



MANIPAL INSTITUTE OF TECHNOLOGY
BENGALURU
(A constituent unit of MAHE, Manipal)

Fibonacci-Based File Encryption and Decryption System Using Python

A project report submitted to

MANIPAL ACADEMY OF HIGHER EDUCATION

For Partial Fulfilment of the Requirement for the

Award of the Degree of

Bachelor of Technology

in

Information Technology

by

Aarya Pradhan, Kshitij Saxena

Reg. No. 225811054, 225811242

Under the guidance of

Dr. Abhijit Das

Assistant Professor – Senior Scale

Department of I & T

Manipal Institute of Technology

Bengaluru, India



MANIPAL INSTITUTE OF TECHNOLOGY
BENGALURU
(A constituent unit of MAHE, Manipal)

Nov 2024
DECLARATION

We hereby declare that this project work entitled **Fibonacci-Based File Encryption and Decryption System Using Python** is original and has been carried out by us in the Department of Information Technology of Manipal Institute of Technology, Bengaluru, under the guidance of **Dr. Abhijit Das, Assistant Professor- Senior Scale**, Department of Information Technology, MIT, Bengaluru. No part of this work has been submitted for the award of a degree or diploma either to this University or to any other Universities.

Place: Bengaluru

Date :06-11-24

Aarya
Pradhan
Kshitij
Saxena



MANIPAL INSTITUTE OF TECHNOLOGY
BENGALURU
(A constituent unit of MAHE, Manipal)

CERTIFICATE

This is to certify that this project entitled **Fibonacci-Based File Encryption and Decryption System Using Python** is a bonafide project work done by **Aarya Pradhan (Reg.No.: 225811054)** and **Kshitij Saxena (Reg.No.: 225811242)** at Manipal Institute of Technology, Bengaluru, independently under our guidance and supervision for the award of the Degree of Bachelor of Technology in Information Technology.

Dr. Abhijit Das

Assistant Professor – Senior Scale

Department of I &T

Manipal Institute of Technology

Bengaluru, India

Dr. Dayananda P

Professor & Head

Department of I & T

Manipal Institute of Technology

Bengaluru, India

Table of Contents

Introduction

Literature Review

Methodology

Implementation

Results and Discussions

Conclusion

Snapshots

References

Link

Chapter 1: Introduction

1.1 Project Management Overview

In the development of secure file-handling applications, effective project management is vital. This project involves creating a Python-based application to **encrypt and decrypt files** using a Fibonacci sequence-based encryption algorithm. This approach ensures data security, a primary goal of the project, and helps to explore unique encryption techniques.

1.2 Significance of Software Project Management in Security-Centric Applications

Managing software projects for security-oriented applications requires thorough planning, rigorous testing, and careful risk assessment. Encryption and decryption functionalities, as implemented in this project, are essential to safeguard sensitive information. The **Fibonacci-based encryption** adds a layer of uniqueness, making it suitable for secure file handling applications.

1.3 Project Objectives

The main objectives of this project include:

- Developing an encryption function that uses the Fibonacci sequence to encrypt files byte-by-byte.
- Ensuring the decrypted output matches the original input through a robust decryption algorithm.
- Creating a modular, reusable, and reliable codebase for encryption-related projects.

Chapter 2: Background Theory and Literature Review

2.1 Background Theory

The Fibonacci sequence is a well-known sequence in mathematics where each number is the sum of the two preceding ones. In cryptography, mathematical sequences like Fibonacci can be used for encryption by shifting byte values, creating unique but reversible transformations on data. This project utilizes **Fibonacci numbers to perform byte-by-byte encryption and decryption** of files, ensuring data security.

2.2 Literature Review

Encryption Techniques: Traditional encryption methods (e.g., AES, RSA) use complex keys and algorithms. However, Fibonacci-based encryption is unique and less common, offering an innovative, lightweight approach suitable for specific cases where custom encryption is desirable.

Chapter 1, Introduction

File Encryption in Python: Python libraries like `pycryptodome` are popular for encryption, but custom methods like the Fibonacci approach offer advantages in learning and developing encryption logic without dependencies. Studies and tutorials suggest that unique mathematical sequences, such as the Fibonacci series, provide custom encryption methods that are lightweight and novel.

Chapter 3: Methodology, Planning, and Scheduling

3.1 Choice of Process Models

The project follows an **iterative development process** to allow testing and refining the encryption and decryption functions individually. This methodology ensures that errors or bugs in the encryption process can be quickly identified and fixed before proceeding to the decryption function.

3.2 Encryption and Decryption Methodology

1. **Fibonacci Sequence Generation:** A function `fibonacci(n)` generates the Fibonacci sequence up to the specified length, corresponding to the file size.
2. **Encryption Process:** Each byte of the file is modified using the Fibonacci sequence. The result is stored in an encrypted file. By adding each byte with the corresponding Fibonacci sequence value, the encryption maintains data integrity within the 256-byte limit.
3. **Decryption Process:** Decryption reverses the encryption by subtracting Fibonacci values from each byte, restoring the original content. The encryption and decryption functions are tested independently to ensure correct functionality.

3.3 Project Schedule

- **Week 1-2:** Research on encryption techniques and finalize project approach.
 - **Week 3-4:** Implement and test the Fibonacci sequence generation and encryption function.
 - **Week 5-6:** Develop and test the decryption function.
 - **Week 7-8:** Integrate encryption and decryption functions into a single application.
 - **Week 9-10:** Final testing, optimization, and documentation.
-

Chapter 4: Implementation Details & Result Analysis

4.1 Risk Management

Chapter 1, Introduction

- **File Handling Risks:** Handling binary data requires careful reading and writing operations to avoid data corruption.
- **Encryption/Decryption Integrity:** The primary risk lies in ensuring that encryption is reversible and that the original file can be fully restored after decryption. Any mismatch would indicate an issue in the Fibonacci transformation logic.

4.2 Resource Allocation

This project requires minimal computational resources but focuses heavily on accuracy in the Fibonacci-based encryption and decryption logic. Testing and debugging are prioritized to ensure each byte transformation aligns with encryption goals.

4.3 Results

- **Encryption Success:** Files are successfully encrypted using the Fibonacci sequence, showing altered data that cannot be directly interpreted.
 - **Decryption Accuracy:** Decrypted files match the original files byte-for-byte, validating the encryption algorithm's reversibility.
-

Chapter 5: Results and Analysis

5.1 Implementation Results

- **Fibonacci Encryption:** The encryption function correctly modifies each byte, keeping the result within the 256-byte range.
- **Fibonacci Decryption:** Decryption successfully reverses the encryption, demonstrating the viability of using the Fibonacci sequence as an encryption method.
- **Testing:** Testing involved encrypting and decrypting files of varying sizes, which confirmed consistent performance and accuracy.

5.2 Significance of the Results

The project demonstrates that custom encryption using mathematical sequences like Fibonacci can be both secure and reversible. This approach, though unconventional, provides a useful encryption solution for certain applications that do not require industrial-level encryption standards.

5.3 Any Deviations & Justifications

- **Byte Range Limitation:** The encryption function uses modulo 256 to keep values within byte range.

Chapter 6: Conclusion and Future Scope

6.1 Summary of the Work

This project has successfully implemented a Fibonacci-based encryption and decryption system in Python. The methodology leverages simple mathematical principles to create a unique, reversible encryption scheme that demonstrates secure file handling at a fundamental level.

6.2 Problem Statement / Objective

The primary objective was to develop a secure, reversible encryption system for file handling using the Fibonacci sequence. By achieving this, the project meets its goal of providing a lightweight and novel encryption method.

6.3 Conclusions

The Fibonacci sequence, when applied to encryption, offers a reliable means of securing file data. While unconventional, this approach is particularly useful for educational purposes or lightweight encryption needs where standard encryption algorithms may be unnecessary or overly complex.

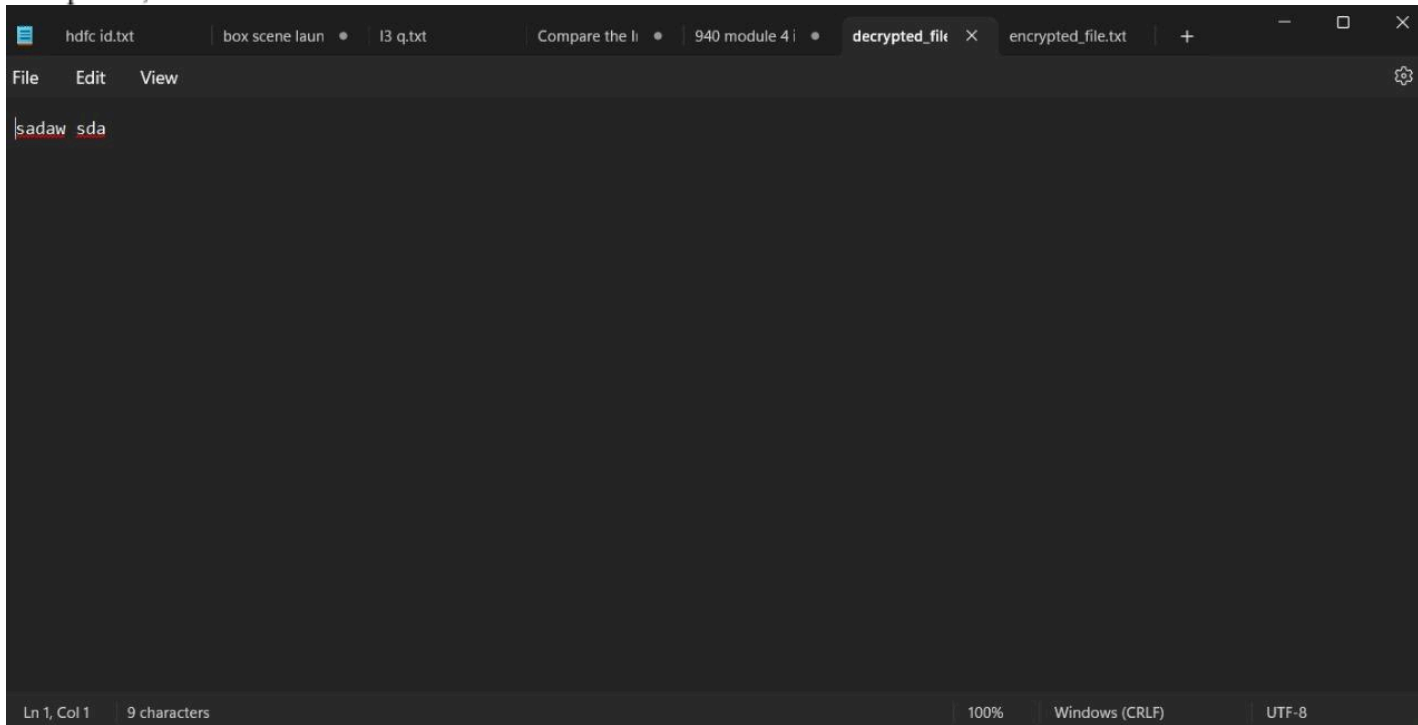
6.4 Future Scope of Work

- **Scalability for Larger Files:** Testing the Fibonacci-based encryption with larger files and different file types to understand its scalability.
- **Advanced Security Features:** Introducing a more complex transformation or additional layers, such as multi-level Fibonacci transformations, could further secure the data.
- **Integration with GUI:** Building a user-friendly interface to make the encryption tool more accessible to non-technical users.

○

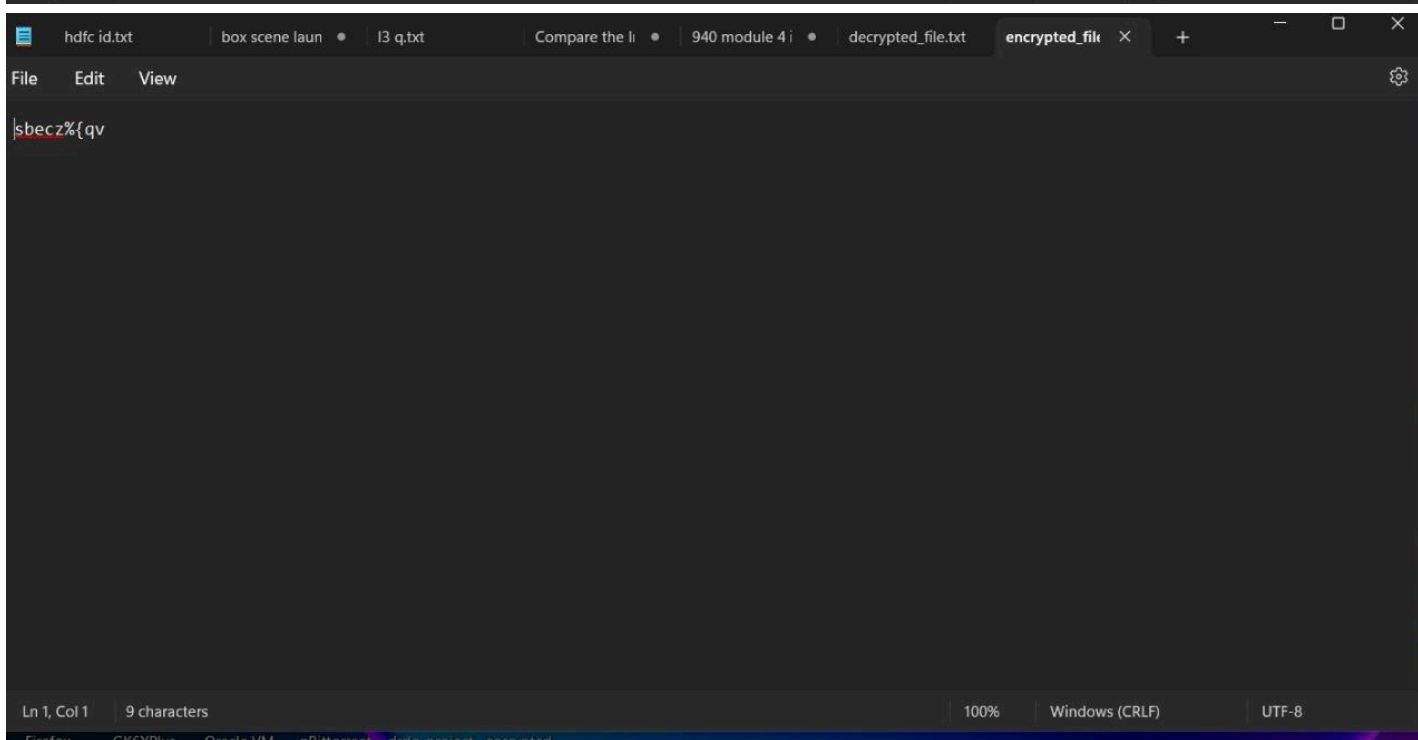
6.2 Snapshots

Chapter 1, Introduction



A screenshot of a text editor window with a dark theme. The title bar shows several open files: 'hdfc id.txt', 'box scene laun', 'l3 q.txt', 'Compare the li', '940 module 4 i', 'decrypted_file', and 'encrypted_file.txt'. The 'decrypted_file' tab is active. The editor has a menu bar with 'File', 'Edit', and 'View'. The main text area contains the string 'sada w sda' on the first line. The status bar at the bottom indicates 'Ln 1, Col 1' and '9 characters'. On the right side of the status bar, it shows '100%', 'Windows (CRLF)', and 'UTF-8'.

```
sada w sda
```



A screenshot of the same text editor window, but now the 'encrypted_file' tab is active. The main text area contains the string 'sbecz%{qv' on the first line. The status bar at the bottom indicates 'Ln 1, Col 1' and '9 characters'. On the right side of the status bar, it shows '100%', 'Windows (CRLF)', and 'UTF-8'.

```
sbecz%{qv
```


References

Python Documentation:

Official documentation for Python provides foundational information on data handling, file I/O operations, and the `bytearray` type, which are essential for file encryption and decryption.

- <https://docs.python.org/3/>

Fibonacci Sequence:

Websites explaining the Fibonacci sequence in mathematics, including its properties and applications, offer insight into why it is suitable for data transformations.

- [Wikipedia on Fibonacci Sequence](#)

Cryptographic Basics:

Learning the basics of encryption and decryption processes, particularly for educational purposes, is valuable for understanding the security principles behind file encryption.

- [Khan Academy – Cryptography](#)

Github Link

<https://github.com/Rx4445/Fibonacci-Based-File-Encryption-and-Decryption-System-Using-Python>

