шагом 3 по шифру Цезаря.

Фамилия и имя были зашифрованы по шифру Цезаря с шагом 5. Результат шифрования SJAJWTAFFSFXYFXDF. Результат дешифрования NEVEROVAANASTASYA. Исходный и полученный текста совпадают.

****

**Вывод:** задания 1-2 были выполнены в полной мере. Написан код для дешифровки шифра Цезаря с использованием наиболее часто используемых биграмм и триграмм. Ключ шифра простой замены найден, нераспознанные символы заменяются символом «-». Фамилия и имя были зашифрованы с помощью шифра Цезаря с шагом 5.