



Cryptography and Cryptanalysis in MATLAB

Creating and Programming
Advanced Algorithms

Marius Iulian Mihailescu
Stefania Loredana Nita

Apress®

Cryptography and Cryptanalysis in MATLAB

**Creating and Programming
Advanced Algorithms**

**Marius Iulian Mihailescu
Stefania Loredana Nita**

Apress®

Cryptography and Cryptanalysis in MATLAB: Creating and Programming Advanced Algorithms

Marius Iulian Mihailescu
Bucharest, Romania

Stefania Loredana Nita
Bucharest, Romania

ISBN-13 (pbk): 978-1-4842-7333-3
<https://doi.org/10.1007/978-1-4842-7334-0>

ISBN-13 (electronic): 978-1-4842-7334-0

Copyright © 2021 Marius Iulian Mihailescu and Stefania Loredana Nita

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

Trademarked names, logos, and images may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, logo, or image we use the names, logos, and images only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

The use in this publication of trade names, trademarks, service marks, and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Managing Director, Apress Media LLC: Welmoed Spahr
Acquisitions Editor: Steve Anglin
Development Editor: Matthew Moodie
Editorial Operations Manager: Mark Powers

Cover designed by eStudioCalamar

Cover image by Devin Avery on Unsplash (www.unsplash.com)

Distributed to the book trade worldwide by Apress Media, LLC, 1 New York Plaza, New York, NY 10004, U.S.A. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail booktranslations@springernature.com; for reprint, paperback, or audio rights, please e-mail bookpermissions@springernature.com.

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at <http://www.apress.com/bulk-sales>.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub via the book's product page, located at www.apress.com/9781484273333. For more detailed information, please visit <http://www.apress.com/source-code>.

Printed on acid-free paper

To our families and wonderful readers. May this book be a true inspiration for everyone...and remember: "If you think technology can solve your security problems, then you don't understand the problems and you don't understand technology." – Bruce Schneier

Table of Contents

About the Authors..... xi

About the Technical Reviewer xiii

Chapter 1: Getting Started in Cryptography and Cryptanalysis..... 1

 Cryptography and Cryptanalysis 2

 Book Structure 3

 Conclusion 5

 References..... 6

Chapter 2: MATLAB Cryptography Functions..... 7

 Conclusion 15

 References..... 16

Chapter 3: Conversions Used in MATLAB for Cryptography..... 17

 Conclusion 23

 References..... 23

Chapter 4: Basic Arithmetic Foundations..... 25

 Euclid’s Division Lemma 26

 Greatest Common Divisor (gcd) 27

 Euclid’s Algorithm 27

 The Extended Euclidean Algorithm 28

 Practical Implementations 28

 The Extended Euclidean Algorithm..... 28

 Prime Factors in MATLAB 30

 Computing the Modular Inverse 31

 Conclusion 32

 References..... 32

TABLE OF CONTENTS

Chapter 5: Number Theory..... 35

 Primality and Factorization 35

 Prime Numbers..... 35

 The Prime Number Theorem..... 38

 Congruency..... 39

 Inverse..... 39

 Chinese Remainder Theorem 39

 Primality Tests..... 40

 The Wilson Primality Test..... 40

 The Little Fermat Primality Test..... 42

 The Miller-Rabin Primality Test..... 47

 Algebraic Structures 47

 Conclusion 49

 References..... 49

Chapter 6: Classic Cryptography 51

 Symmetric Cryptography 51

 Classic Ciphers 52

 The Caesar Cipher 53

 The Vigenère Cipher 57

 The Hill Cipher 63

 Conclusion 66

 References..... 67

Chapter 7: Pseudo-Random Number Generators..... 69

 Simple PRNGs 71

 Linear Congruential Generators..... 71

 Ranrot Generators 72

 Blum-Blum-Shub Generator 73

 Linear Circuit PRNGs..... 74

 Other PRNGs 75

 Practical Implementations 75

Conclusion	80
References.....	80
Chapter 8: Hash Functions.....	83
Security of Hash Functions	84
Cryptographic Hash Functions	84
Birthday Attack	85
MD4 Function	89
MD4 Function Description	89
Cryptanalysis of MD4.....	93
MD5 Function.....	94
SHA1 Function	95
Implementing Hash Functions	96
Implementing SHA-1/256/368/512, MD2, and MD5	97
Conclusion	101
References.....	101
Chapter 9: Block Ciphers: DES and AES.....	103
Preliminaries.....	103
Networks Based on Substitution and Permutation	105
Attacks Based on Linear Cryptanalysis.....	106
Attacks Based on Differential Cryptanalysis	107
The Data Encryption Standard (DES).....	108
DES Description	109
Implementation of DES	110
The Advanced Encryption System (AES)	114
SubBytes Operations	116
The ShiftRows Operation.....	117
The MixColumn Operation	117
The AddRoundKey Operation	118
Key Expansion	118
InvSubBytes Operation	119

TABLE OF CONTENTS

InvShiftRows Operation 119

InvMixColumns Operation..... 119

Conclusion 120

References..... 120

Chapter 10: Asymmetric Encryption Schemes 125

RSA 127

ElGamal 132

Merkle-Hellman 138

 Knapsack Approach..... 138

 The Algorithms 138

Conclusion 139

References..... 140

Chapter 11: Formal Techniques for Cryptography 143

Probability Theory 143

 Random Variables..... 145

 Birthday Problem..... 145

 Entropy 147

Randomness in Cryptography 148

Conclusion 151

References..... 151

Chapter 12: Visual Cryptography..... 153

Conclusion 158

References..... 158

Chapter 13: Chaos-Based Cryptography..... 159

Chaos Maps and Functions..... 160

 Logistic Map 161

Chaos Theory in Cryptography 162

 Sensitivity to Initial Conditions 172

Conclusion 173

References..... 174

Chapter 14: Steganography	179
Algorithms and Methods for Different Steganography Types.....	180
Steganography for Image Files.....	180
Steganography for Audio Files.....	180
Steganography for Video Files.....	181
Practical Implementation	181
Implementing the Least Significant Bit (LSB) Method	181
Implementing the Histogram Analysis Method.....	187
Conclusion	189
References.....	190
Index.....	191

About the Authors

Marius Iulian Mihailescu, PhD is an Associate Professor (Senior Lecturer) at the Spiru Haret University of Bucharest, Romania. He is also the CEO of Dapyx Solution Ltd., a company based in Bucharest, Romania that is focused on information security- and cryptography-related research projects. He is a lead guest editor for applied cryptography journals and a reviewer of multiple publications with information security and cryptography profiles. He has authored and co-authored more than 30 articles in conference proceedings, 25 articles in journals, and four books. For more than six years, he has served as a lecturer at well-known national and international universities (University of Bucharest, Titu Maiorescu University, Kadir Has University, Istanbul, Turkey). He has taught courses on programming languages (C#, Java, C++, and Haskell) and object-oriented system analysis and design with UML, graphs, databases, cryptography, and information security. He served for three years as an IT Officer at Royal Caribbean Cruises Ltd., where he dealt with IT infrastructure, data security, and satellite communications systems. He earned his PhD in 2014 with a thesis on applied cryptography over biometrics data. He holds two MSc degrees, in information security and software engineering.

Stefania Loredana Nita, PhD is a software developer and researcher at the Institute for Computers of the Romanian Academy. She earned her PhD with a thesis on advanced cryptographic schemes using searchable encryption and homomorphic encryption. At the Institute for Computers, she focuses on research and development projects that involve searchable encryption, homomorphic encryption, cloud computing security, the Internet of Things, and big data. She has served for more than two years as an assistant lecturer at the University of Bucharest, where she teaches advanced programming techniques, simulation methods, and operating systems. She has authored and co-authored more than 25 papers for conferences and journals and has co-authored four books. She holds an MSc in software engineering and two BSc degrees in computer science and mathematics.

About the Technical Reviewer



Irfan Turk, PhD, is a math and computer programming instructor and has worked at universities, high schools, and educational institutions for about 15 years. He concentrated on applied mathematics while earning his PhD degree. Dr. Turk finished the computer science track requirements of his MSc degree while a student at the University of Texas at Arlington. He is the author of the *Python Programming: for Engineers and Scientists* and *MATLAB Programming: for Beginners and Professionals* books. Dr. Turk's research interests include, but are not limited to, numerical solutions of differential equations, scientific computing, mathematical modeling, and programming in MATLAB and Python.

CHAPTER 1

Getting Started in Cryptography and Cryptanalysis

Due to the most recent attacks, the electronic communications and transaction require to strengthen their practical security techniques related to digital signature in such way that law enforcement agencies are able to recognize and trust them.

In the last few months alone, as we were writing this book, there were countless news stories detailing attacks that compromised the private data of millions of users on the Internet or dark web. Advanced technologies—such as the Internet of Things, fog computing, edge computing, smart vehicles, drones, smart houses, and many other complex software applications (desktop/web/mobile)—are evolving so fast that it is a challenge to keep up with their security requirements. For example, looking at the CVE platform¹, we can see a total of 152,984 records (vulnerabilities) and exposed users. By examining how these vulnerabilities occurred, developers can identify efficient ways to protect users and customers against malicious hackers.

A security solution is successful when a user's machine (computer or other electronic device) and the networks with which it communicates do not expose the data flowing between them. Because of the increasing speed and complexity of computing technology (e.g., quantum computers), the mission of modern cryptography (and we are not referring to quantum cryptography) is at a very challenging point in time.

¹ CVE - <https://cve.mitre.org/>

Keeping knowledge secure is one of the most important aspects to consider when designing and implementing complex systems. Information falling into the wrong hands can result in huge financial losses or, in extreme cases, even loss of lives. Cryptography can be used to encode private information in such a way that nobody should be able to decode it without the necessary privileges. However, cryptography algorithms, have been broken after hackers found a flaw in their design. When enough computing power is applied (for example quantum computers) to break an encoded message or cryptography algorithm (such as RSA), it is only a matter of time until the encryption or algorithm will be broken.

The current work represents a continuation of previous works, such as [2] and [3], dedicated to applied cryptography using MATLAB environment for developing cryptography algorithms in a more related scientific manner. The book represents an advanced work. It provides a comprehensive view of the most important topics in information security, cryptography, and cryptanalysis. The book can be used in many areas by multiple professionals, including security experts, military experts and researchers, ethical hackers, teachers in academia, researchers, software developers, and software engineers. If you need security and cryptographic solutions in a real business software environment, this book represents a very good starting point, together with [1] and [2]. The book will serve very useful for students (undergraduate- and graduate-level, master degree, professional, and academic doctoral degree students), business analysts, and many others.

Cryptography and Cryptanalysis

There are three main concepts to keep in mind when dealing with information security and data protection. Those concepts are *cryptology*, *cryptography*, and *cryptanalysis*.

- “*Cryptology* is defined as the science or art of secret writings; the main goal is to protect and defend the secrecy and confidentiality of the information with the help of cryptographic algorithms.” [2, 3].
- “*Cryptography* represents the defensive side of the cryptology; the main objective is to create and design the cryptographic systems and their rules. When we are dealing with cryptography, we can observe a special kind of art; an art that is based on protecting the information by transforming it into an unreadable format, called ciphertext.” [2, 3].

- “*Cryptanalysis* is the offensive side of the cryptology; its main objective is to study the cryptographic systems with the scope to provide the necessary characteristics in such a way as to fulfill the function for which they have been designed. Cryptanalysis can analyze the cryptographic systems of third parties through the cryptograms realized with them, in such a way that breaks them to obtain useful information for their business purposes. Cryptanalysts, code breakers, or ethical hackers are the people who are dealing with the field of cryptanalysis.” [2, 3].
- “*Cryptographic primitive* represents a well-established or low-level cryptographic algorithm used to build cryptographic protocols. Examples of such routines include hash functions and encryption functions.” [2, 3].

The book provides a deep examination of all three concepts from the practical side to the theoretical side. It illustrates how a theoretical algorithm should be analyzed for implementation.

Book Structure

The book is divided into 14 chapters meant to cover the most important practical and theoretical aspects of modern cryptography and cryptanalysis. The chapters are structured in such a manner that they cover the key elements, from theoretical cryptography to applied cryptography, especially when implementing the algorithms in MATLAB.

Each chapter is structured into two main parts—the *mathematical background*, which provides the notions required in the implementation process, and the *implementation*, which contains examples of how it should be done. The chapter goals are as follows:

- *Chapter 1*. This chapter’s goal is to highlight the importance of cryptography and cryptanalysis, justifying the importance of applied cryptography in a continuously evolving period of technology and requirements.

- *Chapter 2.* This chapter gives a short introduction to the built-in functions from MATLAB that are used to implement the cryptographic algorithms.
- *Chapter 3.* This chapter explains the different procedures for converting text (messages/plaintext) from lowercase to uppercase, from ASCII codes to characters, etc. These conversions are very useful in cryptography for some special algorithms.
- *Chapter 4.* This chapter covers the main arithmetic operations that are dedicated to cryptography algorithms. Every arithmetic operation is justified through an example, with the goal to make it clear as it how it should be done correctly in MATLAB.
- *Chapter 5.* In cryptography, especially in theoretical cryptography, number theory represents an important part of the research when designing and proposing new cryptography algorithms. Number theory is a fascinating and complex field. Without a proper understanding of its theoretical concepts, developers cannot perform a correct and reliable implementation of a cryptography algorithm. The purpose of this chapter is to give important highlights of important aspects that are useful to those who want to write cryptography algorithms from scratch.
- *Chapter 6.* This chapter contains the implementation of well-known algorithms (e.g., Caesar, Vigenère, and Hill) in order to provide the readers with a strong foundation of the main operations from MATLAB that are used to implement the cryptography algorithms. Each algorithm is done with respect to the two main operations, encryption and decryption.
- *Chapter 7.* This chapter covers the most important aspects of generating random numbers. This chapter is dedicated to those cryptography algorithms in which generating high random numbers is an important part of the security of the cryptography algorithm.
- *Chapter 8.* This chapter presents an implementation of hash functions, such as MD4/MD5, SHA1/256/368/512. It explores the fascinating processes behind generating hashes for different types of data, such as text.

- *Chapter 9.* This chapter presents the main implementation procedure based on the proposed standards by NIST for DES (Data Encryption Standard) and AES (Advanced Encryption System). Designing s-boxes is an important concept in DES and AES [1]. The chapter will examine the construction of S-boxes and underline their importance in implementing cryptographic algorithms.
- *Chapter 10.* This chapter is dedicated to one of the most important types of cryptography, AES. This chapter presents advanced notions and concepts for public-key cryptography that are very useful in designing and implementing advanced, complex systems.
- *Chapter 11.* This chapter goes through a series of advanced, powerful algorithms, such as RSA and ElGamal. They are used every day in many applications (e.g., Internet browsers).
- *Chapter 12.* This chapter provides techniques for allowing visual information, such as pictures, to be encrypted in such way that the decrypted information is shown as a visual image.
- *Chapter 13.* This chapter represents encryption and decryption methods using chaos theory and chaos algorithms proposed at the theoretical level. Chaos-based cryptography is a controversial topic due to the complexity of the operations and the field itself.
- *Chapter 14.* This chapter provides different implementations of methods and algorithms for *steganography*, which is the art of hiding different types of data (text and files) in another type of media file (pictures files, text files, etc.).

Conclusion

In this first chapter, we discussed the objectives of this book, based on addressing the practical aspects of cryptography and information security. Due to the increasing number of requirements for developing secure software applications and using advanced information technologies, they have a deep impact on our lives every day.

The goal of this book is translate the most important theoretical cryptography algorithms and mechanism to practice using one of the most powerful technologies in research, MATLAB.

In this chapter you learned about:

- The mission and goals of this book.
- The differences between cryptography, cryptanalysis, and cryptology.
- The goal of each chapter and the algorithms presented.

References

- [1] *Modern Cryptography Applied Mathematics for Encryption and Information Security*, William Easttom, Springer, 2021.
- [2] Mihailescu, Marius Iulian, and Stefania Loredana Nita. *Pro Cryptography and Cryptanalysis: Creating Advanced Algorithms with C# and .NET*. Apress, 2021. DOI: 10.1007/978-1-4842-6367-9.
- [3] Mihailescu, Marius Iulian, and Stefania Loredana Nita. *Pro Cryptography and Cryptanalysis with C++20: Creating and Programming Advanced Algorithms*. Apress, 2021. DOI: 10.1007/978-1-4842-6586-4.

CHAPTER 2

MATLAB Cryptography Functions

One of the most important elements in cryptography is the *bit*, the foundation of digital information. Working with bits requires different logical operations, such as OR, AND, or XOR. MATLAB implements these operators automatically, as well as some types of conversions to bits, as you will see in this chapter. Other important aspects of cryptography are *arrays* or *matrices*. These have many applications. For example, the elements of a finite group are stored in an array, and the key of an encryption system uses matrices to be generated. In this chapter, you will see some of the most important functions that work with bits and matrices.

In MATLAB, there are several functions already implemented that work directly with bits to implement the main operations on bits (*bitwise* operations). Examples of such functions are `bitset`, `bitget`, `bitshift`, `bitcmp`, `bitor`, `bitand`, `bitxor`, and `swapbytes` [1]. They all take numeric values as input. These functions are defined as follows:

- **Bitset:** Sets a bit value on a given location.
- **Bitget:** Gets the bit value from a given location.
- **Bitshift:** Shifts the bits with a specific number of locations. The sense of shifting (left or right) and the bits that fill the initial places are given by the signs of the bits that need to be shifted and the number of locations to be shifted.
- **Bitcmp:** Returns the complement of the bit(s) given as input.
- **Bitor:** Returns the result of the OR operation between the bits.
- **Bitand:** Returns the result of the AND operation between the bits.

- Bitxor: Returns the result of the XOR operation between the bits.
- Swapbytes: Swaps of the order of the bytes.

Listing 2-1 presents the applications of these functions; the result is shown in Figure 2-1.

Listing 2-1. Using a Bitwise Function from MATLAB

```

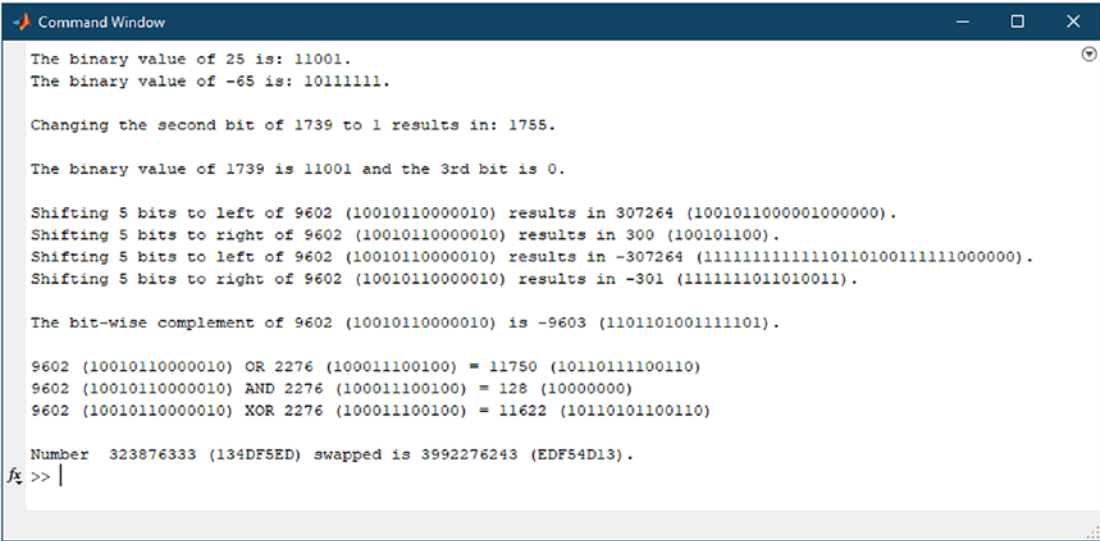
1
2  clc;
3  clear all;
4
5  no1 = 25;
6  bin1 = dec2bin(no1);
7  fprintf("The binary value of %d is: %s. \n", no1, bin1);
8  no2 = -65;
9  bin1 = dec2bin(no2);
10 fprintf("The binary value of %d is: %s. \n", no2, bin1);
11 fprintf("\n");
12
13 no3 = 1739;
14 set_res = bitset(no3, 5);
15 fprintf("Changing the second bit of %d to 1 results in: %d. \n", no3,
16 set_res);
17 fprintf("\n");
18
19 get_res = bitget(no3,3);
20 fprintf("The binary value of %d is %s and the 3rd bit is %d. \n", no3,
21 dec2bin(no1), get_res);
22 fprintf("\n");
23
24 no4 = round(2000 +(10000-2000)*rand(1,1));
25 shift_res = bitshift(no4, 5);
26 fprintf("Shifting 5 bits to left of %d (%s) results in %d (%s). \n",
27 no4, dec2bin(no4), shift_res, dec2bin(shift_res));
28 shift_res2 = bitshift(no4, -5);

```

```

29 fprintf("Shifting 5 bits to right of %d (%s) results in %d (%s). \n",
30 no4, dec2bin(no4), shift_res2, dec2bin(shift_res2));
31 shift_res = bitshift(0-no4, 5, 'int64');
32 fprintf("Shifting 5 bits to left of %d (%s) results in %d (%s). \n",
33 no4, dec2bin(no4), shift_res, dec2bin(shift_res));
34 shift_res2 = bitshift(0-no4, -5, 'int64');
35 fprintf("Shifting 5 bits to right of %d (%s) results in %d (%s). \n",
36 no4, dec2bin(no4), shift_res2, dec2bin(shift_res2));
37 fprintf("\n");
38
39 cmp_res = bitcmp(no4, 'int64');
40 fprintf("The bit-wise complement of %d (%s) is %d (%s). \n", no4,
41 dec2bin(no4), cmp_res, dec2bin(cmp_res));
42 fprintf("\n");
43
44 no5 = round(2000 +(10000-2000)*rand(1,1));
45 fprintf("%d (%s) OR %d (%s) = %d (%s) \n", no4, dec2bin(no4), no5,
46 dec2bin(no5), bitor(no4,no5), dec2bin(bitor(no4,no5)));
47 fprintf("%d (%s) AND %d (%s) = %d (%s) \n", no4, dec2bin(no4), no5,
48 dec2bin(no5), bitand(no4,no5), dec2bin(bitand(no4,no5)));
49 fprintf("%d (%s) XOR %d (%s) = %d (%s) \n", no4, dec2bin(no4), no5,
50 dec2bin(no5), bitxor(no4,no5), dec2bin(bitxor(no4,no5)));
51 fprintf("\n");
52
52 no6 = 0x134DF5ED;
53 swap_res = swapbytes(no6);
54 fprintf("Number %d (%s) swapped is %d (%s). \n", no6, dec2hex(no6),
55 swap_res, dec2hex(swap_res));

```



```

Command Window

The binary value of 25 is: 11001.
The binary value of -65 is: 10111111.

Changing the second bit of 1739 to 1 results in: 1755.

The binary value of 1739 is 11001 and the 3rd bit is 0.

Shifting 5 bits to left of 9602 (100101100000010) results in 307264 (100101100000010000000).
Shifting 5 bits to right of 9602 (100101100000010) results in 300 (100101100).
Shifting 5 bits to left of 9602 (100101100000010) results in -307264 (111111111111011010011111000000).
Shifting 5 bits to right of 9602 (100101100000010) results in -301 (1111111011010011).

The bit-wise complement of 9602 (100101100000010) is -9603 (1101101001111101).

9602 (100101100000010) OR 2276 (100011100100) = 11750 (10110111100110)
9602 (100101100000010) AND 2276 (100011100100) = 128 (10000000)
9602 (100101100000010) XOR 2276 (100011100100) = 11622 (10110101100110)

Number 323876333 (134DF5ED) swapped is 3992276243 (EDF54D13).
fx >> |

```

Figure 2-1. The results obtained using bitwise functions

We started with the `clc;` and `clear all;` commands, which are used to clear the Command Window, in order to remove all items from the workspace and release the corresponding occupied memory. MATLAB provides different types of conversion functions. One of these functions is `dec2bin`, which converts a decimal number into its corresponding binary representation. Another example of a conversion function is `dec2hex`, which converts a decimal number into its hexadecimal representation. A complete list of conversion functions can be found at [2].

Note that the code in Listing 2-1 uses the `rand` function for `no4` and `no5`, so every time the program runs, different results will be obtained. This function, without parameters, generates a random number in the $(0, 1)$ interval. However, Listing 2-1 uses `rand(1,1)`, which generates a 1×1 matrix, which actually means one element. In general, `rand(a,b)` generates a $a \times b$ matrix, with elements in the $(0, 1)$ interval. Note also that `no4` and `no5` are not in the interval $(0, 1)$ and we used a formula to generate them. The general formula is

$$x + (y - x) * rand(1,1)$$

to generate a number in the interval (x, y) . As we used $x = 2000$ and $y = 10000$, `no4` and `no5` were generated in this interval. The values for `no4` and `no5` are integer values because we used the `round` function to round the result.