

# INDIAN INSTITUTE OF INFORMATION TECHNOLOGY DHARWAD

# Cryptography project CS352

**Dr. Rajendra Hegadi**, Indian Institute of Information Technology Dharwad

**TOPIC:** RSA Cipher Machine: Implementation of RSA Algorithm in Python

# GROUP MEMBERS

1. M Karthikeya(22BCS073)
2. Jagruth Dangi(22BCS048)
3. Chirag Bhagel (22BCS031)
4. Sharath Medijala(22BCS069)

**Aim of the Project**

The aim of this project is to design and implement a Python-based RSA encryption system that demonstrates the practical application of secure communication techniques. The project seeks to provide users with a clear understanding of how public-key cryptography works, focusing on key generation, message encryption, and decryption processes. Additionally, the project introduces custom ASCII encoding to handle extended character sets and offers an interactive environment to facilitate hands-on learning and experimentation with encryption concepts. By simplifying complex cryptographic operations, this project bridges the gap between theory and practice. It empowers users to experiment with encryption securely and efficiently while enhancing their understanding of digital security principles. Moreover, the implementation focuses on creating a reliable and user-friendly interface for educational and practical purposes. Through this project, users can grasp the importance of RSA in securing modern communication systems.

**Tools and Technologies Used**

**Python Programming Language**: The core language used for implementing the RSA encryption system, leveraging its rich libraries and versatility.

**Prime Number Generator**: A custom script developed to generate large prime numbers essential for secure RSA key generation.

**Mathematical Functions**: Modular arithmetic and exponentiation techniques for encryption and decryption processes.

**Custom ASCII Encoder**: A utility for converting extended characters into secure numeric formats for encryption.

**Interactive Shell**: A Python-based shell interface allowing users to input messages, generate keys, and perform encryption or decryption in real time.

**IDE or Text Editor**: For coding and debugging the scripts, such as VS Code, PyCharm, or any preferred development environment.

**Introduction**

The RSA algorithm is a cornerstone of modern cryptography, known for its reliance on the computational challenge of factoring large numbers. This project aims to simplify RSA concepts for learners and enthusiasts by providing a Python-based implementation. It incorporates every step of RSA encryption, from generating secure keys to encoding and decoding messages. The addition of an interactive shell ensures that users can experiment with encryption hands-on, enhancing their understanding of how RSA secures digital communication.

**Methodology**

* **Prime Number Generation:**

Prime numbers are critical for RSA key generation. The project includes a dedicated script that efficiently generates large primes. These primes are used to calculate the RSA modulus \(N\), which forms the basis of the encryption and decryption process.

### RSA Key Pair Generation

RSA key pairs are generated by selecting two large primes, calculating (N = p \* q) and (phi(N) = (p-1) \* (q-1)), and deriving the public key exponent (e) and private key exponent (d) using modular arithmetic for secure encryption and decryption.

* **Character Encoding:**

The project implements custom ASCII encoding to support a wide range of characters, including special symbols. This encoding ensures that encrypted messages remain secure and readable during the decryption process.

* **String to Integer Conversion:**

To perform RSA encryption, text messages are first converted into integer blocks. This is achieved through a process that maps characters to numeric values, enabling the RSA algorithm to work seamlessly with string data.

* **Modular Exponentiation:**

Modular exponentiation is central to the RSA process, allowing efficient calculation of large power moduli. This ensures the encryption and decryption processes are computationally efficient and secure.

* **Interactive Shell**

An interactive shell enables users to experiment with RSA encryption and decryption in real time. It provides feedback on key generation, block sizes, and encoding mechanisms, offering a hands-on approach to learning RSA principles.

**Conclusion**

The RSA Cipher Machine demonstrates the implementation of a secure cryptographic system in Python. Through its interactive shell and comprehensive features, the project simplifies complex encryption concepts while maintaining robust functionality. By combining secure key generation, modular arithmetic, and character encoding, it offers a practical and educational insight into the RSA algorithm.This project not only provides a working encryption tool but also underscores the significance of RSA in securing digital communication, making it an invaluable resource for learning and experimentation.

**THANK YOU**