CS12: Caesar Cipher Encryption

Understanding Cryptography Basics with C++

Mr. Gullo

Computer Science Department

March, 2025

Table of Contents

Table of Contents

Learning Objectives

By the end of this lesson, you will be able to:

- Understand the basic principles of cryptography
- Explain how the Caesar cipher works
- Implement Caesar cipher encryption and decryption in C++
- Calculate encryption and decryption keys
- Apply modular arithmetic in cryptographic algorithms

What is Cryptography?

- The practice of secure communication in the presence of adversaries
- From Greek: "kryptós" (hidden) and "graphein" (to write)
- Historically used for military and diplomatic communications
- Now essential for internet security, banking, and privacy

Key Terms

- **Plaintext**: Original, readable message
- **Encryption**: Process of converting plaintext to ciphertext
- **Ciphertext**: Encrypted, unreadable message
- **Decryption**: Process of converting ciphertext back to plaintext
- **Key**: Secret information used in encryption/decryption

Table of Contents

What is the Caesar Cipher?

- One of the earliest and simplest encryption techniques
- Named after Julius Caesar, who used it to communicate with his generals
- A type of **substitution cipher**
- Each letter in the plaintext is shifted a certain number of places down the alphabet
- Example: With a shift of 3, 'A' becomes 'D', 'B' becomes 'E', etc.

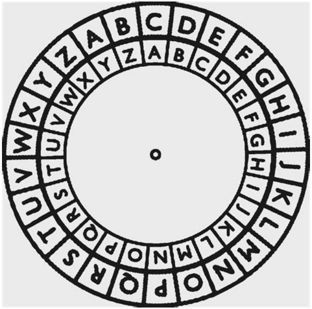
Historical Note

Caesar reportedly used a shift of 3 for all his communications, making it quite easy to break if you knew the system!

How Caesar Cipher Works

Encryption:

- Each letter is shifted forward by a fixed value (the key)
- If shift goes past 'z', it wraps around to 'a'



Example with key = 3:

- a $\rightarrow$ d
- b $\rightarrow$ e
- c $\rightarrow$ f
- z $\rightarrow$ c

Mathematical Representation

For a key K, each letter L becomes: $E(L) = (L + K) \mod 26$
Where letters are represented by their position in the alphabet (a=0, b=1, ..., z=25)

Decryption in Caesar Cipher

- Decryption is the reverse process of encryption
- Each letter is shifted backward by the same fixed value
- If shift goes past 'a', it wraps around to 'z'

Mathematical Representation

For a key K, decryption of letter C is: $D(C) = (C - K) \mod 26$
Alternatively, we can use a "decryption key": $D_K = 26 - K \mod 26$
Then decrypt using: $D(C) = (C + D_K) \mod 26$

Example: If encryption key is 3, decryption key is $26 - 3 = 23$

Table of Contents

caesarSingle.cpp - Overview

Let's examine the complete program:

```
char caesarShift(char message, char key){
    return 'a' + (message - 'a' + key - 'a') % 26;}

char getDecodeKey(char key){
    return 'a' + (26 - key + 'a') % 26;}
int main(){
    char plainText, key, secretMessage, decodeKey,
        decodedMessage;
    cout << "Plain text: ";
    cin >> plainText;
    cout << "Key: ";
    cin >> key;
    secretMessage = caesarShift(plainText, key);
    decodeKey = getDecodeKey(key);
    decodedMessage = caesarShift(secretMessage,
        decodeKey);
    cout << "Secret Message: " << secretMessage <<
        endl;
```

Program Output

When we run the program with inputs 'a' and 'x', we get:

```
Plain text: a
Key: x
Secret Message: x
Decode key: d
Decoded Message: a
```

- Input 'a' was encrypted to 'x' using key 'x'
- Decode key was calculated as 'd'
- 'x' was decrypted back to 'a' using the decode key 'd'
- The original message was successfully recovered!

Table of Contents

Limitations of Current Implementation

Our current implementation has several limitations:

- Only handles single characters, not strings
- Only works with lowercase letters
- Doesn't preserve spaces or punctuation
- Very easy to break (only 26 possible keys)

Security Considerations

The Caesar cipher is extremely weak by modern standards:

- Only 26 possible keys to try (brute force)
- Vulnerable to frequency analysis
- No protection against known-plaintext attacks

Table of Contents

Modern Cryptography vs. Caesar Cipher

Caesar Cipher

- Simple substitution
- Single fixed shift
- Very weak security
- Educational value

Modern Cryptography

- Complex mathematical algorithms
- Keys with billions of possibilities
- Asymmetric encryption
- Secure against current computational power

Historical Evolution

The Caesar cipher evolved into the Vigenère cipher, which led to more sophisticated polyalphabetic substitution methods, eventually giving way to modern cryptographic algorithms.

Applications of Cryptography Today

- **Secure Communications**
  - HTTPS for web browsing
  - End-to-end encrypted messaging (WhatsApp, Signal)
- **Data Protection**
  - Disk encryption
  - Password storage
- **Digital Signatures**
  - Document authentication
  - Software verification
- **Cryptocurrency**
  - Blockchain technology
  - Secure transactions

Table of Contents

Summary: Caesar Cipher

- The Caesar cipher is a simple substitution cipher
- Each letter is shifted by a fixed value (the key)
- Encryption: $E(x) = (x + k) \mod 26$
- Decryption: $D(x) = (x - k) \mod 26$ or $D(x) = (x + (26 - k)) \mod 26$
- In C++, we can implement this using character arithmetic
- While not secure for modern use, it introduces important cryptographic concepts

Key Concepts Learned

- Basic principles of encryption and decryption
- Character manipulation in C++
- Modular arithmetic
- Relationship between encryption and decryption keys