

Task-1 Implement Caesar Cipher

Report By

Leburi SriRam

Intern At

Prodigy Infotech

July 05,2024

Table of Contents:

Abstract

Introduction

Understanding Python,Encryption and Decryption

Algoritithm

Program

Conclusion

Reference

Abstract:

Creating a python program that can encrypt and decrypt text using Caesar Cipher Algorithm by allowing users to input a message and shift value to perform encryption and decryption.

Introduction:

As an intern at prodigy infotech, my task is clear: Python program to encrypt and decrypt text using Caesar cipher algorithm. This task provides me with unique opportunity to enhance my Programming skills, focusing on the encryption and decryption of text using Caesar cipher algorithm. Learn to code the Caesar cipher in Python and encrypt messages.

With the support of prodigy infotech and access to knowledge and resources necessary for the endeavor. Together, we embark on a journey to uncover the intricacies of encryption and decryption techniques, driven by a passion for cyber security, a commitment to ethical practices and an eagerness to learn.

Understanding Python,Encryption and Decryption

Python:

Python is a programming language that is interpreted, object-oriented, and considered to be high-level too. Python is one of the easiest yet most useful programming languages which is widely used in the software industry. People use Python for Competitive Programming, Web Development, and creating software. Due to its easiest syntax, it is recommended for beginners who are new to the software engineering field. Its demand is growing at a very rapid pace due to its vast use cases in Modern Technological fields like Data Science, Machine learning, and Automation Tasks. For many years now, it has been ranked among the top Programming languages.

Python is a set of instructions that we give in the form of a Program to our computer to perform any specific task. It is a Programming language having properties like it is interpreted, object-oriented and it is high-level too. Due to its beginner-friendly syntax, it became a clear choice for beginners to start their programming journey. The major focus behind creating it is making it easier for developers to read and understand, also reducing the lines of code.

Advantages of Python

Easy to learn, read, and understand

Versatile and open-source

Interpreted Language

Supports libraries

Huge library

Strong community

Disadvantages of Python

Restrictions in design

Memory inefficient

Weak mobile computing

Runtime errors

Slow execution speed

Encryption :

Encryption is a form of data security in which information is converted to ciphertext. Only authorized people who have the key can decipher the code and access the original plaintext information.

In even simpler terms, encryption is a way to render data unreadable to an unauthorized party. This serves to thwart cybercriminals, who may have used quite sophisticated means to gain access to a corporate network—only to find out that the data is unreadable and therefore useless.

Encryption not only ensures the confidentiality of data or messages but it also provides authentication and integrity, proving that the underlying data or messages have not been altered in any way from their original state.

An encryption algorithm is a mathematical formula used to transform plaintext (data) into ciphertext. An algorithm will use the key to alter the data in a predictable way. Even though the encrypted data appears to be random, it can actually be turned back into plaintext by using the key again. Some commonly used encryption algorithms include Blowfish, Advanced Encryption Standard (AES), Rivest Cipher 4 (RC4), RC5, RC6, Data Encryption Standard (DES), and Twofish.

Types of Encryption:

Symmetric: Only one secret key is used to both cipher and decipher information. symmetric encryption algorithms include AES-128, AES-192, and AES-256.

Asymmetric: Asymmetric encryption is a relatively new method that uses two different but related keys to encrypt and decrypt data. One key is secret and one key is public. The public key is used to encrypt data, and the private key is used to decrypt (and vice versa).

Data encryption Standard: DES works by using the same key to encrypt and decrypt a message, so both the sender and the receiver must have access to the same private key. DES has been superseded by the more secure AES algorithm.

Triple data encryption standard: The Triple Data Encryption Standard involved running the DES algorithm three times, with three separate keys. 3DES was largely seen as a stopgap measure, as the single DES algorithm was increasingly becoming seen as too weak to stand up to brute force attacks and the stronger AES was still under evaluation

RSA: Rivest-Shamir-Adleman (RSA) is an algorithm and the basis of a cryptosystem—a suite of cryptographic algorithms used for specific security services or purposes. This enables public key encryption and is often used by browsers to connect to websites and by virtual private networks (VPNs). RSA is asymmetric, in which two different keys are used for encryption: one public and one private. If decryption is carried out with the public key, encryption is performed with the private key, or vice versa.

Advanced encryption standard: Developed in 1997 by the National Institute of Standards and Technology (NIST) as an alternative to the Data Encryption Standard, the Advanced Encryption Standard is a cipher chosen by the U.S. government to protect sensitive information. AES has three different key lengths to encrypt and decrypt a block of messages: 128-bit, 192-bit, and 256-bit. AES is widely used for protecting data at rest in such applications as databases and hard drives.

Encryption in the cloud: Cloud encryption is a service offered by cloud storage providers in which data is first encrypted using algorithms before being pushed to a storage cloud. They rely on a Bring Your Own Encryption (BYOE) model in which they use their own encryption software and manage their own encryption keys to ensure a level of cloud computing security they are comfortable with.

End to end encryption: End-to-end encryption (E2EE) ensures that only the two users communicating with one another can read the messages. Even the intermediary, such as the telecom or internet service provider, cannot decrypt the messages. E2EE is generally seen as the most secure way to communicate privately and securely online. Examples of E2EE in use include the WhatsApp messaging service, which famously asserts that users' messages are secured with "locks."

Decryption:

Decryption is the process of taking an encrypted message and restoring it to its original format. Once completed, it can be read again by anyone. It's like solving a puzzle, where the right combination of codes or passwords is used to reveal the original message.

Decryption Techniques and Algorithms:

Symmetric Decryption

Symmetric decryption is like a secret handshake between two people. It uses the same key for both encrypting and decrypting data, ensuring that only those with the key can access the information. Popular symmetric algorithms include AES, DES, and 3DES, which are used to secure private conversations and other sensitive data.

Asymmetric Decryption

Asymmetric decryption takes a different approach, using two keys instead of one. Imagine a locked mailbox where anyone can drop mail in, but only the owner can unlock and retrieve it. This is how asymmetric decryption works – it uses a public key for encryption and a private key for decryption.

Public-Key Cryptography

Public-key cryptography is the backbone of secure communication and data protection in the digital world. It involves the use of a pair of keys – a public key and a private key – to encrypt and decrypt messages. Anyone can encrypt a message using the recipient's public key, but only the recipient who possesses the corresponding private key can decrypt it.

Advanced Encryption Standard (AES)

The Advanced Encryption Standard (AES) is a widely-used encryption standard that offers robust security and performance. It supports key lengths of 128-bit, 192-bit, and 256-bit, providing varying levels of security depending on the needs of the user. AES is resistant to all known attacks except brute force, making it a trusted choice for securing sensitive data.

RSA Algorithm

The RSA algorithm is a popular public-key encryption and decryption method that relies on the mathematical complexity of factoring large numbers. With RSA, a user generates a pair of keys – a public key for encryption and a private key for decryption. The user can share the public key with anyone, allowing them to encrypt messages that can only be decrypted by the user with the corresponding private key.

Triple Data Encryption Standard (Triple DES)

Triple Data Encryption Standard (Triple DES), or 3DES, is a symmetric-key block cipher that builds upon the original DES algorithm, providing enhanced security through three rounds of encryption. The encryption process consists of three steps: Encrypt-Decrypt-Encrypt (EDE), using three 56-bit keys (K1, K2, and K3) as a key bundle to encrypt plaintext

Decryption Process and Tools

Manual Decryption

Manual decryption involves deciphering encrypted data using codes or keys without the help of automated tools or software. This method can be quite time-consuming and complex, as it requires a deep understanding of encryption algorithms and key management.

Automated Decryption

Automated decryption utilizes software tools and programs to decrypt data without manual intervention, making the process faster and more efficient. These tools can decode multiple base encodings, classical ciphers, hashes, or even more complex cryptography, depending on the specific requirements of the user. Some popular tools for decrypting data include password managers, encryption software, and decryption software.

Encryption strategies:

- Privacy and security
- Regulations
- Secure internet browsing

Caesar Cipher in Cryptography:

The Caesar cipher is a simple encryption technique that was used by Julius Caesar to send secret messages to his allies. It works by shifting the letters in the plaintext message by a certain number of positions, known as the “shift” or “key”. For example, with a shift of three, the letter ‘A’ becomes ‘D’, ‘B’ becomes ‘E’, and so on.

For example, if the shift is 3, then the letter A would be replaced by the letter D, B would become E, C would become F, and so on. The alphabet is wrapped around so that after Z, it starts back at A.

Here is an example of how to use the Caesar cipher to encrypt the message “PRODIGY” with a shift of 3:

- Write down the plaintext message: PRODIGY
- Choose a shift value. In this case, we will use a shift of 3

Replace each letter in the plaintext message with the letter that is three positions to the right in the alphabet.

P becomes S (shift 3 from P)

R becomes U (shift 3 from R)

O becomes R (shift 3 from O)

D becomes G (shift 3 from D)

I becomes L(shift 3 from I)

G becomes J (shift 3 from G)

Y becomes B (shift 3 from Y)

- The encrypted message is now “SURGLJB”. $E_{\{n\}}(x) = (x+n) \bmod \{26\}$
To decrypt the message, $D_{\{n\}}(x) = (x-n) \bmod \{26\}$ you simply need to shift each letter back by the same number of positions. In this case, you would shift each letter in “SURGLJB” back by 3 positions to get the original message, “PRODIGY”.

Features of Caesar Cipher

- ✓ Substitution cipher: The Caesar cipher is a type of substitution cipher, where each letter in the plaintext is replaced by a letter some fixed number of positions down the alphabet.
- ✓ Fixed key: The Caesar cipher uses a fixed key, which is the number of positions by which the letters are shifted. This key is known to both the sender and the receiver.
- ✓ Symmetric encryption: The Caesar cipher is a symmetric encryption technique, meaning that the same key is used for both encryption and decryption.
- ✓ Limited keyspace: The Caesar cipher has a very limited keyspace of only 26 possible keys, as there are only 26 letters in the English alphabet.
- ✓ Vulnerable to brute force attacks: The Caesar cipher is vulnerable to brute force attacks, as there are only 26 possible keys to try.
- ✓ Easy to implement: The Caesar cipher is very easy to implement and requires only simple arithmetic operations, making it a popular choice for simple encryption tasks.

Rules for the Caesar Cipher

- ❖ Choose a number between 1 and 25. This will be your “shift” value.
- ❖ Write down the letters of the alphabet in order, from A to Z.
- ❖ Shift each letter of the alphabet by the “shift” value. For example, if the shift value is 3, A would become D, B would become E, C would become F, and so on.
- ❖ Encrypt your message by replacing each letter with the corresponding shifted letter. For example, if the shift value is 3, the word “hello” would become “SURGLJB”.
- ❖ To decrypt the message, simply reverse the process by shifting each letter back by the same amount. For example, if the shift value is 3, the encrypted message “SURGLJB” would become “PRODIGY”.

Algorithm for Caesar Cipher

Input:

Choose a shift value between 1 and 25.

Write down the alphabet in order from A to Z.

Create a new alphabet by shifting each letter of the original alphabet by the shift value. For example, if the shift value is 3, the new alphabet would be:

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

Replace each letter of the message with the corresponding letter from the new alphabet. For example, if the shift value is 3, the word “PRODIGY” would become “SURGLJB”.

To decrypt the message, shift each letter back by the same amount. For example, if the shift value is 3, the encrypted message “SURGLJB” would become “PRODIGY”.

Python Program:

Encrypt:

#A python program to illustrate Caesar Cipher Technique

```
def encrypt(text,s):
    result = ""

    # traverse text
    for i in range(len(text)):
        char = text[i]

        # Encrypt uppercase characters
        if (char.isupper()):
            result += chr((ord(char) + s-65) % 26 + 65)

        # Encrypt lowercase characters
        else:
            result += chr((ord(char) + s - 97) % 26 + 97)

    return result

#check the above function
text = input("Enter text: ")
s = int(input("Enter shift value: "))
print ("Text : " + text)
print ("Shift : " + str(s))
print ("Cipher: " + encrypt(text,s))
```

Output:

Enter text: PRODIGY

Enter shift value: 3

Text : PRODIGY

Shift : 3

Cipher: SURGLJB

Decrypt:

#A python program to illustrate Caesar Cipher Technique

```
def decrypt(cipher,s):  
    result = ""  
  
    # traverse text  
    for i in range(len(cipher)):  
        char = cipher[i]  
  
        # decrypt uppercase characters  
        if (char.isupper()):  
            result += chr((ord(char) - s-65) % 26 + 65)  
  
        # decrypt lowercase characters  
        else:  
            result += chr((ord(char) - s - 97) % 26 + 97)  
  
    return result  
  
#check the above function  
cipher = input("Enter cipher: ")  
s = int(input("Enter shift value: "))  
print ("Cipher : " + cipher)  
print ("Shift : " + str(s))  
print ("Text: " + decrypt(cipher,s))
```

Output:

Enter cipher: SURGLJB

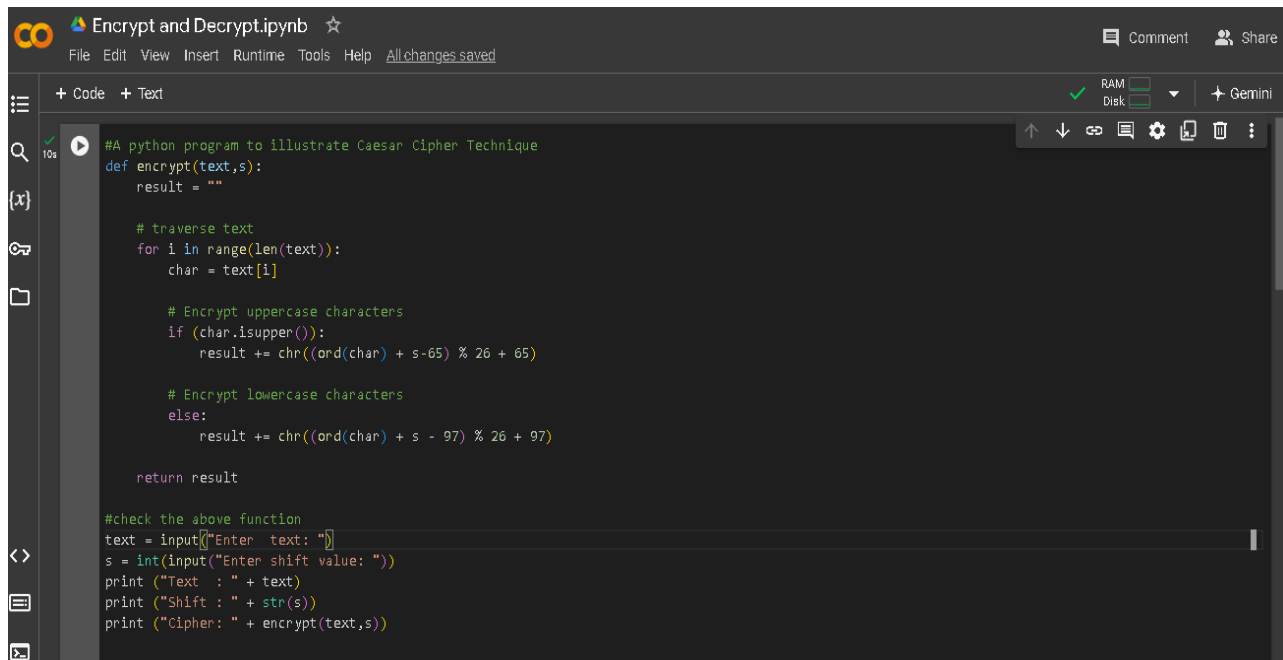
Enter shift value: 3

Cipher : SURGLJB

Shift : 3

Text: PRODIGY

Screen shots:



The screenshot shows a Jupyter Notebook titled "Encrypt and Decrypt.ipynb". The code defines an `encrypt` function that takes `text` and `s` as arguments. It traverses the text, encrypting uppercase and lowercase characters using a Caesar cipher. The code also includes a check for the function by taking user input for text and shift value.

```
#A python program to illustrate Caesar Cipher Technique
def encrypt(text,s):
    result = ""

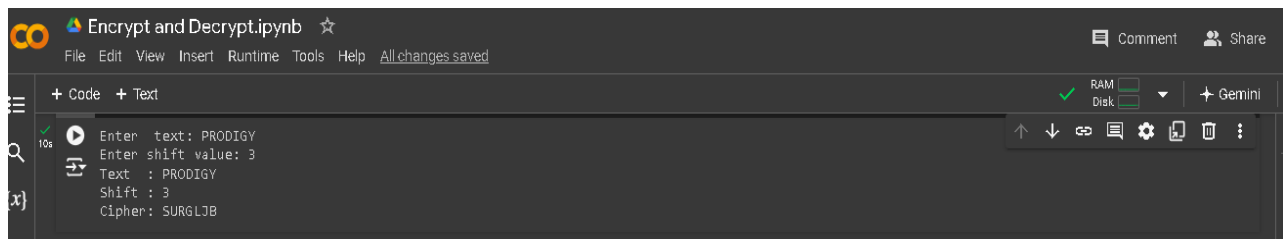
    # traverse text
    for i in range(len(text)):
        char = text[i]

        # Encrypt uppercase characters
        if (char.isupper()):
            result += chr((ord(char) + s-65) % 26 + 65)

        # Encrypt lowercase characters
        else:
            result += chr((ord(char) + s - 97) % 26 + 97)

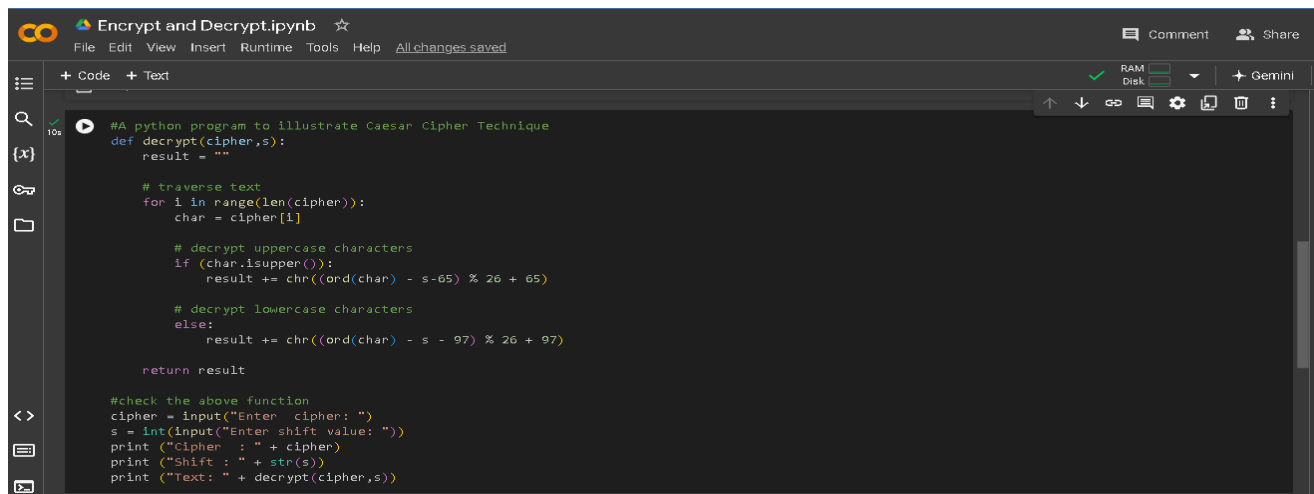
    return result

#check the above function
text = input("Enter text: ")
s = int(input("Enter shift value: "))
print ("Text : " + text)
print ("Shift : " + str(s))
print ("Cipher: " + encrypt(text,s))
```



The screenshot shows the output of the encryption function. The user entered "PRODIGY" as the text and "3" as the shift value. The output displays the text, shift, and the resulting cipher "SURGLJB".

```
Enter text: PRODIGY
Enter shift value: 3
Text : PRODIGY
Shift : 3
Cipher: SURGLJB
```



The screenshot shows a Jupyter Notebook titled "Encrypt and Decrypt.ipynb". The code defines a `decrypt` function that takes `cipher` and `s` as arguments. It traverses the cipher, decrypting uppercase and lowercase characters using a Caesar cipher. The code also includes a check for the function by taking user input for cipher and shift value.

```
#A python program to illustrate Caesar Cipher Technique
def decrypt(cipher,s):
    result = ""

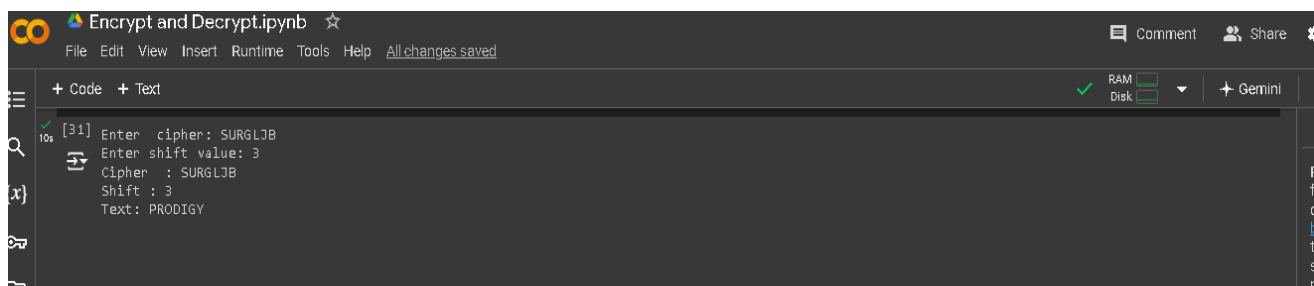
    # traverse text
    for i in range(len(cipher)):
        char = cipher[i]

        # decrypt uppercase characters
        if (char.isupper()):
            result += chr((ord(char) - s-65) % 26 + 65)

        # decrypt lowercase characters
        else:
            result += chr((ord(char) - s - 97) % 26 + 97)

    return result

#check the above function
cipher = input("Enter cipher: ")
s = int(input("Enter shift value: "))
print ("Cipher : " + cipher)
print ("Shift : " + str(s))
print ("Text: " + decrypt(cipher,s))
```



The screenshot shows the output of the decryption function. The user entered "SURGLJB" as the cipher and "3" as the shift value. The output displays the cipher, shift, and the resulting text "PRODIGY".

```
[31] Enter cipher: SURGLJB
Enter shift value: 3
Cipher : SURGLJB
Shift : 3
Text: PRODIGY
```

Conclusion:

This comprehensive guide has outlined the python code that can encrypt and decrypt text using Caesar Cipher Algorithm by allowing users to input a message and shift value to perform encryption and decryption, Caesar cipher algorithm in Cryptography and complete detail about Encryption, Decryption and their Algorithms.

Data encryption is a critical and foundational component of data security and privacy in today's digital age. It helps protect sensitive information from unauthorized access, theft and other security threats. Encrypting data ensures that even if it falls into the wrong hands, it cannot be easily read or understood. Organizations should implement encryption to counter the ever-growing threat of data breaches

During this task allotted by Prodigy Infotech. I have gained valuable insights into importance of encryption, decryption techniques and algorithms in securing data. The support and resources provided by Prodigy infotech have been instrumental in enhancing my understanding and skills in Cyber security.

Reference:

<https://www.geeksforgeeks.org/what-is-python/>

<https://www.fortinet.com/resources/cyberglossary/encryption>

<https://softwarelab.org/blog/what-is-decryption/>

<https://www.geeksforgeeks.org/caesar-cipher-in-cryptography/>

<https://www.titanfile.com/blog/what-is-data-encryption-and-why-is-it-important/>