DOcumentation

RSA Cryptosystem Implementation

Zhengxing Li (D00171597)

Cryptographic Foundations

Table of Contents

[1. Methodology 2](#_Toc480907387)

[1.1 BigInteger Class 2](#_Toc480907388)

[1.2 Key Pairs Creation 2](#_Toc480907389)

[1.3 Encrypt Message 3](#_Toc480907390)

[1.4 Decrypt Message 3](#_Toc480907391)

[2. Limitations 3](#_Toc480907392)

[3. Problems Encountered 3](#_Toc480907393)

[4. Example Output 4](#_Toc480907394)

[4.1 A Digit / Character 4](#_Toc480907395)

[ digit 4](#_Toc480907396)

[ letter 5](#_Toc480907397)

[ character 7](#_Toc480907398)

[4.2 A Word 9](#_Toc480907399)

[4.3 A Sentence 11](#_Toc480907400)

[4.4 Large Pieces of text 13](#_Toc480907401)

[ A Paragraph 13](#_Toc480907402)

[ Several Paragraphs 17](#_Toc480907403)

# Methodology

## BigInteger Class

BigInteger class plays an important role in my RSA cryptosystem implementation. BigInteger is very similar to Integer class, but it’s especially used to handle very large integer and the size of the integer is only limited by Java Virtual Machine.

I have two reasons to use BigInteger class in my implementation. The first one is when the numbers being calculated are too large for long or int types, it will cause error in program. Another one reason is the BigInteger class has many in-built methods which will do the calculation needed in my implementation efficiently. So, I don’t need to implement my own code which might be far less efficient.

In my implementation, I took advantage of the following in-built methods of BigInteger class:

* java.math.BigInteger.isProbablePrime(int certainty) – to determine if a BigInteger is prime
* java.math.BigInteger.equals(Object x) – to determine if two BigIntegers are equal to each other
* java.math.BigInteger.multiply(BigInteger val)
* java.math.BigInteger.subtract(BigInteger val)
* java.math.BigInteger.gcd(BigInteger val) – to get the greatest common divisor of two BigIntegers
* java.math.BigInteger.modInverse(BigInteger m) – to get the mod inverse of this BigInteger
* java.math.BigInteger.modPow(BigInteger exponent, BigInteger m) – to return the value of the BigInteger ^ exponent mod m

## Key Pairs Creation

For creating key pairs, I first ask users to input two prime numbers p and q which are integers. Because we need to use several in-built function of BigInteger class, then I transform the two integers in to BigInteger object.

Then I take advantage of BigInteger. isProbablePrime() and BigInteger.equals() functions to check if there two numbers are prime number and if they are equal to each other and user will be kept being asking to reenter number until the two numbers meet requirements.

Then I calculate N and ϕ(N) and ask user to enter another number e which should be co-prime with ϕ(N).

Then I get d by calculating the modulo inverse of e in ϕ(N) using modPow() function.

The last step is to store (e, N) and (d,N) into two BigInteger arrays and name them public key and private key respectively.

## Encrypt Message

First I prompt user to enter the message they wish to encrypt. Then I use toCharArray() function to convert string into characters and store in an char array.

To convert each character to a number, I simply cast each char variable to int and then I’ll get a numeric value which point to the original character in ASCHII table. This approach also enables me to encrypt any non-alphanumeric character.

Then I calculate the cipher number for each number of plain text using modPow() function and cast the cipher number to a string variable. Then I add ‘0’ to the front of each string whose length is less than the length of N to make the length of string equal to the length of N, this is make it easier to split cipher text into blocks when decrypting.

The last step is to add all those strings together, then I get the encrypted message.

## Decrypt Message

To decrypt cipher text, I accept cipher text from user and I split the cipher text string into blocks with the length of each equal to the length of N.

Then I cast each block of string to BigInteger and calculate the numeric value of plain text letter and then I cast the BigInteger to an integer which points to a character in ASCHII table.

The next step is to cast the int to char and use Character.toString() function to get the plain text letter back.

The last step is to add all those letter together, and then I get the decrypted message.

# Limitations

There is a limitation of Implementation and it’s also an insecure factor. The length of cipher text is a multiple of the length N. This may give the attacker a chance to find the pattern and break the cipher text.

The reason for this limitation is that if I don’t make the length of each block of cipher text equal to the length of N, I’ll not be able to correctly split cipher text into blocks and decrypt them. Because after modulo calculation, each block of cipher text can be any number between 0 and N, and of course its length is different from each other.

Another limitation of my implementation is that if several paragraphs or even the whole article are copied directly into the cryptosystem when encryption, it’ll cause an error. I haven’t figured out the reason yet, but one possible solution is to copy the paragraphs or article into a text box and then copy into the cryptosystem.

# Problems Encountered

The trickiest problem I have encountered when I was implementing this RSA cryptosystem is that I don’t know how to correctly break the cipher text into blocks of the same size, because after modulo calculation, each block of cipher text can be any number between 0 and N, and of course its length is different from each other, so I don’t the digit is with the last digit of the next digit.

After considering several solutions, I choose to the length of each block of cipher text equal to the length of N when encrypting. This solution enables to this cryptosystem to encrypt and decrypt message of any size, like a digit, a letter, a word, a sentence and even a large piece of text. However, it may become a security issue.

# Example Output

## A Digit / Character

### digit

Welcome! Please select option to proceed.

1. Create Key Pairs

2. Encrypt Message

3. Decrypt Message

4. Display Keys

5. Exit

Option: 2

Please enter the message you wish to encrypt:

1

Encoded message: 282

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome! Please select option to proceed.

1. Create Key Pairs

2. Encrypt Message

3. Decrypt Message

4. Display Keys

5. Exit

Option: 3

Please enter the message you wish to decrypt:

282

Decoded message: 1



### letter

Welcome! Please select option to proceed.

1. Create Key Pairs

2. Encrypt Message

3. Decrypt Message

4. Display Keys

5. Exit

Option: 2

Please enter the message you wish to encrypt:

a

Encoded message: 030

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome! Please select option to proceed.

1. Create Key Pairs

2. Encrypt Message

3. Decrypt Message

4. Display Keys

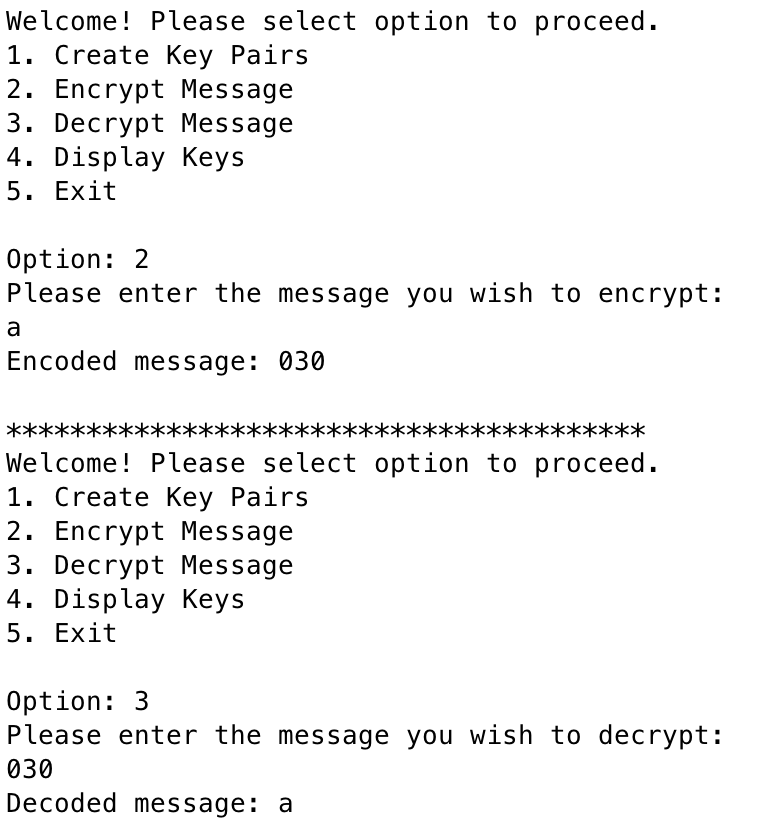
5. Exit

Option: 3

Please enter the message you wish to decrypt:

030

Decoded message: a



### character

Welcome! Please select option to proceed.

1. Create Key Pairs

2. Encrypt Message

3. Decrypt Message

4. Display Keys

5. Exit

Option: 2

Please enter the message you wish to encrypt:

?

Encoded message: 833

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome! Please select option to proceed.

1. Create Key Pairs

2. Encrypt Message

3. Decrypt Message

4. Display Keys

5. Exit

Option: 3

Please enter the message you wish to decrypt:

833

Decoded message: ?



## A Word

Welcome! Please select option to proceed.

1. Create Key Pairs

2. Encrypt Message

3. Decrypt Message

4. Display Keys

5. Exit

Option: 2

Please enter the message you wish to encrypt:

Crytography

Encoded message: 503091225047106084091030753054225

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome! Please select option to proceed.

1. Create Key Pairs

2. Encrypt Message

3. Decrypt Message

4. Display Keys

5. Exit

Option: 3

Please enter the message you wish to decrypt:

503091225047106084091030753054225

Decoded message: Crytography



## A Sentence

Welcome! Please select option to proceed.

1. Create Key Pairs

2. Encrypt Message

3. Decrypt Message

4. Display Keys

5. Exit

Option: 2

Please enter the message you wish to encrypt:

Alphanumeric is a combination of alphabetic and numeric characters, and is used to describe the collection of Latin letters and Arabic digits or a text constructed from this collection.

Encoded message: 152380753054030798417741151091025586542025046542030542586106741018025798030047025106798542106941542030380753054030018151047025586542030798303542798417741151091025586542586054030091030586047151091046273542030798303542025046542417046151303542047106542303151046586091025018151542047054151542586106380380151586047025106798542106941542540030047025798542380151047047151091046542030798303542152091030018025586542303025084025047046542106091542030542047151260047542586106798046047091417586047151303542941091106741542047054025046542586106380380151586047025106798115

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome! Please select option to proceed.

1. Create Key Pairs

2. Encrypt Message

3. Decrypt Message

4. Display Keys

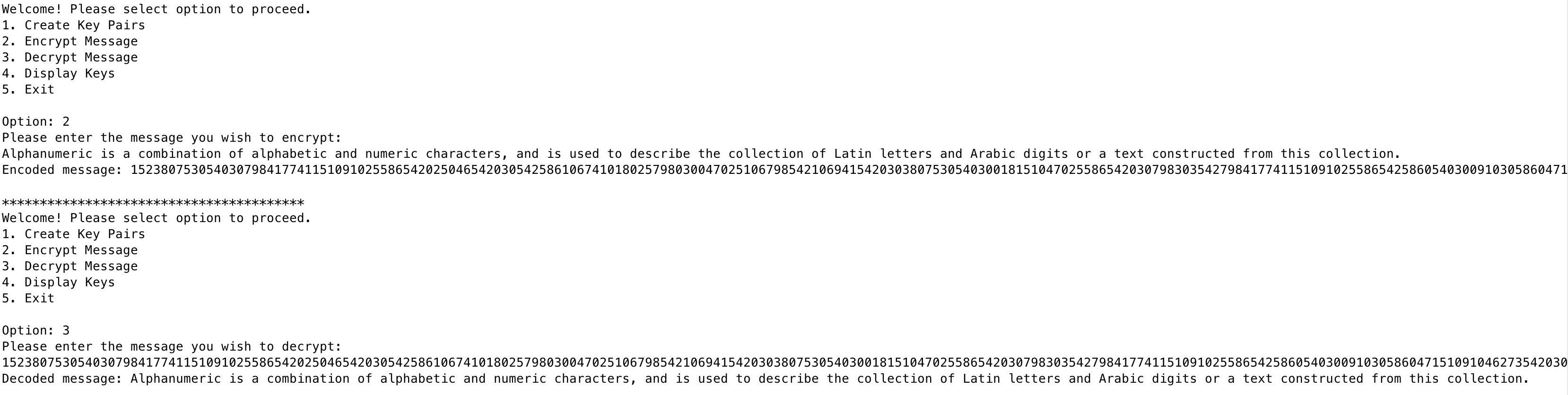
5. Exit

Option: 3

Please enter the message you wish to decrypt:

152380753054030798417741151091025586542025046542030542586106741018025798030047025106798542106941542030380753054030018151047025586542030798303542798417741151091025586542586054030091030586047151091046273542030798303542025046542417046151303542047106542303151046586091025018151542047054151542586106380380151586047025106798542106941542540030047025798542380151047047151091046542030798303542152091030018025586542303025084025047046542106091542030542047151260047542586106798046047091417586047151303542941091106741542047054025046542586106380380151586047025106798115

Decoded message: Alphanumeric is a combination of alphabetic and numeric characters, and is used to describe the collection of Latin letters and Arabic digits or a text constructed from this collection.



## Large Pieces of text

### A Paragraph

Welcome! Please select option to proceed.

1. Create Key Pairs

2. Encrypt Message

3. Decrypt Message

4. Display Keys

5. Exit

Option: 2

Please enter the message you wish to encrypt:

The code itself was patterned so that most control codes were together and all graphic codes were together, for ease of identification. The first two so called ASCII sticks[a][14] (32 positions) were reserved for control characters.[1]:220, 236 § 8,9) The "space" character had to come before graphics to make sorting easier, so it became position 20hex;[1]:237 § 10 for the same reason, many special signs commonly used as separators were placed before digits. The committee decided it was important to support uppercase 64-character alphabets, and chose to pattern ASCII so it could be reduced easily to a usable 64-character set of graphic codes,[1]:228, 237 § 14 as was done in the DEC SIXBIT code (1963). Lowercase letters were therefore not interleaved with uppercase. To keep options available for lowercase letters and other graphics, the special and numeric codes were arranged before the letters, and the letter A was placed in position 41hex to match the draft of the corresponding British standard.[1]:238 § 18 The digits 0–9 are prefixed with 011, but the remaining 4 bits correspond to their respective values in binary, making conversion with binary-coded decimal straightforward.

Encoded message: 226054151542586106303151542025047046151380941542584030046542753030047047151091798151303542046106542047054030047542741106046047542586106798047091106380542586106303151046542584151091151542047106084151047054151091542030798303542030380380542084091030753054025586542586106303151046542584151091151542047106084151047054151091273542941106091542151030046151542106941542025303151798047025941025586030047025106798115542226054151542941025091046047542047584106542046106542586030380380151303542152493503009009542046047025586433046114030231114282846231542327283032542753106046025047025106798046246542584151091151542091151046151091213151303542941106091542586106798047091106380542586054030091030586047151091046115114282231422032032486273542032283924370478370481273797246542226054151542498046753030586151498542586054030091030586047151091542054030303542047106542586106741151542018151941106091151542084091030753054025586046542047106542741030433151542046106091047025798084542151030046025151091273542046106542025047542018151586030741151542753106046025047025106798542032486054151260508114282231422032283243370478370282486542941106091542047054151542046030741151542091151030046106798273542741030798225542046753151586025030380542046025084798046542586106741741106798380225542417046151303542030046542046151753030091030047106091046542584151091151542753380030586151303542018151941106091151542303025084025047046115542226054151542586106741741025047047151151542303151586025303151303542025047542584030046542025741753106091047030798047542047106542046417753753106091047542417753753151091586030046151542924846482586054030091030586047151091542030380753054030018151047046273542030798303542586054106046151542047106542753030047047151091798542152493503009009542046106542025047542586106417380303542018151542091151303417586151303542151030046025380225542047106542030542417046030018380151542924846482586054030091030586047151091542046151047542106941542084091030753054025586542586106303151046273114282231422032032481273542032283243370478370282846542030046542584030046542303106798151542025798542047054151542413552503542493009198638009226542586106303151542327282797924283246115542540106584151091586030046151542380151047047151091046542584151091151542047054151091151941106091151542798106047542025798047151091380151030213151303542584025047054542417753753151091586030046151115542226106542433151151753542106753047025106798046542030213030025380030018380151542941106091542380106584151091586030046151542380151047047151091046542030798303542106047054151091542084091030753054025586046273542047054151542046753151586025030380542030798303542798417741151091025586542586106303151046542584151091151542030091091030798084151303542018151941106091151542047054151542380151047047151091046273542030798303542047054151542380151047047151091542152542584030046542753380030586151303542025798542753106046025047025106798542846282054151260542047106542741030047586054542047054151542303091030941047542106941542047054151542586106091091151046753106798303025798084542638091025047025046054542046047030798303030091303115114282231422032283481370478370282481542226054151542303025084025047046542486805797542030091151542753091151941025260151303542584025047054542486282282273542018417047542047054151542091151741030025798025798084542846542018025047046542586106091091151046753106798303542047106542047054151025091542091151046753151586047025213151542213030380417151046542025798542018025798030091225273542741030433025798084542586106798213151091046025106798542584025047054542018025798030091225482586106303151303542303151586025741030380542046047091030025084054047941106091584030091303115

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome! Please select option to proceed.

1. Create Key Pairs

2. Encrypt Message

3. Decrypt Message

4. Display Keys

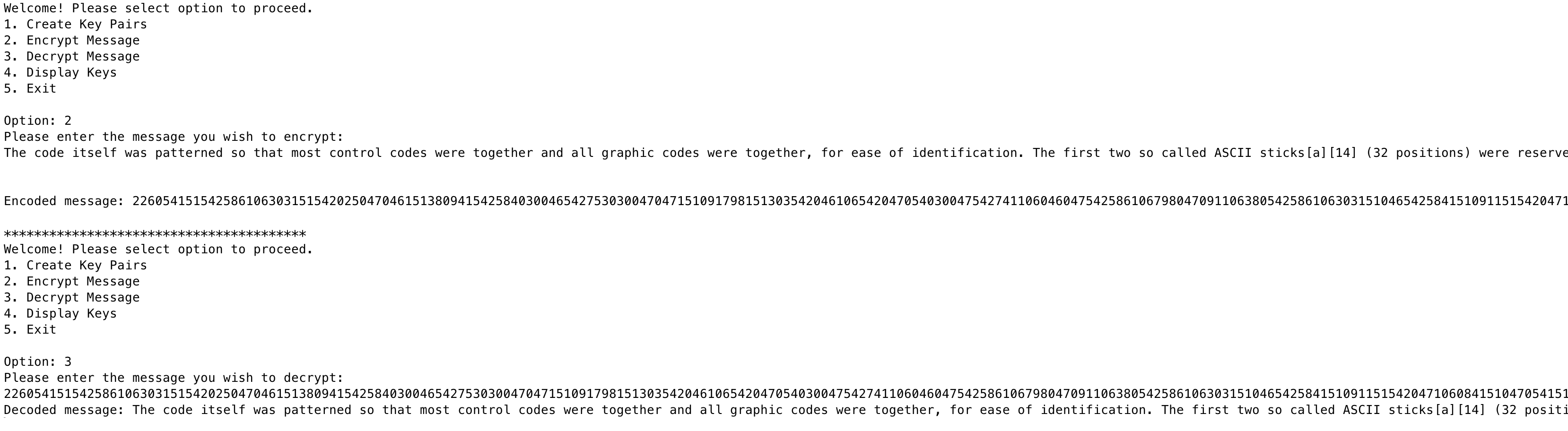
5. Exit

Option: 3

Please enter the message you wish to decrypt:

226054151542586106303151542025047046151380941542584030046542753030047047151091798151303542046106542047054030047542741106046047542586106798047091106380542586106303151046542584151091151542047106084151047054151091542030798303542030380380542084091030753054025586542586106303151046542584151091151542047106084151047054151091273542941106091542151030046151542106941542025303151798047025941025586030047025106798115542226054151542941025091046047542047584106542046106542586030380380151303542152493503009009542046047025586433046114030231114282846231542327283032542753106046025047025106798046246542584151091151542091151046151091213151303542941106091542586106798047091106380542586054030091030586047151091046115114282231422032032486273542032283924370478370481273797246542226054151542498046753030586151498542586054030091030586047151091542054030303542047106542586106741151542018151941106091151542084091030753054025586046542047106542741030433151542046106091047025798084542151030046025151091273542046106542025047542018151586030741151542753106046025047025106798542032486054151260508114282231422032283243370478370282486542941106091542047054151542046030741151542091151030046106798273542741030798225542046753151586025030380542046025084798046542586106741741106798380225542417046151303542030046542046151753030091030047106091046542584151091151542753380030586151303542018151941106091151542303025084025047046115542226054151542586106741741025047047151151542303151586025303151303542025047542584030046542025741753106091047030798047542047106542046417753753106091047542417753753151091586030046151542924846482586054030091030586047151091542030380753054030018151047046273542030798303542586054106046151542047106542753030047047151091798542152493503009009542046106542025047542586106417380303542018151542091151303417586151303542151030046025380225542047106542030542417046030018380151542924846482586054030091030586047151091542046151047542106941542084091030753054025586542586106303151046273114282231422032032481273542032283243370478370282846542030046542584030046542303106798151542025798542047054151542413552503542493009198638009226542586106303151542327282797924283246115542540106584151091586030046151542380151047047151091046542584151091151542047054151091151941106091151542798106047542025798047151091380151030213151303542584025047054542417753753151091586030046151115542226106542433151151753542106753047025106798046542030213030025380030018380151542941106091542380106584151091586030046151542380151047047151091046542030798303542106047054151091542084091030753054025586046273542047054151542046753151586025030380542030798303542798417741151091025586542586106303151046542584151091151542030091091030798084151303542018151941106091151542047054151542380151047047151091046273542030798303542047054151542380151047047151091542152542584030046542753380030586151303542025798542753106046025047025106798542846282054151260542047106542741030047586054542047054151542303091030941047542106941542047054151542586106091091151046753106798303025798084542638091025047025046054542046047030798303030091303115114282231422032283481370478370282481542226054151542303025084025047046542486805797542030091151542753091151941025260151303542584025047054542486282282273542018417047542047054151542091151741030025798025798084542846542018025047046542586106091091151046753106798303542047106542047054151025091542091151046753151586047025213151542213030380417151046542025798542018025798030091225273542741030433025798084542586106798213151091046025106798542584025047054542018025798030091225482586106303151303542303151586025741030380542046047091030025084054047941106091584030091303115

Decoded message: The code itself was patterned so that most control codes were together and all graphic codes were together, for ease of identification. The first two so called ASCII sticks[a][14] (32 positions) were reserved for control characters.[1]:220, 236ʑ§ʑ8,9) The "space" character had to come before graphics to make sorting easier, so it became position 20hex;[1]:237ʑ§ʑ10 for the same reason, many special signs commonly used as separators were placed before digits. The committee decided it was important to support uppercase 64-character alphabets, and chose to pattern ASCII so it could be reduced easily to a usable 64-character set of graphic codes,[1]:228, 237ʑ§ʑ14 as was done in the DEC SIXBIT code (1963). Lowercase letters were therefore not interleaved with uppercase. To keep options available for lowercase letters and other graphics, the special and numeric codes were arranged before the letters, and the letter A was placed in position 41hex to match the draft of the corresponding British standard.[1]:238ʑ§ʑ18 The digits 0ʛ9 are prefixed with 011, but the remaining 4 bits correspond to their respective values in binary, making conversion with binary-coded decimal straightforward.



### Several Paragraphs

Welcome! Please select option to proceed.

1. Create Key Pairs

2. Encrypt Message

3. Decrypt Message

4. Display Keys

5. Exit

Option: 2

Please enter the message you wish to encrypt:

Remember me functions are convenient for users as they don't have to type in a username and password each time they use an application on a particular device. However, sometimes this functionality is badly designed and may leave the application vulnerable to attack. Some applications use a persistent cookie that contains the username. When the application receives the cookie, the application trusts the cookie and authenticates the user. This bypasses the login and allows an attacker not only to gain access to one users account but also to change the username in the cookie to access other users accounts. Other applications may use a cookie with a session ID. When the session ID is presented to the application, the application checks for the user associated with that ID and creates an application session for that user. If the session ID's can be easily be predicted the attacker can go through a large number of session ID's to find ones associated with users, to gain access to users accounts without authentication. In general, client-side remember me functions should only remember non-secret information such as the username. The user may be allowed to opt-in to functionality which will remember the password. The user should be warned of the risks. Cleartext passwords should never be stored on the client as they are vulnerable to both local and remote attacks. Passwords should be encrypted using public key encryption, with the private key stored on the application server.

Encoded message: 738151741151741018151091542741151542941417798586047025106798046542030091151542586106798213151798025151798047542941106091542417046151091046542030046542047054151225542303106798840047542054030213151542047106542047225753151542025798542030542417046151091798030741151542030798303542753030046046584106091303542151030586054542047025741151542047054151225542417046151542030798542030753753380025586030047025106798542106798542030542753030091047025586417380030091542303151213025586151115542742106584151213151091273542046106741151047025741151046542047054025046542941417798586047025106798030380025047225542025046542018030303380225542303151046025084798151303542030798303542741030225542380151030213151542047054151542030753753380025586030047025106798542213417380798151091030018380151542047106542030047047030586433115542493106741151542030753753380025586030047025106798046542417046151542030542753151091046025046047151798047542586106106433025151542047054030047542586106798047030025798046542047054151542417046151091798030741151115542361054151798542047054151542030753753380025586030047025106798542091151586151025213151046542047054151542586106106433025151273542047054151542030753753380025586030047025106798542047091417046047046542047054151542586106106433025151542030798303542030417047054151798047025586030047151046542047054151542417046151091115542226054025046542018225753030046046151046542047054151542380106084025798542030798303542030380380106584046542030798542030047047030586433151091542798106047542106798380225542047106542084030025798542030586586151046046542047106542106798151542417046151091046542030586586106417798047542018417047542030380046106542047106542586054030798084151542047054151542417046151091798030741151542025798542047054151542586106106433025151542047106542030586586151046046542106047054151091542417046151091046542030586586106417798047046115542711047054151091542030753753380025586030047025106798046542741030225542417046151542030542586106106433025151542584025047054542030542046151046046025106798542009413115542361054151798542047054151542046151046046025106798542009413542025046542753091151046151798047151303542047106542047054151542030753753380025586030047025106798273542047054151542030753753380025586030047025106798542586054151586433046542941106091542047054151542417046151091542030046046106586025030047151303542584025047054542047054030047542009413542030798303542586091151030047151046542030798542030753753380025586030047025106798542046151046046025106798542941106091542047054030047542417046151091115542009941542047054151542046151046046025106798542009413840046542586030798542018151542151030046025380225542018151542753091151303025586047151303542047054151542030047047030586433151091542586030798542084106542047054091106417084054542030542380030091084151542798417741018151091542106941542046151046046025106798542009413840046542047106542941025798303542106798151046542030046046106586025030047151303542584025047054542417046151091046273542047106542084030025798542030586586151046046542047106542417046151091046542030586586106417798047046542584025047054106417047542030417047054151798047025586030047025106798115542009798542084151798151091030380273542586380025151798047482046025303151542091151741151741018151091542741151542941417798586047025106798046542046054106417380303542106798380225542091151741151741018151091542798106798482046151586091151047542025798941106091741030047025106798542046417586054542030046542047054151542417046151091798030741151115542226054151542417046151091542741030225542018151542030380380106584151303542047106542106753047482025798542047106542941417798586047025106798030380025047225542584054025586054542584025380380542091151741151741018151091542047054151542753030046046584106091303115542226054151542417046151091542046054106417380303542018151542584030091798151303542106941542047054151542091025046433046115542503380151030091047151260047542753030046046584106091303046542046054106417380303542798151213151091542018151542046047106091151303542106798542047054151542586380025151798047542030046542047054151225542030091151542213417380798151091030018380151542047106542018106047054542380106586030380542030798303542091151741106047151542030047047030586433046115542061030046046584106091303046542046054106417380303542018151542151798586091225753047151303542417046025798084542753417018380025586542433151225542151798586091225753047025106798273542584025047054542047054151542753091025213030047151542433151225542046047106091151303542106798542047054151542030753753380025586030047025106798542046151091213151091115

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome! Please select option to proceed.

1. Create Key Pairs

2. Encrypt Message

3. Decrypt Message

4. Display Keys

5. Exit

Option: 3

Please enter the message you wish to decrypt:

738151741151741018151091542741151542941417798586047025106798046542030091151542586106798213151798025151798047542941106091542417046151091046542030046542047054151225542303106798840047542054030213151542047106542047225753151542025798542030542417046151091798030741151542030798303542753030046046584106091303542151030586054542047025741151542047054151225542417046151542030798542030753753380025586030047025106798542106798542030542753030091047025586417380030091542303151213025586151115542742106584151213151091273542046106741151047025741151046542047054025046542941417798586047025106798030380025047225542025046542018030303380225542303151046025084798151303542030798303542741030225542380151030213151542047054151542030753753380025586030047025106798542213417380798151091030018380151542047106542030047047030586433115542493106741151542030753753380025586030047025106798046542417046151542030542753151091046025046047151798047542586106106433025151542047054030047542586106798047030025798046542047054151542417046151091798030741151115542361054151798542047054151542030753753380025586030047025106798542091151586151025213151046542047054151542586106106433025151273542047054151542030753753380025586030047025106798542047091417046047046542047054151542586106106433025151542030798303542030417047054151798047025586030047151046542047054151542417046151091115542226054025046542018225753030046046151046542047054151542380106084025798542030798303542030380380106584046542030798542030047047030586433151091542798106047542106798380225542047106542084030025798542030586586151046046542047106542106798151542417046151091046542030586586106417798047542018417047542030380046106542047106542586054030798084151542047054151542417046151091798030741151542025798542047054151542586106106433025151542047106542030586586151046046542106047054151091542417046151091046542030586586106417798047046115542711047054151091542030753753380025586030047025106798046542741030225542417046151542030542586106106433025151542584025047054542030542046151046046025106798542009413115542361054151798542047054151542046151046046025106798542009413542025046542753091151046151798047151303542047106542047054151542030753753380025586030047025106798273542047054151542030753753380025586030047025106798542586054151586433046542941106091542047054151542417046151091542030046046106586025030047151303542584025047054542047054030047542009413542030798303542586091151030047151046542030798542030753753380025586030047025106798542046151046046025106798542941106091542047054030047542417046151091115542009941542047054151542046151046046025106798542009413840046542586030798542018151542151030046025380225542018151542753091151303025586047151303542047054151542030047047030586433151091542586030798542084106542047054091106417084054542030542380030091084151542798417741018151091542106941542046151046046025106798542009413840046542047106542941025798303542106798151046542030046046106586025030047151303542584025047054542417046151091046273542047106542084030025798542030586586151046046542047106542417046151091046542030586586106417798047046542584025047054106417047542030417047054151798047025586030047025106798115542009798542084151798151091030380273542586380025151798047482046025303151542091151741151741018151091542741151542941417798586047025106798046542046054106417380303542106798380225542091151741151741018151091542798106798482046151586091151047542025798941106091741030047025106798542046417586054542030046542047054151542417046151091798030741151115542226054151542417046151091542741030225542018151542030380380106584151303542047106542106753047482025798542047106542941417798586047025106798030380025047225542584054025586054542584025380380542091151741151741018151091542047054151542753030046046584106091303115542226054151542417046151091542046054106417380303542018151542584030091798151303542106941542047054151542091025046433046115542503380151030091047151260047542753030046046584106091303046542046054106417380303542798151213151091542018151542046047106091151303542106798542047054151542586380025151798047542030046542047054151225542030091151542213417380798151091030018380151542047106542018106047054542380106586030380542030798303542091151741106047151542030047047030586433046115542061030046046584106091303046542046054106417380303542018151542151798586091225753047151303542417046025798084542753417018380025586542433151225542151798586091225753047025106798273542584025047054542047054151542753091025213030047151542433151225542046047106091151303542106798542047054151542030753753380025586030047025106798542046151091213151091115

Decoded message: Remember me functions are convenient for users as they don't have to type in a username and password each time they use an application on a particular device. However, sometimes this functionality is badly designed and may leave the application vulnerable to attack. Some applications use a persistent cookie that contains the username. When the application receives the cookie, the application trusts the cookie and authenticates the user. This bypasses the login and allows an attacker not only to gain access to one users account but also to change the username in the cookie to access other users accounts. Other applications may use a cookie with a session ID. When the session ID is presented to the application, the application checks for the user associated with that ID and creates an application session for that user. If the session ID's can be easily be predicted the attacker can go through a large number of session ID's to find ones associated with users, to gain access to users accounts without authentication. In general, client-side remember me functions should only remember non-secret information such as the username. The user may be allowed to opt-in to functionality which will remember the password. The user should be warned of the risks. Cleartext passwords should never be stored on the client as they are vulnerable to both local and remote attacks. Passwords should be encrypted using public key encryption, with the private key stored on the application server.

