# PRIME NUMBERS AND CRYPTOSYSTEMS

PROJECT PLAN

FAWAZ HAKEEM-JIMOH

FINAL YEAR PROJECT

SUPERVISED BY: LUO, ZHIYUAN

# ABSTRACT

This project aims to explore the relationship between prime numbers and cryptographic systems by implementing fundamental routines for prime number handling and developing cryptographic applications using these principles. The primary focus will be designing and implementing cryptographic protocols, such as RSA (Rivest-Shamir-Adleman) and the Diffie-Hellman key exchange, which rely heavily on number theory, specifically the intractability of certain problems involving prime numbers. These algorithms are critical in ensuring secure communication over public and insecure channels, making them essential for applications such as bank networks, secure communications, and data encryption.

The initial phase of the project will focus on building basic cryptographic tools, such as prime number generation, RSA encryption/decryption routines, and key generation using standard data types in programming languages like Java or C++. In this phase, a proof-of-concept program will demonstrate RSA encryption and decryption of numbers and generate keys. A detailed report will accompany these deliverables, describing the RSA process and showcasing examples checked by the implemented program.

The final phase will focus on developing a full-fledged cryptographic application using an object-oriented design approach, incorporating a complete implementation life cycle that follows modern software engineering principles. The program will be capable of encrypting and decrypting messages and files using integers of arbitrary size, with a graphical user interface (GUI) to facilitate secure communication tasks, such as secure chat or file transfers. The project will also integrate RSA with a symmetric encryption scheme, where RSA will distribute the keys for the symmetric encryption.

The final report will offer a comprehensive overview of modern cryptography, focusing on RSA and relevant number theory issues, such as primality checking and factorization. Additionally, it will cover the implementation challenges (e.g., choice of data structures, and numerical methods) and provide insights into the software engineering process followed during the project. Performance data on the running time of the cryptographic programs will also be presented.

Project Objectives:

1. Develop a deep understanding of prime numbers and their role in cryptographic systems.

2. Implement RSA and Diffie-Hellman protocols using prime numbers.

3. Design proof-of-concept programs for RSA encryption/decryption and key generation.

4. Build a full object-oriented cryptographic application, complete with encryption, decryption, and key distribution.

5. Implement a GUI for secure communication (e.g., secure chat or file transfer).

6. Combine RSA with symmetric encryption for hybrid cryptographic protocols.

7. Analyze the performance and security of the cryptographic implementations.

8. Produce a comprehensive report covering cryptography theory, implementation challenges, and software engineering processes.

The project will serve as an exploration of how prime numbers, number theory, and modern software engineering converge to create secure cryptographic solutions.

# Timeline

Term 1

Week 4:

- Study the foundations of number theory and prime numbers.
- Review cryptographic protocols, specifically RSA and Diffie-Hellman.

Week 5- 6:

- Implement basic routines for prime number generation and primality testing.
- Study the mathematics behind RSA encryption and decryption.

Week 7-8:

- Develop proof-of-concept programs for RSA encryption/decryption using standard data types (Java/C++).
- Implement key generation for RSA.

Week 9:

- Analyze and test the RSA proof-of-concept program with manually worked examples.
- Begin working on a report describing the RSA process and results.

Week 10 :

- Research symmetric encryption schemes and their combination with RSA.
- Evaluate initial program performance and security.

Week 11:

- Prepare the interim report detailing RSA, key generation, and early deliverables.
- Present proof-of-concept results for interim evaluation.

Term 2

Week 1-2:

- Extend the prime number generation algorithm to handle arbitrary-sized integers.
- Refine RSA implementation to encrypt and decrypt text messages.

Week 3:

- Implement a basic Graphical User Interface (GUI) for encryption/decryption tasks.
- Perform RSA-based secure communication tests (e.g., secure message transfer).

Week 4-5:

- Integrate RSA with a symmetric encryption scheme to securely distribute keys.
- Encapsulate the code following object-oriented design principles using modern software engineering practices.

Week 6:

- Test encryption and decryption of files, ensuring large data handling.

Week 7-8:

- Re-evaluate project goals and address performance optimizations.

Week 9-11:

- Prepare the final project report, detailing:
    - RSA and number theory concepts.
    - Implementation challenges (e.g., data structures, numerical methods).
    - Software engineering processes followed.
    - Performance analysis of encryption programs.
- Prepare for the final presentation

# Risks and mitigation

Difficulty in Understanding Number Theory Concepts:

prime number algorithms and the mathematics behind RSA and Diffie-Hellman, can be challenging to comprehend. it can slow down the implementation process and lead to errors in cryptographic algorithms. Utilize online resources, textbooks, and tutorials to strengthen knowledge in these areas and seek guidance from my supervisor if there are conceptual roadblocks.

Difficulty in GUI Development:

Developing a graphical user interface (GUI) that allows for secure file transfer or chat could be a challenge, especially if there is limited experience with GUI design. To mitigate this start GUI development early, using high-level GUI frameworks like JavaFX and Regular feedback from potential users should be sought to ensure the interface is user-friendly.

Insecure Implementation of Cryptographic Algorithms:

Mistakes in key generation, padding, or message handling can result in security vulnerabilities, making the cryptosystem prone to attacks. Thorough testing should be done to ensure the security of key exchanges and encryption/decryption routines. Consulting my supervisor could also help identify any potential security flaws.

# REFERENCES:

[1] Kraft, J. and Washington, L. (2018). *An Introduction to Number Theory with Cryptography, Second Edition*. Milton Crc Press Ann Arbor, Michigan Proquest.

[2] Kraft, J.S. and Washington, L.C. (2016). *An Introduction to Number Theory with Cryptography*. CRC Press.