HADMÉRNÖK

X. Évfolyam 3. szám - 2015. szeptember

HORVÁTH Zoltán – PÁNDI Erik
horvath.zoltan@uni-nke.hu - pandi.erik@uni-nke.hu

PRACTICAL IMPLEMENTATION OF ASYMMETRIC ENCRYPTION, DESKTOP COMPUTERS EQUIPPED CLASSROOMS

Absztrakt

The authors presents an own developed software. The program has an arithmetic skills, which makes it suitable for classroom computers on a practical way to deepen the theoretical knowledge acquired.

A szerzők bemutatnak egy saját fejlesztésű szoftvert. A program olyan aritmetikai képességekkel rendelkezik, mely alkalmassá teszi a tantermi számítógépeket a megszerzett elméleti ismeretek gyakorlati úton történő elmélyítésére.

Kulcsszavak: *training development, cryptographic algorithm, RSA encryption ~ képzésfejlesztés, rejtjelező algoritmus, RSA kódolás*

INTRODUCTION

The signal officer training includes theoretical awareness and funds related to the practical implementation of encryption. The examples stays in theoretical planes, because lack of the practical experience. This is due to limited arithmetic ability of the classroom computers. The program to be presented will allow basic mathematical operations, asymmetric cryptography by carry out with huge natural numbers, prim check and search for congruent pairs on this numbers.

1. The problem of arithmetic operations with large natural numbers

If the processing a natural number of hundred digits are required, the conventional means is not feasible. It is important the accurate representation of the results. The basic operations can be carried out, it should be ensured. For further demand the exponentiation, prime check, search congruent number pairs. This software solves these problems.

1.1. Arithmetic operations with text

If on the numbers do not perform the basic operations, but also held a series of digits as text characters, the extended range of number representation. The length of the numbers will no longer be restricted. Therefore, be drawn to the basic operations are carried out procedures over text. This means the implementation of the traditional paper-based counting.

1.1.1. Addition, multiplication

Basic operation performed on the one digit, addition and multiplication.

For example: **’8’ + ’6’ = 14**; where the carry is **’1’**, the result is **’4’**;
’8’ ∗ ’6’ = 48; where the carry is **’4’**, the result is **’8’**.

During the operations, if a number stored as digit performed, an addition or multiplication it must take into account the value the previous carry during operations. In this case we get the result and the value of the next carry.

For example: If **C = ’1’**; where **’C’** is the previous carry, then
’8’ + ’6’ + ’1’ = 15; where the new carry is **’1’**, the result is **’5’**;
’8’ ∗ ’6’ + ’1’ = 49; where the new carry is **’4’**, the result is **’9’**.

1.1.2. Make a complement for substraction and division

The basic operations include subtraction and division too. The extraction can be achieved in the following manner. It must be extracted to form the complement of ten. The amount of decimal numbers and tent’s complement of this number is zero, and the carry is **'1'**. In this case, if a tent’s complement of the decimal number by adding a number, subtraction is realized.

Creating the nine’s complement is achieved by looking to that number. The amount a number and this number’s comlement of nine is **'999…..'**. It's easy to do, because digit by digit we have performing the **'b' = 9 - 'a'.** To do this we obtain the complement of nine.

For example: **a := ’3675’**, the nine’s complement of **’a’** is **’6324’**.

We can made from a nine's comlement a ten's complement adding to it that even one. A number added to another number ten's complement subtraction is realized. One digit at a complement of nine preset, adding an initial value of the carry '1' to set, in addition to the subtraction takes place.

For example: **a := ’3675’**, the ten’s complement of **’a’** is **’6325’**.

$$\begin{array}{r} 8000 \\ \underline{-\ 3675} \\ 4325 \end{array} \qquad\qquad \begin{array}{r} 8000 \\ \underline{+\ 6325} \\ 14325 \end{array}$$

The substraction will give a good results if the end of the carry operation is '**1**' and in this case we have to leave this digit.

The division carried out a series of substraction. Then can be determined the result of the integer division **(a div b)**, and the remainder **(a mod b)**.

The exponential operations carried out a series of multiplication.

2. Other features of the RSA key generation and encryption needed

There is a need to provide additional opportunities produce of RSA keys. In case the number should be checked, that it is a prime numbers. Should be checked for a given two numbers that they are prime numbers on each-other. To find a congruent number for any number's on a basis number if there is.

During the prime check the program checks whether there is a number. which examined the number is divisible without remainder.

During the inspection of relatively prime, the program checks whether there is a common core of factors.

Running a bivariate parametric equations you need to solve in search of a few congruent numbers. The equation is as follows:

$$\mathbf{e * d = c * f + 1}$$

where:	**e** and **d**	the congruent numbers;
	c	a running parameters;
	f	based on the congruence.

Look out for the '**e**' number assigned '**d**' number. If '**e**' and '**f** ' relatively prime, then there exists a '**c**', which is the case '**c**' ∗ '**f** ' divided even with '**e**' or '**d**' the remainder is **’1’**. The program is known as '**e**' and '**f** ' in case calculate '**d**'. If '**e**' and '**f** ' relatively prime, the equation has a solution.

3. Other services that make the program more user-friendly.

Advantageous if a calculator able to handle the ASCII code table directly. If the memory of the calculator provides interoperability between the operands, the partial results and the ASCII codes, it makes the application more user-friendly.

4. An example of the practical use

The following practical exercise shows how to generate an RSA key, and how can a character encoding of RSA carried out.

4.1. Generate an RSA key

The RSA encryption key generation need two primes. The program can be verified that the selected numbers actually prime. For example: let '**p**'**: = 37** and '**q**'**: = 47**, the two primes.

The two primes multiplication will be the second segment of the RSA key pair, which is in this case, '**n**' **= 1739**.

The program provides incremental and decrement functions. Reducing one of **p** and **q** values, and these are multiplied by **'f' = 1656** was obtained. This will be a search for pairs of numbers congruent based on.

Let's find a number which number and **’f ’** is relatively prime. The program provides a relatively prime to be checked. In this case, **'e': = 119**.

Then find the **'e'** congruent numbers match **'f '** away. In this case, **'d' = 167** arises. The program provides a search function for congruent numbers.

Thus, the generated key pair: **'e'** and **'n'**, that **’K_1’ = 119 1739** and
'd' and' **n** ', that **’K_2’ = 167 1739**.

4.2. Presentation of RSA encryption

As an example of coded **'R'** character whose ASCII code is **82**, the key is **’K_1’** = 119 1739.

$\mathbf{82^{119}}$ = 554432185467607097354240087807286407466831135156589286803650522040664691422534845106733761420641234832687474104136345777145439768082364148623991444395487858861744062094132337697803148549625156729172449383034620876026179609427968

This is **228** characters.

The value of the earned Code: $\mathbf{82^{119}}$ **mod 1739** = **273**.

The decryption is the same procedure with **'K_2'** key is happening. where the Code is **273**, and **'K_2'** = 167 1739.

$\mathbf{273^{167}}$ = 69049741300825420041547351986796483665408721812044453495370403174494054554845039134175534755328478776796662512644296240834117863412459060948111206796413099311339390825750998896425108527408847941651165509457962470826292892726070642064222963288779905794905450651016490598073543981721143911742952321080436270980508225456019818689841033795430654615969433753689828357771025003694289232559580256995547666113078897

This is **407** characters.

The value of **’273^{167} mod 1739’** is = **82**. So we got back in coded **'R'** character’s ASCII code.

It is clear that the performance of these operations excluding special number representation and realization of advanced features time-consuming, if not impossible. Using the program is repeated opportunities to execute its tasks. This will ensure the deepening of theoretical knowledge, gaining experience through practice.

SUMMARY

Based on arithmetic capabilities of the program is suitable for RSA key generation, RSA coding. Through the practical task of implementing the key generation, encoding and decoding processes are no longer given knowledge, but experience. The experience gained in the stand-alone task, resulting in faster and deeper knowledge can be acquired.

References

[1] Dr. Berta István Zsolt: Nagy e-szignó könyv. Budapest, 2011 Microsec Kft, ISBN: 978-963-08-1168-2

[2] Ködmön József: Kriptográfia. Budapest, 1999, ComputerBooks, ISBN: 963-618-224-8

[3] Gonda János: A rejtjelezés néhány kérdése. Budapest, 2010 (http://www.inf.elte.hu/karunkrol/digitkonyv/Jegyzetek2010/A_rejtjelezes_nehany_kerdese.pdf) (2015. augusztus 27.)