нечном итоге видеопроектор позволяет многократно увеличивать объем нужных видеоматериалов, с которыми надо знакомить студентов во время лекций, что дает возможность значительно сократить количество учебных часов без снижения качества занятий. Это позволяет уменьшить расход бумаги и электроэнергии.

В современных условиях студентам значительная часть учебных материалов передается в электронном виде, что также дает большую экономию бумаги. Большую роль в учебном процессе играет использование текстового редактора Word. Его применение избавило пользователей применять для печати изжившую себя пишущую машинку. Использование этого редактора в составе компьютера и с применением принтера дает огромное количество возможностей, которых нет при применении пишущей машинки. Появление табличного процессора Excel в программном обеспечении компьютеров позволило значительно упростить проведение инженерных и экономических расчетов по сравнению с применением языков программирования. В настоящее время студентам и выпускникам ВУЗов в подавляющем числе случаев нет необходимости при проведении расчетов применять языки программирования.

УДК 621.762.4

Францкевич А.А.

КРИПТОГРАФИЯ. НЕКОТОРЫЕ АЛГОРИТМЫ ШИФРОВАНИЯ, ИХ РЕАЛИЗАЦИЯ В VISUAL C#

БГПУ имени М.Танка, Минск, Республика Беларусь
Научный руководитель: ст. преподаватель Нарейко Н.Н.

Компьютерные технологии дали человечеству уникальные возможности по хранению информации и передачи ее из одной точки пространства в другую. При этом возникла проблема обеспечения секретности хранимых и передаваемых

данных. Решить эту проблему позволяет такая современная информационная технология как криптография. Она базируется на шифровании текстовых данных. Существует много различных алгоритмов шифрования. Нами рассмотрены такие алгоритмы как ГОСТ-2814789, Виженера, RSA. Реализация алгоритмов продемонстрирована в разработанном автором на языке Visual C# Windows-приложении.

С давних пор и до сегодняшних дней актуальной остается защита информации. Шифрование является одним из способов защиты данных и предназначено для решения трех основных задач:

- конфиденциальность: защита данных пользователя или его идентификации от несанкционированного чтения;
- целостность: защита данных от изменений;
- аутентификация: гарантия того, что данные поступили от указанного в сообщении отправителя.

Применяемые схемы шифрования принято классифицировать следующим образом:

- симметричное шифрование с закрытым ключом (Например, алгоритм шифрования ГОСТ-2814789);
- ассиметричное шифрование с открытым ключом;
- цифровая подпись;
- хеширование.

Дадим теперь более точные определения основных понятий, используемых в криптографии. Пусть X и Y – это два множества, элементы которых будем называть *данными*. Под шифром будем понимать алгоритм или отображение:

$$y = F(k, x); \quad y \in Y; \quad x \in X; \quad k \in K;$$

Процесс получения элемента y по заданному элементу x называют *шифрованием.* Элементы x – это *исходные данные*, y – *зашифрованные данные*. При шифровании используется ключ k – элемент некоторого множества K, называемого *множеством ключей.* Всегда подразумевается возможность

дешифрования – существование обратного отображения, позволяющего восстановить исходный элемент:

$$x = G(k, y) = G(k, F(k, x)); \quad y \in Y; \quad x \in X; \quad k \in K;$$

Рассмотрим несколько примеров простых шифров.

Пример 1 (алгоритм сложения). Пусть *X, Y, K* – множества целых чисел, а алгоритм шифрования задается операцией сложения: $y = x+k$. Понятно, что существует обратное отображение: $x = y – k$.

Пример 2 (алгоритм сложения по модулю). Пусть *X, Y, K* – множества целых чисел в диапазоне [*0, p-1*], а алгоритм шифрования задается операцией сложения по модулю *p*: $y = (x+k) \pmod p$. Понятно, что существует обратное отображение: $x = (y – k) \pmod p$.

Шифрование применяется, прежде всего, к текстовым данным. В памяти компьютера тексты, как и любая другая информация, хранится в виде последовательности битов. При шифровании эта последовательность битов нарезается на блоки, как правило фиксированной длины, и каждый блок шифруется. Чаще всего, при шифровании учитывается контекст и шифруется смесь блока с его соседями, например с предшествующим блоком. В этом случае задача раскрытия шифра значительно усложняется.

Рассмотрим основные шаги криптопреобразования алгоритма шифрования ГОСТ 28147-89 на языке программирования C# (рисунок 1).

Шаг 0. Определяет исходные данные для основного шага криптопреобразования:

N – преобразуемый 64-битовый блок данных;

X – 32-битовый элемент ключа;

private byte[,] matrixH = new byte[8, 16]

Шаг 1. Сложение с ключом. Младшая половина преобразуемого блока складывается по модулю 2^{32} с используемым на шаге элементом ключа, результат передается на следующий шаг;

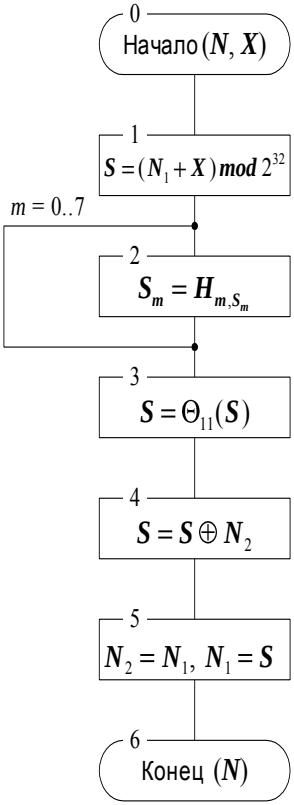

Рисунок 1 – Схема основного шага криптопреобразования алгоритма ГОСТ 28147-89

```
uint sm = (uint)((N1 + iP) / (2 ^ 32));
```

Шаг 2. Поблочная замена. 32-битовое значение, полученное на предыдущем шаге, интерпретируется как массив из восьми 4-битовых блоков кода:
$\boldsymbol{S}=(\boldsymbol{S}_0,\boldsymbol{S}_1,\boldsymbol{S}_2,\boldsymbol{S}_3,\boldsymbol{S}_4,\boldsymbol{S}_5,\boldsymbol{S}_6,\boldsymbol{S}_7)$.

```
BitArray S = new BitArray(BitConverter.GetBytes(sm));
            byte[] bufer = new byte[8];
            for (int m = 0; m <= 7; m++)
            {   int smi = GetByteFromFourBit(S, m);
                byte b = matrixH[m, smi];
                bufer[m] = b;
            }
            S = GetArray4FromByte(bufer);
            byte[] b1 = new byte[4];
            S.CopyTo(b1, 0);
            uint S1 = BitConverter.ToUInt32(b1, 0);
```

Шаг 3. Циклический сдвиг на 11 бит влево. Результат предыдущего шага сдвигается циклически на 11 бит в сторону старших разрядов и передается на следующий шаг. На схеме алгоритма символом $\square_{11}$ обозначена функция циклического сдвига своего аргумента на 11 бит в сторону старших разрядов.

Шаг 4.

Шаг 5.

S1 = S1 << 11;

Шаг 6. Побитовое сложение: значение, полученное на шаге 3, побитно складывается по модулю 2 со старшей половиной преобразуемого блока.

S1 = S1 ^ N2;

Шаг 7. Сдвиг по цепочке: младшая часть преобразуемого блока сдвигается на место старшей, а на ее место помещается результат выполнения предыдущего шага.

```
N2 = N1;
N1 = S1;
List<byte> result = new List<byte>();
result.AddRange(BitConverter.GetBytes(N1));
result.AddRange(BitConverter.GetBytes(N2));
return result.ToArray();
```

Шаг 8. Полученное значение преобразуемого блока возвращается как результат выполнения алгоритма основного шага криптопреобразования.

```
        // цикл шифрования
for (int partmes = 0; partmes < Message.Count / 64; partmes++)
 { BitArray blockMes = GetPartFromBitArray(Message, partmes * 64, 64);
  cryptmes.AddRange(CryptStep(blockMes)); }
return cryptmes.ToArray();
        // цикл расшифрования
for (int partmes = 0; partmes < CryptMessage.Count / 64; partmes++)
{BitArray blockMes = GetPartFromBitArray(CryptMessage, partmes * 64, 64);
decryptmes.AddRange(DecryptStep(blockMes));}
return decryptmes.ToArray();
```