Encryption based on Ballot, Stack permutations and Balanced Parentheses using Catalan-keys

Muzafer Saračević, Edin Korićanin, Enver Bišević
muzafers@gmail.com, edinkoricanin@gmail.com, e.bisevac@uninp.edu.rs
Department of Computer Sciences, University of Novi Pazar, Srbija

Contribution to the state of the art
DOI: **10.7251/JIT1702069S** UDC: **004.056.55:003.26**

Abstract: This paper examines the possibilities of applying Catalan numbers in cryptography. It also offers the application of appropriate combinatorial problems (Ballot Problem, Stack permutations and Balanced Parentheses) in encryption and decryption of files and plaintext. The paper analyzes the properties of Catalan numbers and their relation to these combined problems. Applied copyright method is related to the decomposition of Catalan numbers in the process of efficient keys generating. Java software solution which enables key generating with the properties of the Catalan numbers is presented at the end of the paper. Java application allows encryption and decryption of plaintext based on the generated key and combinatorial problems.

Keywords: Cryptography, Catalan numbers, Ballot notation, Stack permutations, Balanced Parentheses.

Introduction

Catalan numbers have found widespread use in solving many combinatorial problems. In references [7,15] are given concrete applications of these numbers with possible solutions when it comes to representation of certain combinatorial problems. All the aforementioned combinatorial problems can be solved on the basis of values which have the Catalan numbers properties. In short, the number of combinations and the manner of Catalan numbers generating represent a solution for certain combinatorial problems. Some can be enumerated: a binary tree, the triangulation of polygons, steak permutations, paired brackets problem, Ballot problem, the problem of motion through an integer grid, etc. All the aforementioned combinatorial problems can be solved on the basis of values which have the properties of the Catalan numbers, namely the solution of these combinatorial problems lies in the application of these numbers.

Combinatorial schemes based on permutations are used in the most symmetric cryptographic methods. The important place in these procedures, particularly in keys generating, have the pseudo-random numbers. In this paper we will try to determine the important place the number theory has in cryptology, primarily in the development of algorithms for generating pseudo-random numbers that are necessary for keys generating. Number theory with asymmetric systems has the main place not only in generating keys, but also in the design of the cryptographic algorithm itself, and in the cryptographic analysis [5,8].

In this paper, our aim was to give our proposal of Catalan numbers application in keys generating, or more precisely, that these numbers can be used as generators of pseudo-random numbers. By using Catalan numbers the basic idea is realized, and this is generated from a long and unpredictable sequence of symbols from an alphabet (e.g. Binary) on

the basis of short key selected in a random manner (basis n).

Encryption can be realized in a combination with various combinatorial problems that are based and whose solution is hiding within these numbers. In this way, a successful system performance for encryption has been achieved. The subject of the research refers to the testing of Catalan numbers and the possibility of their use in cryptography. Beside that, the subject of this research is also the application of appropriate combinatorial problems, which are based on the properties of the Catalan numbers, for encryption and decryption of files and plaintext. The idea arose based on our previous research in the field of number theory, combinatorial problems and Catalan numbers.

RELATED WORKS

On the basis of many scientific and expert papers which were published in reputable international journals, we have come to the knowledge that many combinatorial problems are used both in cryptography and as well as in the process of cryptanalysis. Among the most commonly used combinatorial problems are the problems of balanced or paired parentheses, application of stack permutations, voting problem (Ballot) etc. In paper [6] are given concrete applications of combinatorial problems in cryptography and cryptanalysis. Applied number theory has many applications in cryptography, especially the integer sequences. Previous cryptographic algorithms were designed by using the integer sequences of Fibonacci sequence and Lucas numbers.

In paper [1] were proposed cryptographic algorithms based on integer sequences of Catalan numbers, where new methods of encryption were proposed. In the proposed method of encryption, by using the Catalan numbers, a large random number "n" is set as a secret key of the communicating parties. Binary values in Catalan number C_n corresponding to the agreed secret key (i.e. the basis n) are used for encryption/decryption of the message.

In paper [16] is presented an advanced technique for encryption based on Catalan numbers. This technique combines the characteristics of substitution and transposition. In this paper, the displacement technique of each bit to the left is proposed and the displaced data is supplemented so that every generating bit of ciphertext is changed. Catalan numbers have the property of recursiveness and their generating can be efficiently implemented with the dynamical programming. Dynamical programming is a method that reduces the runtime of those problems that require finding an optimal substructure and having subproblems that are repetitive.

In paper [3], the fact that combinatorial Ballot problem or voting problem is widely used in cryptography is stated. This paper describes the theory and implementation of the electronic voting system with emphasis on the practical scheme of voting based on the so-called Gomomorf encryption.

In paper [10], a new voting method based on paper with interesting security properties is presented. The authors have attempted to examine whether the same safety features of proposed cryptographic protocols of voting can be achieved, but without the use of any cryptography.

Properties and space of *Catalan-keys*

Catalan numbers represent a sequence of numbers which were primarily used in geometry and in solving many combinatorial problems. Catalan numbers C_n, $n > 0$, present a series of natural numbers, which appear as a solution to a large number of known combinatorial problems (number of possible entries in form of n-balanced parenthesis, stack permutations, Ballot problem, binary trees, triangulation of polygons, etc.).

Property of Catalan-keys: Number can be labeled as Catalan number when its binary form consists of numbers equal to "1" and "0" and start with "1"**.** If binary notation of Catalan number is conected with another mode of writing, most often with the mode of balanced parentheses, then "1" represents an open parenthesis and "0" represents closed parenthesis, and it can be said that each opened parenthesis closes, or every bit 1 has its couple and that is bit 0. Also, the binary record of the Catalan number can be presented in the form of a stack permutation or *Ballot* record. Representation using Stack permutation treats bit 1 as a *PUSH* command and bit 0 as a POP command. More in the following sections.

Space of Catalan-keys: Catalan numbers are defined as [7]:

$$C_n = \frac{(2n)!}{(n+1)!n!} \quad (1)$$

The given formula is also an expression for the calculation of space of Catalan-keys.

Table 1. shows the Catalan base numbers n ∈ {1,2,…,30}, which are calculated by the formula (1).

Table 1. *List of Catalan numbers*

n	C_n	n	C_n	n	C_n
1	1	11	58,786	21	24,466,267,020
2	2	12	208,012	22	91,482,563,640
3	5	13	742,900	23	343,059,613,650
4	14	14	2,674,440	24	1,289,904,147,324
5	42	15	9,694,845	25	4,861,946,401,452
6	132	16	35,357,670	26	18,367,353,072,152
7	429	17	129,644,790	27	69,533,550,916,004
8	1,430	18	477,638,700	28	263,747,951,750,360
9	4,862	19	1,767,263,190	29	1,002,242,216,651,368
10	16,796	20	6,564,120,420	30	3,814,986,502,092,304

In this paper, we will use Catalan numbers as keys generator for encryption and decryption. From Table 1., it can be seen that *n* is a basis for keys generating and C_n determines the number of valid keys, i.e. those values that satisfy the characterstic of the Catalan number (space of keys). For example, for basis *n=30* we have space of keys C_{30}*=3,814,986,502,092,304*, i.e. the values that satisfy the property of the Catalan number. Increasing *n* basis, the space of keys is also drastically increasing. Table 1. shows the values for the first 30 Catalan numbers. In order to provide stronger, i.e. more resistant mechanism of cryptanalysis encryption, it is neccesary to choose keys whose value base is mainly greater than 30. Now we are going to analyze the values which are generated in the C_n set. For the purposes of Catalan number validity verification we will use the binary notation. The basic feature that must be fullfiled is bit property balance in binary form for a certain number from the C_n set (we will refere to this property as *bit-balance* property).

Cryptanalysis of Catalan-keys: From the last notation it follows that, if *n* is a basis for keys generating, then C_n is the total number of different binary formats which meet the Catalan numbers property. For example, for the basis *n=30,* C_{30} *= 3.814.986.502.092.304* is calculated according to the formula for calculating the Catalan number. So, for 60-bit keys there are *3.814.986.502.092.304* valid values that satisfy the condition of balance, i.e. Catalan number property. In an attempt to find all of these Catalan numbers and to perform their writing on a disk, the necessary space for this is *30.519.892.016.738.432* bytes or *28.423.864* GB or *27.757* TB. It means, that this process is very demanding when it comes to memory resources. On the other hand, if we want to find out all the 60-bit Catalan numbers and if for access to every element from the C_n set is needed 1 ms, the execution time would take *120972* years. The average time will be *120972 / 2 = 60486* years. It means, this process is very demanding when it comes to time as well.

Example 1: For n=3, based on the formula (1), we have a set of C_3=5 values that appease Catalan number property. These are the numbers *C[3]={42, 44, 50, 52, 56},* or, based on their binary format *101010, 101100, 110010, 110100, 111000*. We determine their property which corresponds to the Catalan's number, which is the bit- balance property. Observing the binary records of the Catalan numbers, we can see that the number of 1's and 0's is equal, which means that the bit balanace of 0's and 1's is the main feature of each Catalan number. Beside that, we can conclude that *n* basis shows the number of pairs of "1" and "0". In this way we can know the length of the key on the basis of a given base. In this example, the basis is 3, which means that the length of key will be 6 bits, i.e. it will have three pairs of "1" and "0", that satisfy the balancing property.

Example 2: Now we will analyze one record that does not match the balance property. Examples of numbers from the C_3 set which have balance property are 42, 44**.** Now we will check what is the situation with the number between them, and that is 43. Its binary record is *101011*. We can immediately notice that there is balance property violation and this happens in the sixth bit. The same case is with number 45. Its binary record is *101101*. We can notice that here balance property violation is in the fourth bit. These numbers cannot be used for encryption of the text on basis of a combinatorial problem which is based on the Catalan number property. More about this in the following sections.

Example 3: We will analyze Catalan number $(1142920503054772802)_{10}$ *= (11111101110001111 000001001000101111100010111001010100100000*

$10)_2$. From this record we can conclude that the condition of the bit-balance is met, which means that the value 11429205030547728O2 has the property of the Catalan number. The next step is to determine the basis for generating this value. The lenght of the binary record of this value is 60 bits, which means that there are 30 pairs of "1" and "0" in this record. Based on this, we can determine the basis, which is n=30.

The record of the Catalan number, in addition to the binary form, can be presented in other manners, that is, it can be represented through many combinatorial problems. It can be represented in the form of paired brackets "( ) ( ) (( ))" or in the form of *Ballot* record "ABABAABB" or via Stack permutations. In our authors' papers [9,11,12,13,14] we have performed generating testing of all numbers for a given basis *n*, which fullfill the above mentioned Catalan numbers properties.

The application of Ballot, Stack permutations and Balanced Parentheses

In this section, we will consider the application of some other combinatorial problems within the encryption process. We will implement a key that has Catalan number property on the problems such as *Ballot* (voting problem), *stack permutation* and *Balanced Parentheses*.

The general case of the *Ballot* problem is [17]: " *How many combinations there are to put the 2n votes in such way that in each adding a new vote, the number of votes that has been won by candidate A is greater than or equal to the number of the votes that candidate B has received".* This general case is related to the rule *a=b=n*, where *a* is a number of votes for candidate *A*, and *b* a number of votes for *B*, and where for each iteration applies $a_i \geq b_i$.

Since each combination begins with the first voice for candidate *A* (because for each iteration applies the rule: $a_i \geq b_i$, and so, for the first), then all the combinations of records that start with *B* are listed. In the last iteration the condition $a_i = b_i$ must be satisfied.

Example 4: The order of voting can be displayed in a tabular or matrix form for all the combinations. For example, for the order of voting *AABABB*, the following table is formed:

Candidate	i_1	i_2	i_3	i_4	i_5	i_6
	A	A	B	A	B	B
A	1	2	2	3	3	3
B	0	0	1	1	2	3

Based on the voting order table for both candidates, matrices can be formed for candidates A and B, where the matrix elements are recorded as ordinal numbers of votes at the time of adding each vote. For case *a=b=3*, where applies that $a_i \geq b_i$ for C_3 there are 5 such combinations.

$$A=\begin{bmatrix}1&2&3&3&3&3\\1&2&2&3&3&3\\1&1&2&3&3&3\\1&1&2&2&3&3\\1&2&2&2&3&3\end{bmatrix} \qquad B=\begin{bmatrix}0&0&0&1&2&3\\0&0&1&1&2&3\\0&1&1&1&2&3\\0&1&1&2&2&3\\0&0&1&2&2&3\end{bmatrix}$$

Example 5: In the matrices A and B, the example of 6 voters is given, of which 3 are voting for candidate A and 3 for candidate B. These matrices can be represented in the form of a matrix with Ballot records. If we respect the condition that candidate A got the first vote, and the balance of the appearance of another candidate is respected, where it applies that always in a given voting moment we have $a_i \geq b_i$, then we have 5 such permissible combinations:

A A A B B B
A A B A B B
A A B B A B
A B A A B B
A B A B A B

As we can see, once again we got 5 key values that possess the balance property, only in this case they were not presented in binary form, instead they are presented in the form of *Ballot record*. Precisely, in papers [3,10] is stated the fact that combinatorial Ballot problem or voting lists problem is widely used in cryptography and in the method of recording and encryption.

Example 6: For the given key *K=877268*, for which we have already established that has Catalan number property and the given plaintext *P="SINGIDUNUM"*. If we present the key in a form of Ballot notation, then bit 1 is a vote for the first candidate A and it represents character reading operation from the message *P*, and bit 0 is a vote for

message	*S*	*I*	*N*	*G*	*I*	*D*	*U*	*N*	*U*	*M*									
voting key	1	1	0	1	0	1	1	0	0	0	1	0	1	1	0	1	0	1	0	0
voting phases	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
candidate A	1	2	2	3	3	4	5	5	5	5	6	6	7	8	8	9	9	10	10	10
candidate B	0	0	1	1	2	2	2	3	4	5	5	6	6	6	7	7	8	8	9	10
divergence (index)	1	2	2	3	3	4	5	5	/4/	/1/	6	6	7	8	8	9	9	10	10	/7/
ciphertext			I		N			I	G	S		D			N		U		M	U

Figure 1. *Encryption on the principle of the voting order (Ballot problem)*

the second candidate B and represents operation of writing in the ciphertext *C*. The resulting ciphertext is *C="INIGSDNUMU".*

Now we will introduce the possibility of applying the stack structure in the encryption process. The stack is an abstract type and data structure that is based on the *LIFO* principle *(last in, first out),* with two basic operations: *push and pop*.

It is also possible to generate stack permutations based on the Catalan number properties; by making the connection between the operation over the stack and the characters that appear in binary notation of a key that has Catalan number property:

1. If the current character in a record is 1, then the *push* operation is invoked and the number of appearances of this bit is located in a record on the left side.
2. When the value 0 appears, then the *pop* operation is invoked and the bit from the stack is pushed out to the exit.

In our paper [11], the number of permutations used in the stack corresponds to the Catalan number. Therefore, applying a stack can only map each binary record (or equivalent ballot record) of 2n length on one permutation of the set {1,2, ..., n}. A binary record is expected at the input, and since the push operation records only the number of occurrences of value 1, a two-time output is obtained twice (figure shows the relationship between the input and the output of the stack).

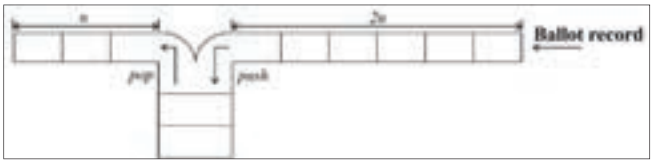


Figure 2. *The processing of input data through stack structure*

For the decryption procedure, it is necessary to allow, on the basis of output values obtained using the stack, the corresponding binary or *Ballot* record (reverse process). Output values obtained from the stack are determined by the sequence of operations over the stack. In our paper [11], the values obtained from the stack structure we have named SVB (*stack value based binary/ballot notation*). Below, an algorithm for mapping SVB records to the *Ballot* record is proposed.

Algorithm 1. Mapping SVB records in *Ballot* record [11]

Input: Stream P_i , i = *1, . . . , n* contains a set of permutation {1, . . . , n} using stack.
1: for i = 1 to n do
1.1: If $P_i > 0$ send P_i characters A to the output..
1.2: Send one character B to the output..
1.3: If $P_i > 0$ for j = i + 1 all the way to n reduce P_j for P_i .
Output: The appropriate ballot record.

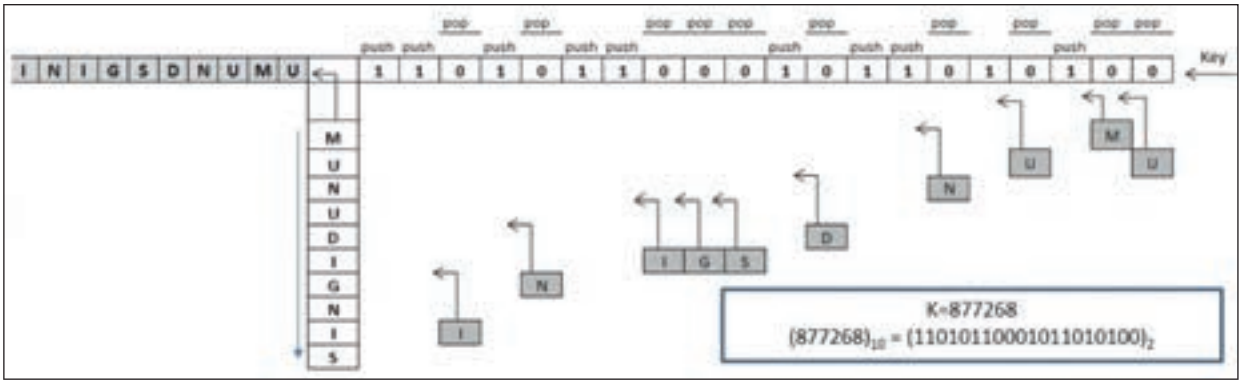


Figure 3. *Encryption based on the stack permutation principle*

Example 7: For the given key *K=877268*, for which we have already established that it has Catalan number property and the given plaintext *P="SINGIDUNUM"*. Bit 1 in the binary key record represents the *push* operation or the character reading from the message P, and the bit 0 represents the *pop* operation or writing in the ciphertext C. The resulting ciphertext is *C="INIGSDNUMU*".

A similar example of encryption, based on the sequences of the stack operations is given in the case study [2]. Equivalent to the stack permutations and the *Ballot* notation can only be the *Balanced Parentheses* notation, i.e., a record in a form of balanced parenthesis [4]. In our paper [9] thoroughly are considered the possibilities of representation of the Catalan numbers based on combinatorial problems of the *Balanced Parentheses*. In that paper we have provided a method for alphanumeric notation of paired brackets which is exactly based on the Catalan numbers properties.

We will display an example of text encryption based on *Balanced Parentheses* combinatorial problems, which is performed in a similar manner as with the stack permutations or Ballot problems.

Example 8: For the given key *K=877268*, for which we have already established that it has Catalan number property and the given plaintext *P="SINGIDUNUM".* If the key is presented in the form of paired parentheses, then bit 1 is opened parenthesis and presents reading operation of the characters from the message, and bit 0 is closed parenthesis and presents writing operation in the ciphertext. The last opened parenthesis must be closed first (LIFO). The resulting ciphertext is *C="INIGSDNUMU".*

Java software solution for encryption based on Ballot notation and Catalan-keys

Java application for encryption based on Ballot notation consists of three phases. The first phase involves finding Catalan numbers (keys) based on the given *n* basis. This phase involves the next steps:

- On the input *n* is assigned,
- On the basis of *n,* the set C_n (space of keys) is calculated,
- Selecting one Catalan number (key) from the C_n set,
- The selected key is converted from decimal to Ballot record (notation).

The second phase is the encryption process based on a key that has the Catalan number property. This phase includes the following steps:

- Loading the message (plaintext),
- Converting the message *(ASCII Text to Binary)* into binary sequence,
- The binary sequence is divided into *X* segments whose length corresponds to the *n* basis,
- By using the binary key record and reading it, starting from the first bit and ending with the last bit in the key (the occurrence of bit 1 denotes an open pair and 0 a closed pair), the sequence of bits permutation $X_1,....X_n$ is performed,
- The process of mixing the bits is performed according to the described principle of Ballot record (notation).
- The obtained permutations of the bits are converted - *Binary to ASCII Text,* and in this way the cipher text is obtained.

The third phase is a decryption process based on a key that has the Catalan number property. This phase includes the following steps:

- Loading the cipher text (textual record)
- Converting the cipher text *(ASCII Text to Binary)* into binary sequence,
- The binary sequence is divided into X segments whose length corresponds to the *n* basis
- By using the binary key record and reading it in reverse order, starting from the last bit and ending with the first bit in the key (the occurrence of bit 0 denotes an open pair and 1 a closed pair), the sequence of bits permutation $X_1,.... X_n$ is performed.

Key (bin)	1	1	0	1	0	1	1	0	0	0	1	0	1	1	0	1	0	1	0	0
Balanced parentheses	(	(	)	(	)	(	(	)	)	)	(	)	(	(	)	(	)	(	)	)
message	S	I		N		G	I				D		U	N		U		M		
ciphertext			I		N			I	G	S		D			N		U		M	U

Figure 4. *Encryption in the form of balanced parentheses*

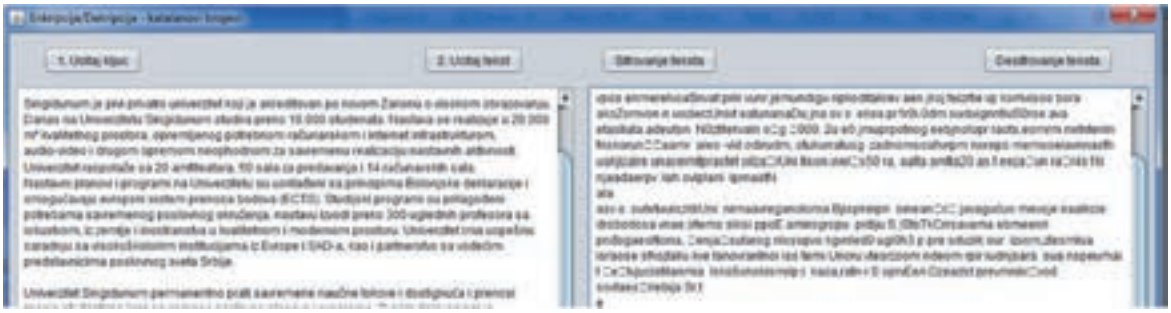


Figure 5. *Encryption of plaintext and displaying the ciphertext in the right panel*

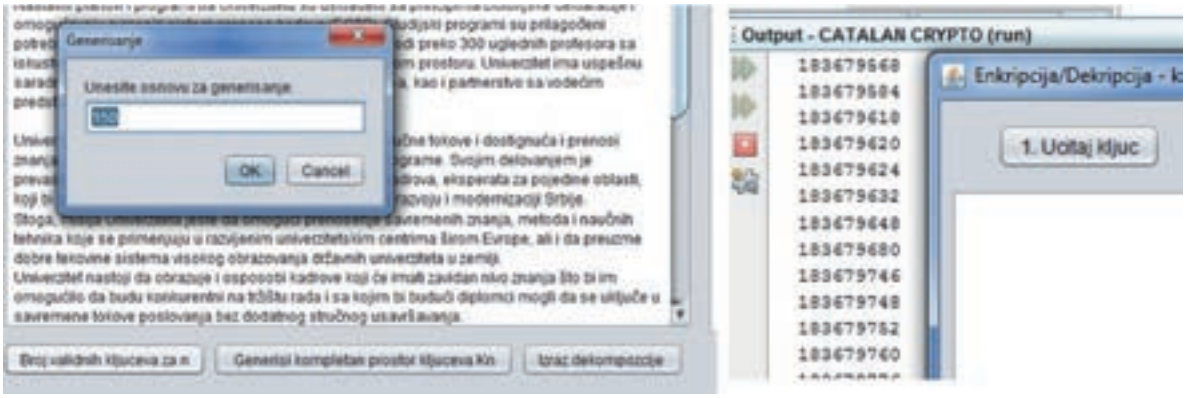


Figure 6. *Generating all valid keys for a given n basis*

- The process of mixing the bits is performed according to the described principle of Ballot record (notation), in reverse order,
- The obtained permutations of the bits are converted - *Binary to ASCII Text,* and in this way the source message is obtained.

For the implementation of Java software solution, it is important to note that we did not use a ready-made Java classes from the two standard APIs (JCA, JCE). The classes for working with files (*File, FileWriter, FileReader, BigInteger, Vector, logging.Logger...)* were used. In addition, we used the *Swing* and *AWT* package for programming the GUI application. The project consists of three classes: *CatalanCrypto, CatalanNumbers and CatalanDecomposition.*

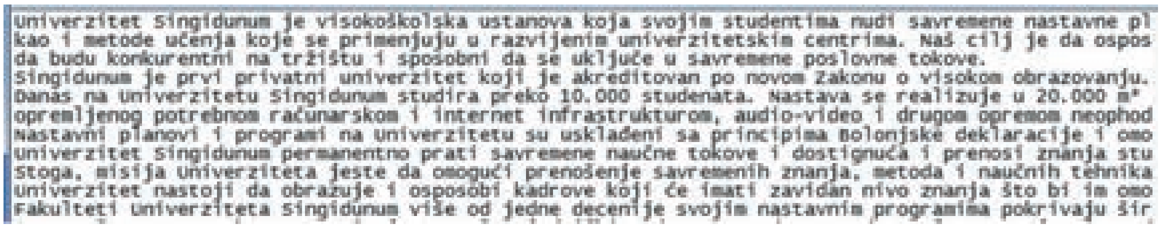
I) Within the *CatalanCrypto* class, the following methods are given:

- File encryptionCatalanBallot (File file, String key)
- File decryptionCatalanBallot (File file, String key)
- public String loadKey (File file).

II) The second class in our Java project is *CatalanNumbers* with a method for finding Catalan number: *find_catalan(File file).*

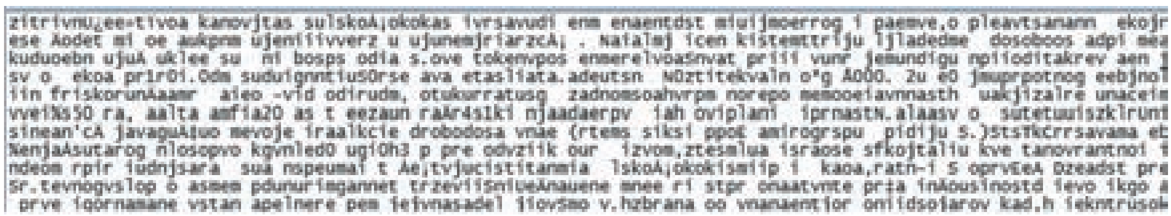
The main form of the *Java GUI application* has the following functionalities: Loading a key, Loading a file, Encryption of plaintext, Decryption of ciphertext, Checking the number of valid keys for given *n* (spaces of keys), Generating the keys that have the Catalan number properties, Dynamic generating of binary keys records. When starting the Java application, the first step is loading the key from an external file. In the application, there is an algorithm for generating a complete space of keys for a specific *n* basis. We use the method of manual taking of one of those values and storing them in the *KEY.TXT* file. After that, we include the specified file as the current active key. In this way, we can create multiple keys, but in one process we have to determine which key is active for the current encryption or decryption process.

After loading the key, the next step is loading the file with plaintext. File with the extension .txt, .doc can be loaded. After successful loading of the key and the message, the "*text encryption*" button becomes enabled. Otherwise, it is disabled before this. After getting the ciphertext, the "*text decryption*" button becomes enabled. Otherwise, before that this button was disabled.



Univerzitet Singidunum je visokoškolska ustanova koja svojim studentima nudi savremene nastavne pl
kao i metode učenja koje se primenjuju u razvijenim univerzitetskim centrima. Naš cilj je da ospos
da budu konkurentni na tržištu i sposobni da se uključe u savremene poslovne tokove.
Singidunum je prvi privatni univerzitet koji je akreditovan po novom Zakonu o visokom obrazovanju.
Danas na Univerzitetu Singidunum studira preko 10.000 studenata. Nastava se realizuje u 20.000 m²
opremljenog potrebnom računarskom i internet infrastrukturom, audio-video i drugom opremom neophod
Nastavni planovi i programi na Univerzitetu su usklađeni sa principima Bolonjske deklaracije i omo
Univerzitet Singidunum permanentno prati savremene naučne tokove i dostignuća i prenosi znanja stu
Stoga, misija Univerziteta jeste da omogući prenošenje savremenih znanja, metoda i naučnih tehnika
Univerzitet nastoji da obrazuje i osposobi kadrove koji će imati zavidan nivo znanja što bi im omo
Fakulteti Univerziteta Singidunum više od jedne decenije svojim nastavnim programima pokrivaju šir

Figure 7. *The content of the plaintext (message)*



zitrivnu¿ee=tivoa kanovjtas sulskoA¡okokas ivrsavudi enm enaentdst miuijmoerrog i paemve,o pleavtsanann ekojr
ese Aodet mi oe aukpnm ujeniiivverz u ujunemjriarzcA¡ . Naialmj icen kistemttriju ijladedme dosoboos adpi mea
kuduoebn ujuA uklee su ni bosps odia s.ove tokenvpos enmerelvoaŠnvat priii vunr jemundigu npiioditakrev aen i
sv o ekoa prirOi.Odm suduignntiusOrse ava etasliata.adeutsn NOztitekvaln o°g A000. Zu eO jmuprpotnog eebjno
iin friskorunAaamr aieo -vid odirudm, otukurratusg zadnomsoahvrpm norepo memooeiavnnasth uakjizaire unaceim
vveiN550 ra, aalta amfia20 as t eezaun raAr4siki njaadaerpv iah oviplani iprnastN.alaasv o sutetuuiszklrUni
sinean'cA javaguA¿uo mevoje iraalkcie drobodosa vnae (rtems siksi ppoɛ amirogrspu pidiju S.)StsTkCrrsavama et
NenjaAsutarog nlosopvo kgvnledO ugiOh3 p pre odvziik our izvom,ztesmlua israose sfkojtaliu kve tanovrantnoi i
ndeom rpir iudnjsara sua nspeumai t Ae¡tvjucistitanmia lskoA¡okokismiip i kaoa,ratn-i S oprvEeA Dzeadst pre
Sr.tevnogvslop o asmem pdunurimgannet trzeviiSnieAnauene mnee ri stpr onaatvnte prɟa inAousinostd ievo ikgo a
prve iqornamane vstan apelnere pem iejvnasadel iiovSmo v.hzbrana oo vnanaentior oniidsoiarov kad.h ieknrusok

Figure 8. *Content of the encrypted text based on the 60-bit Catalan number-key K=1142920503054772772, Ballot key record ((((()((())(((()))))()()))()((((()))() (((())()()()))()())*

By clicking the button "*text decryption*", we can decrypt ciphertext and compare it to the original message.

In Figure 6, a method for generating keys for a given *n* base is given. The condition to start generating the entire space of keys for particualar *n* basis is to determine the file in which the entire space of keys will be recorded. After that, generating and recording of keys starts. This process may take a time, depending on the input of *n* basis.

Conculsion

In this paper, we investigated and presented how Catalan numbers are widely used in solving many combinatorial problems, such as stack permutations, paired parentheses problem, *Ballot* problem etc. Having in regard the fact that cryptography is very dinamic field, that it is up to date and very widespread, this paper covers only some of its mathematical concepts and gives a contribution when it comes to the application of number theory in the field of cryptography.

Also, we mentioned the theoretical basis of the research where the basic Catalan numbers properties were tested, and the focus was placed on the bit balance property in binary notation of the Catalan number. After that, we gave a few suggestions and examples of combinatorial problem application in encrypting files. We put the emphasis on the application of stack permutation in text encryption, where also the equivalent combinatorial problems in encoding were displayed, such as the *Ballot* problem, stack permutations and *Balanced Parentheses*.

Our theoretical basis of the research was related to the method of keys generating based on the bit balance property of Catalan number and the encryption based on Ballot notation. Specifically, a case study is given that includes specific algorithms for encryption and decryption, which are implemented in the Java programming language. The implemented GUI application has all necessary elements for easy and efficient files encrypt and decrypt, keys loading, displaying the content of ciphertext and messages, keys generating, etc.

References

[1] Amounas F, El-Kinani H, Hajar M (2013) Novel Encryption Schemes Based on Catalan Numbers. *International Journal of Information & Network Security*, 2 (4), 339-347.

[2] Chouse C (2002) Catalan sayılarını kullanarak Stack Permutasyonu yöntemi ile bir dosyann şifrelenmesi [on-line]. www.chasanc.com/ [Available 24.05.2017.]

[3] Damgard I, Groth J, Salomonsen G (2003) The Theory and Implementation of an Electronic Voting System. *Advances in Information Security*, 7, 77-99.

[4] Geary FR, Rahman N, Raman R (2006) A Simple Optimal Representation for Balanced Parentheses. *Theoretical Computer Science*, 368 (3), 231-246.

[5] Higgins PM (2008) Number Story: From Counting to Cryptography. Springer Science & Business Media, Berlin, Germany.

[6] Horak P, Semaev I, Tuza IZ (2015) An application of Combinatorics in Cryptography. *Electronic Notes in Discrete Mathematics,* 49, 31-35.

[7] Koshy T (2009) Catalan Numbers with Applications. Oxford University Press, New York.

[8] Lachaud G, Ritzenthaler C, Tsfasman MA (2009) Arithmetic, Geometry, Cryptography, and Coding Theory. American Mathematical Society, United States.

[9] Mašović S, Saračević M, Stanimirović P (2014) Alpha-Numeric notation for one Data Structure in Software Engineering. *Acta Polytechnica Hungarica: Journal of Applied Sciences,* 11 (1), 193-204.

[10] Rivest RL (2006) The ThreeBallot Voting System: Publication on Computer Science and Artificial Intelligence. Massachusetts Institute of Technology [on-line]. https://people.csail.mit.edu/rivest/pubs/Riv06c.pdf [Available 24.05.2017.]

[11] Saračević M, Stanimirović P, Krtolica P, Mašović S (2014) Construction and Notation of Convex Polygon Triangulation based on ballot problem. *ROMJIST- Journal of Information Science and Technology*, 17 (3), 237-251.

[12] Saračević M (2013) Methods for solving the polygon triangulation problem and their implementation (in Serbian). PhD thesis, Faculty of Science and Mathematics, University of Niš.

[13] Saračević M (2017). Application of Catalan numbers and some combinatorial problems in cryptography (in Serbian). BSc thesis, Faculty of informatics and computing, Singidunum University in Belgrade.

[14] Stanimirović P, Krtolica P, Saračević M (2014) Decomposition of Catalan numbers and Convex Polygon Triangulations. *International Journal of Computer Mathematics,* 91 (6), 1315-1328.

[15] Stanley RP (2012), Catalan addendum to Enumerative Combinatorics [on-line]. http://www-math.mit.edu/~rstan/ec/catadd.pdf. [Available 24.05.2017.]

[16] Srikantaswamy SG, Phaneendra HD (2012) A Cryptosystem Design with Recursive Key Generation Techniques. *Procedia Engineering*, 30, 170 – 173.

[17] Takacs L (1962) A Generalization of the Ballot Problem and its Application in the Theory of Queues. *Journal of the American Statistical Association*, 57, (298), 327-337.

Submitted: November 3, 2017.
Accepted: November 10, 2017.

About the authors

Muzafer Saračević (1984) is a associate professor of computer sciences at the University of Novi Pazar and dean of the Department of computer sciences at the same university. He defend bachelor thesis on Faculty of Informatics and Computing in Belgrade, master thesis on Faculty of Technical Sciences (University of Kragujevac) and 2013th year defend doctoral thesis on Faculty of Science and Mathematics (University of Niš). He specialized on Oracle academy in the field of database and programming. He is the author of over 120 professional and scientific papers, one monograph and practicum. He is a member of the editorial board for five journals.

Edin Korićanin (1989) is a teaching assistant of computer sciences at the University of Novi Pazar. He defend bachelor thesis on Department of Computer Sciences in Novi Pazar and master thesis on Faculty of Technical Sciences (University of Kragujevac). He specialized on Oracle academy in the field of programming. He is the author of 20 professional and scientific papers. He is a tehnical editor of two journals.

Enver Biševac (1980) is a teaching assistant of computer sciences at the University of Novi Pazar. He defend bachelor thesis on Department of Computer Sciences in Novi Pazar and master thesis on Faculty of Technical Sciences (University of Novi Sad). He is the author of 10 professional and scientific papers.