Национальный исследовательский университет «МЭИ» Институт автоматики и вычислительной техники

Кафедра вычислительных машин, систем и сетей

Лабораторная работа № 9

«Электронная цифровая подпись RSA»

по курсу «Защита информации»

Выполнил: Кутузов И.Г.

Группа: А-08-16

Подпись: 

Преподаватель: Рытов А.А.

Москва, 2020 г.

In[ ]:=

In[ ]:=
```
ngr = 8;
nstd = 10;
npage = ngr * 25 + nstd
```

Out[ ]=
```
210
```

In[ ]:=
```
(*1. Создание дайджеста сообщения.*)
```

In[ ]:=
```
(*1.1.Создать строку текста (text0)→
  выбрать 2 абзаца на странице из книги Романец и др.Номер страницы-
 Ngp*25+Nсп,где Ngp-номер группы,Nсп-номер по списку в группе.*)
text0 = "Контейнер -это специальный файл, создаваемый при помощи Панели Управления
    системы Crypton Sigma. Контейнер можно открыть для доступа через
    логический диск, обслуживаемый драйвером системы Crypton Sigma. Все
    файлы, находящиеся на этом логическом диске, хранятся в зашифрованном
    виде. Пользователь может создать любое количество контейнеров.Каждый
    контейнер имеет свой собственный пароль. Пользователь должен ввести этот
    пароль при создании контейнера и использовать его для получения доступа
    к тем данным, которые будут храниться в данном контейнере. Используя
    Панель Управления Crypton Sigma, пользователь может сменить пароль для
    выбранного контейнера при условии, что ему известен прежний пароль.
";
(*1.2.Ввести следующие изменения в text0 и создать модифицированные строки:text0-
 убрать точку в text0;
text2-добавить пробел в text0;
text3-поменять местами две расположенные рядом (разные)
буквы в text0.*)
SeedRandom[nstd];
инициализация генератора псевдослучайных чисел
text1 = StringReplace[text0, "." → "", 1]
        заменить в строке
position = RandomInteger[StringLength[text0]];
           случайное цело··· длина строки
text2 = StringJoin[StringTake[text0, position],
        соединить с··· взять часть строки
  " ", StringTake[text0, {position + 1, StringLength[text0]}]]
       взять часть строки                 длина строки

text3 = Characters[text0];
        символы
positionletter1 = RandomInteger[StringLength[text0] - 1];
                  случайное цело··· длина строки
While[text3[[positionletter1]] == text3[[positionletter1 + 1]],
цикл-пока
 positionletter1 = RandomInteger[StringLength[text0] - 1]]
                   случайное цело··· длина строки
{text3[[positionletter1]], text3[[positionletter1 + 1]]} =
  {text3[[positionletter1 + 1]], text3[[positionletter1]]};

text3 = StringJoin[text3]
        соединить строки
text0 == text0
text0 == text1
text0 == text2
text0 == text3
```

Out[ ]= Контейнер –это специальный файл, создаваемый при помощи Панели Управления системы Crypton Sigma Контейнер можно открыть для доступа через логический диск, обслуживаемый драйвером системы Crypton Sigma. Все файлы, находящиеся на этом логическом диске, хранятся в зашифрованном виде. Пользователь может создать любое количество контейнеров.Каждый контейнер имеет свой собственный пароль. Пользователь должен ввести этот пароль при создании контейнера и использовать его для получения доступа к тем данным, которые будут храниться в данном контейнере. Используя Панель Управления Crypton Sigma, пользователь может сменить пароль для выбранного контейнера при условии, что ему известен прежний пароль.

Out[ ]= Контейнер –это специальный файл, создаваемый при помощи Панели Управления системы Crypton Sigma. Контейнер можно открыть для доступа через логический диск, обслуживаемый драйвером системы Crypton Sigma. Все файлы, находящиеся на этом логическом диске, хранятся в зашифрованном виде. Пользователь может создать любое количество контейнеров.Каждый контейнер имеет свой собственный пароль. Пользователь должен ввести этот пароль при создании контейнера и использовать его для получения доступа к тем данным, которые будут храниться в данном контейнере. Используя Панель Управления Crypton Sigma, пользователь может сменить пароль для выбранного контейнера при условии, что ему известен прежний пароль.

Out[ ]= Контейнер –это специальный файл, создаваемый при помощи Панели Управления системы Crypton Sigma. Контейнер можно открыть для доступа через логический диск, обслуживаемый драйвером системы Crypton Sigma. Все файлы, находящиеся на этом логическом диске, хранятся в зашифрованном виде. Пользователь может создать любое количество контейнеров.Каждый контейнер имеет свой собственный пароль. Пользователь должен ввести этот пароль при создании контейнера и использовать его для полученияд оступа к тем данным, которые будут храниться в данном контейнере. Используя Панель Управления Crypton Sigma, пользователь может сменить пароль для выбранного контейнера при условии, что ему известен прежний пароль.

Out[ ]= True

Out[ ]= False

Out[ ]= False

Out[ ]= False

In[ ]:=
```
(*1.3.Найти расстояние Дамерау-Левенштейна (ДЛ)-
 минимальное количество операций вставки одного символа,
удаления одного символа и замены одного символа на другой,
необходимых для превращения одной строки в другую-(DamerauLevenshteinDistance[,])
                                                    расстояние Дамерау–Левенштейна
  поочередно между строкой text0 и строками text1,text2,text3.*)
DamerauLevenshteinDistance[text0, text1]
расстояние Дамерау–Левенштейна
DamerauLevenshteinDistance[text0, text2]
расстояние Дамерау–Левенштейна
DamerauLevenshteinDistance[text0, text3]
расстояние Дамерау–Левенштейна
```

Out[ ]= 1

Out[ ]= 1

Out[ ]= 1

In[ ]:=
```
(*1.4.Найти расстояние Дамерау-
 Левенштейна (DamerauLevenshteinDistance[,]) поочередно между значениями хэш-
              расстояние Дамерау–Левенштейна
 функций (ToString[Hash[]]) строки text0 и значениями хэш-
          преобраз··· хэш
 функций строк text1,text2,text3.*)
ToString[Hash[text0]]
          хэш
ToString[Hash[text1]]
          хэш
ToString[Hash[text2]]
          хэш
ToString[Hash[text3]]
          хэш
DamerauLevenshteinDistance[ToString[Hash[text0]], ToString[Hash[text1]]]
рас···                     преобраз··· хэш          преобра··· хэш
DamerauLevenshteinDistance[ToString[Hash[text0]], ToString[Hash[text2]]]
                           преобра··· хэш          преобраз··· хэш
DamerauLevenshteinDistance[ToString[Hash[text0]], ToString[Hash[text3]]]
                           преобраз··· хэш          преобраз··· хэш
```

Out[ ]= 3872271878246152678

Out[ ]= 5306445914224393148

Out[ ]= 8677542619139990387

Out[ ]= 9223049405699960675

Out[ ]= 16

Out[ ]= 17

Out[ ]= 16

In[ ]:=
```
(*Определить расстояние ДЛ между значениями хэш-
 функций строк text0 и text1 для алгоритмов хэширования,приведенных в таблице.*)
{{"HASH", "DamerauLevenshteinDistance"},
                расстояние Дамерау–Левенштейна
   {"исходный текст без хеширования", DamerauLevenshteinDistance[text0, text1]},
                                       расстояние Дамерау–Левенштейна
   {"CRC32", DamerauLevenshteinDistance[ToString[Hash[text0, "CRC32"]],
                расстояние Дамерау–Левенштейна  преобра···  хэш
     ToString[Hash[text1, "CRC32"]]]}, {"MD5", DamerauLevenshteinDistance[
                 хэш                                 расстояние Дамерау–Левенштейна
     ToString[Hash[text0, "MD5"]], ToString[Hash[text1, "MD5"]]]},
    преобраз···  хэш                  преобра···  хэш
   {"SHA", DamerauLevenshteinDistance[ToString[Hash[text0, "SHA"]],
                расстояние Дамерау–Левенштейна  преобраз···  хэш
     ToString[Hash[text1, "SHA"]]]}, {"SHA256", DamerauLevenshteinDistance[
    преобраз···  хэш                                  расстояние Дамерау–Левенштейна
     ToString[Hash[text0, "SHA256"]], ToString[Hash[text1, "SHA256"]]]},
    преобра···  хэш                  преобраз···  хэш
   {"SHA384", DamerauLevenshteinDistance[ToString[Hash[text0, "SHA384"]],
                расстояние Дамерау–Левенштейна  преобра···  хэш
     ToString[Hash[text1, "SHA384"]]]},
                 хэш
   {"SHA512", DamerauLevenshteinDistance[ToString[Hash[text0, "SHA512"]],
                расстояние Дамерау–Левенштейна  преобра···  хэш
     ToString[Hash[text1, "SHA512"]]]}} // Grid // Framed
                 хэш                                 таблица  обрамлённый
```

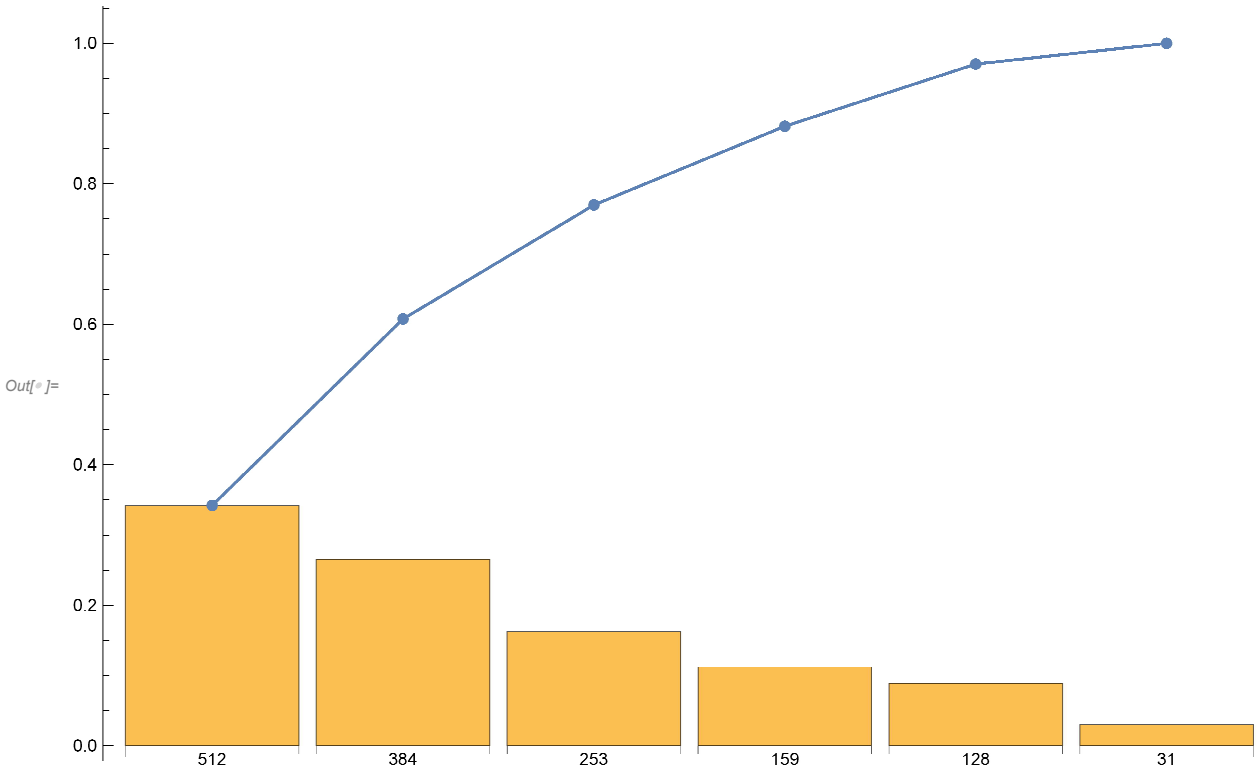
Out[ ]=

HASH	DamerauLevenshteinDistance
исходный текст без хеширования	1
CRC32	10
MD5	30
SHA	38
SHA256	55
SHA384	90
SHA512	116

In[ ]:=
```
(*Построить диаграмму Парето,отражающую зависимость
  расстояния ДЛ от числа бит результирующего значения хэш-функции.*)
```

In[ ]:=
```
Needs["StatisticalPlots`"];
ParetoPlot[{{BitLength[Hash[text0, "CRC32"]], DamerauLevenshteinDistance[
    ToString[Hash[text0, "CRC32"]], ToString[Hash[text1, "CRC32"]]]},
  {BitLength[Hash[text0, "MD5"]], DamerauLevenshteinDistance[
    ToString[Hash[text0, "MD5"]], ToString[Hash[text1, "MD5"]]]},
  {BitLength[Hash[text0, "SHA"]], DamerauLevenshteinDistance[
    ToString[Hash[text0, "SHA"]], ToString[Hash[text1, "SHA"]]]},
  {BitLength[Hash[text0, "SHA256"]], DamerauLevenshteinDistance[
    ToString[Hash[text0, "SHA256"]], ToString[Hash[text1, "SHA256"]]]},
  {BitLength[Hash[text0, "SHA384"]], DamerauLevenshteinDistance[
    ToString[Hash[text0, "SHA384"]], ToString[Hash[text1, "SHA384"]]]},
  {BitLength[Hash[text0, "SHA512"]], DamerauLevenshteinDistance[
    ToString[Hash[text0, "SHA512"]], ToString[Hash[text1, "SHA512"]]]}}]
```

Out[ ]=



In[ ]:=

In[ ]:=

```
In[ ]:= (*2.Формирование элементов системы RSA.*)
        (*2.1.По номеру в списке группы Nсп определить два случайных номера n1 и n2,
        лежащих в интервале[10000+1000*(Nсп-1),10000+1000*Nсп].Используемая
         функция RandomInteger[{,}]*)
                  случайное целое число
        n1 = RandomInteger[{10 000 + 1000 * (nstd - 1), 10 000 + 1000 * nstd}]
             случайное целое число
        n2 = RandomInteger[{10 000 + 1000 * (nstd - 1), 10 000 + 1000 * nstd}]
             случайное целое число
```

Out[ ]= 19 613

Out[ ]= 19 642

```
In[ ]:= (*2.2.Найти два простых числа pA и qA,
        а также модуль nA=pA*qA.Определить функцию Эйлера phiA=EulerPhi[] для nA.*)
                                                                  функция Эйлера
        pA = RandomPrime[n1]
             случайное простое число
        qA = RandomPrime[n2]
             случайное простое число
        nA = pA * qA
        phiA = EulerPhi[nA]
               функция Эйлера
```

Out[ ]= 12 497

Out[ ]= 113

Out[ ]= 1 412 161

Out[ ]= 1 399 552

```
In[ ]:= (*2.3.Выбрать открытый ключ:ko-случайное число,
         которое меньше phiA и в тоже время НОД(ko,phiA)=1 (GCD[,]==1).*)
                                                              НОД
        ko = RandomPrime[phiA - 1]
             случайное простое число
        While[GCD[ko, phiA] != 1, ko = RandomPrime[phiA - 1]]
              НОД                      случайное простое число
        GCD[ko, phiA] == 1
        НОД
        phiA
        ko
```

Out[ ]= 500 389

Out[ ]= True

Out[ ]= 1 399 552

Out[ ]= 500 389

```
In[ ]:= (*2.4. Определить секретный ключ ks.Проверить условие НОД(ks,nA)=1*)
```

```
In[ ]:= ks = PowerMod[ko, -1, phiA]
        степень по модулю
        GCD[ks, nA] == 1
        НОД
Out[ ]= 197 933
Out[ ]= True

In[ ]:= (*2.5.Преобразовать свою фамилию в числовой код m (а→1,..я→32),
        получить криптограмму c,зашифровав m на открытом ключе ko.*)
        m = ToCharacterCode[ToUpperCase["аКутузовя"], "WindowsCyrillic"] -
            код символа          перевести в верхний регистр
          ToCharacterCode["A", "WindowsCyrillic"][[1]] + 1
          код символа
        m = Take[m, {2, Length[m] - 1}]
            извлечь      длина
        c = PowerMod[m, ko, nA]
            степень по модулю

Out[ ]= {1, 11, 20, 19, 20, 8, 15, 3, 32}
Out[ ]= {11, 20, 19, 20, 8, 15, 3}
Out[ ]= {1 023 091, 686 627, 554 545, 686 627, 1 237 471, 656 432, 1 089 591}

In[ ]:= (*2.6.Расшифровать криптограмму c на ключе ks и получить m.*)
        mm = PowerMod[c, ks, nA]
             степень по модулю
        m == mm
Out[ ]= {11, 20, 19, 20, 8, 15, 3}
Out[ ]= True

In[ ]:= (*.Преобразовать строку хеш-кода сообщения m в последовательность (список) чисел,
        определить длину этого списка и подготовить два новых списка такой же
          величины для шифр текста и восстановленного (расшифрованного)хеш-кода.*)
        hsh = IntegerDigits[Hash[m]]
              цифры целого ч···  хэш
        hshlen = Length[hsh]
                 длина
Out[ ]= {4, 4, 6, 5, 2, 9, 9, 4, 7, 9, 5, 5, 1, 7, 4, 7, 8, 3, 0}
Out[ ]= 19

In[ ]:= (*2.8.Провести операцию шифрования хеш-
         кода на ключе ks и зафиксировать результат.*)
        hshcript = PowerMod[hsh, ks, nA]
                   степень по модулю
Out[ ]= {171 764, 171 764, 118 108, 317 379, 119 782, 410 633, 410 633, 171 764,
         25 997, 410 633, 317 379, 317 379, 1, 25 997, 171 764, 25 997, 461 839, 812 764, 0}
```

In[ ]:=
```
(*2.9.Провести операцию расшифрования хеш-
 кода на ключе ko и зафиксировать результат.*)
hshdecript = PowerMod[hshcript, ko, nA]
```
степень по модулю

Out[ ]= {4, 4, 6, 5, 2, 9, 9, 4, 7, 9, 5, 5, 1, 7, 4, 7, 8, 3, 0}

In[ ]:=
```
(*2.10.Сравнить результат,полученный в п.2.8.с исходным хэш-кодом.*)
hsh == hshdecript
```

Out[ ]= True