Министерство образования Республики Беларусь

Учреждение образования

“Белорусский государственный университет информатики и радиоэлектроники”

Кафедра инженерной психологии и эргономики

Практическая работа №4

«Создание ключей в системе PGP, передача подписанных и защищенных сообщений»

Проверил: Выполнила:

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

Минск 2022

**Цель работы:** изучить метод шифрования и выполнить задания для усвоения полученного материала.

**Задание:**

1. Изучите теоретическую часть.
2. Напишите программу на любом языке программирования:
3. Переведите число 3 43 в двоичную систему счисления.
4. Пусть каждая из 16 первых букв русского алфавита (абвгдежзийклмноп…) имеет четырехразрядный двоичный код, соответствующий ее номеру от 0 до 15, т.е. а ‑00002, б ‑00012, …, п ‑11112. а) Составьте из этих букв произвольное сообщение состоящее из 32 символов, б) затем разбейте полученное сообщение на блоки длиной 64 бита. в) Значения полученных блоков запишите в десятичной системе счисления.
5. Найдите состояние 28‑разрядного двоичного регистра сдвига после циклического сдвига влево на 5, числа X, предварительно записанного в регистр.

**Код программы:** namespace CaesarsSipher;  
  
public static class Pgp  
{  
 private const string **Symbols** = "aбвгдежзийклмноп";  
 private const int **X** = 179357333; *// 11 variant* private const long **Value** = 22448993011;  
  
 private static string ToBinary(long value)  
 {  
 return Convert.ToString(value, 2);  
 }  
  
 private static Dictionary<char, string> GetBinaryCodes(string symbols)  
 {  
 var result = new Dictionary<char, string>();  
 for (var index = 0; index < symbols.Length; index++)  
 {  
 var binary = ToBinary(index);  
 result.Add(symbols[index],  
 binary.Length switch  
 {  
 1 => $"000{binary}",  
 2 => $"00{binary}",  
 3 => $"0{binary}",  
 \_ => binary  
 });  
 }  
  
 return result;  
 }  
  
 private static string GenerateMessage(Dictionary<char, string> dictionary)  
 {  
 var result = string.Empty;  
 var random = new Random();  
  
 for (var i = 0; i < 32; i++)  
 {  
 int index = random.Next(**Symbols**.Length);  
 dictionary.TryGetValue(**Symbols**[index], out var binaryResult);  
 result += binaryResult;  
 }  
  
 return result;  
 }  
  
 private static ulong ToDecimal(string value)  
 {  
 return (ulong)Convert.ToInt64(value, 2);  
 }  
  
 private static string GetValueAfterPledge(string valueIn28Bits)  
 {  
 return string.Concat(valueIn28Bits.AsSpan(5, valueIn28Bits.Length - 5), valueIn28Bits[..5]);  
 }  
  
 private static long GetSumByMod(long value1, int value2)  
 {  
 return (value1 + value2) % 2;  
 }  
 public static void ShowWork()  
 {  
 Console.WriteLine($"3^43 is {ToBinary(BitConverter.DoubleToInt64Bits(Math.Pow(3, 43)))} in binary");  
 var result = GenerateMessage(GetBinaryCodes(**Symbols**));  
 Console.WriteLine(result);  
 Console.WriteLine(  
 $"Binary value: {result[..64]} - Decimal value: {ToDecimal(result[..64])}");  
 Console.WriteLine(  
 $"Binary value: {result.Substring(64, 64)} - Decimal value: {ToDecimal(result.Substring(64, 64))}");  
 Console.WriteLine($"{ToBinary(Convert.ToInt32(**X**))} => {GetValueAfterPledge(ToBinary(Convert.ToInt32(**X**)))}");  
 Console.WriteLine($"{**X**} => {ToDecimal(GetValueAfterPledge(ToBinary(Convert.ToInt32(**X**))))}");  
 Console.WriteLine($"({**Value**} + {**X**}) mod 2 = {GetSumByMod(**Value**, **X**)}");  
 }  
}

**Вывод:** написала программа, которая решает все поставленные в практической работе задачи.