VIETNAM GENERAL CONFEDERATION OF LABOR

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**FINAL REPORT**

**DISCRETE STRUCTURES**

*Instructor*: **NGUYEN QUOC BINH**

*Executor*: **VO NHAT HAO – 522H0090**

**DANG THANH NHAN – 522H0006**

Class **: 22H50202**

Course  **: 26**

**HO CHI MINH CITY , YEAR 2024**

VIETNAM GENERAL CONFEDERATION OF LABOR

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**FINAL REPORT**

**DISCRETE STRUCTURES**

*Instructor*: **NGUYEN QUOC BINH**

*Executor*: **VO NHAT HAO – 522H0090**

**DANG THANH NHAN – 522H0006**

Class **: 22H50202**

Course  **: 26**

**HO CHI MINH CITY, YEAR 2021**

THANK YOU

* We are deeply grateful to Mr. Nguyen Quoc Binh for her constant support and enthusiastic direction throughout our investigation and final report.
* We also like to thank Ton Duc Thang University's Faculty of Information Technology for providing us with an enriching academic environment. The faculty's willingness to share vital expertise and reference materials has not only aided our research endeavor, but has also improved our overall educational experience at the university.
* As we wrap up our study project, we reflect on the vital lessons and insights learned from our educators. Regardless of our limitations and areas for improvement, we are willing to learn and grow. We really seek further assistance to improve our work and appreciate the critical input from our professors and classmates. With their continuous assistance, we are determined to improve our research talents in future initiatives.
* We wish all of our teachers and friends ongoing health and happiness, as their support and care have been invaluable to us on our path.

**WE THANK YOU!**

**THE PROJECT IS COMPLETED**

**AT TON DUC THANG UNIVERSITY**

I hereby declare that this is my own project product and is guided by Mr. Nguyen Quoc Binh. The research content and results in this topic are honest and have not been published in any form before. The data in the tables for analysis, comments, and evaluation were collected by the author from different sources and clearly stated in the reference section.

In addition, the project also uses a number of comments, assessments as well as data from other authors and other organizations, all with citations and source notes.

**If any fraud is discovered, I will take full responsibility for the content of my project**. Ton Duc Thang University is not involved in copyright violations caused by me during the implementation process (if any).

*Ho Chi Minh City, 20 May 2024*

*Author*

*(sign and write full name)*

*Vo Nhat Hao*

INSTRUCTOR VERIFICATION AND EVALUATION SECTION

**Confirmation from the instructor**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ho Chi Minh City, day month year

(sign and write full name)

**The teacher's evaluation part marks the test**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ho Chi Minh City, day month year

(sign and write full name)

SUMMARY

1. Euclid’s Algorithm and Bezout’s Identity

* Utilize Euclid's algorithm to calculate the gcd and lcm of numbers 2024 and 1000 + m, where m is the last three digits of your student ID.
* Find five integer solution pairs (x, y) that satisfy the linear equation derived from the gcd result.

1. Recurrence Relation

* Solve the recurrence relation an = 8.an-1 – 15.an-2 with initial conditions a0 = 5 and a1 = m (last two digits of your student ID).

1. Set Operations

* Create a set of characters from your case-insensitive, non-diacritical full name.
* Perform operations like union, intersection, non-symmetric difference, and symmetric difference between this set and another predefined set.

1. Relations

* Analyze a binary relation defined on integers involving divisibility by m (last two digits of your student ID).
* Determine if this relation is reflexive, symmetric, antisymmetric, and transitive.

1. Kruskal’s Algorithm

* Propose a method for circuit checking in Kruskal's algorithm to ensure that no cycles are formed while selecting edges.
* Provide an example to illustrate this method.

1. Eulerian Circuit

* Determine whether a provided graph has an Eulerian circuit or path.
* Discuss Hierholzer’s algorithm for finding an Eulerian circuit and apply it if applicable.

1. Map Coloring

* Model a provided map by a graph.
* Apply graph coloring to color the map with a minimum number of colors, using specific conditions based on the last four digits of your student ID.

1. Finding an Inverse Modulo n

* Study and describe the process of finding an inverse modulo n using the extended Euclidean algorithm.
* Implement a Python program to perform this calculation and verify its correctness with examples.

1. RSA Cryptosystem

* Conduct research on the RSA cryptosystem, including its mathematical foundations.
* Develop and test a Python program for RSA encryption and decryption.
* Analyze the efficiency and security of your implementation, discussing potential threats and providing improvement recommendations.

TABLE OF CONTENTS

[THANK YOU 3](#_Toc167483275)

[INSTRUCTOR VERIFICATION AND EVALUATION SECTION 5](#_Toc167483276)

[SUMMARY 6](#_Toc167483277)

[TABLE OF CONTENTS 8](#_Toc167483278)

[CHAPTER – 1: EUCLID’S ALGORITHM AND BEZOUT’S IDENTITY 10](#_Toc167483279)

[a) Using Euclid’s algorithm to calculate gcd(2024, 1000 + m) and lcm(2024, 1000 + m), where m is the last 3 digits of your student ID. 10](#_Toc167483280)

[b) Apply above result(s) in to find 5 integer solution pairs (x,y) of this equation: 2024x + (1000 + m)y = gcd(2024, 1000 + m) 10](#_Toc167483281)

[CHAPTER – 2: RECURRENCE RELATION 12](#_Toc167483282)

[CHAPTER – 3: SET 13](#_Toc167483283)

[a) Create a set Γ of characters from your case-insensitive non-diacritical full name. 13](#_Toc167483284)

[b) Find the union, intersect, non-symmetric difference, and symmetric difference of Γ and Δ, where Γ and Δ are from question 3a. 13](#_Toc167483285)

[CHAPTER – 4: RELATIONS 14](#_Toc167483286)

[CHAPTER – 5: KRUSKAL’S ALGORITHM 16](#_Toc167483287)

[CHAPTER – 6: EULERIAN CIRCUIT 18](#_Toc167483288)

[a) Does the following graph have an Eulerian circuit or Eulerian path? Why? 18](#_Toc167483289)

[b) Study and present your knowledge about Hierholzer’s algorithm to find an Eulerian circuit. 19](#_Toc167483290)

[c) If the graph has an Eulerian circuit, use Hierholzer's algorithm to find an Eulerian circuit of that graph when the initial circuit R1 is: 20](#_Toc167483291)

[CHAPTER – 7: MAP COLORING 21](#_Toc167483292)

[a) Modeling this map by a graph. 21](#_Toc167483293)

[b) Color the map (graph) with a minimum number of colors. Present your solution step by step. 21](#_Toc167483294)

[CHAPTER – 8: FINDING AN INVERSE MODULO N 22](#_Toc167483295)

[CHAPTER – 9: RSA CRYPTOSYSTEM 23](#_Toc167483296)

CHAPTER – 1: EUCLID’S ALGORITHM AND BEZOUT’S IDENTITY

1. Using Euclid’s algorithm to calculate gcd(2024, 1000 + m) and lcm(2024, 1000 + m), where m is the last 3 digits of your student ID.

(Student code 522H0006 is the smallest of the group's 3 student codes, we have gcd(2024, 1006) and lcm(2024, 1006)).

* gcd(2024, 1006) 🡪 gcd(1006, 12) => 2024 = 1006 2 + 12

🡪 gcd(12, 10) => 1006 = 12 83 + 10

🡪 gcd(10, 2) => 12 = 10 1 + 2

🡪 gcd(2, 0) => 10 = 2 5 + 0

🡺 **Thus gcd(2024, 1006) = 2**

* lcm(2024, 1006)

2024 = 11 23

1006 = 2 503

🡺 **lcm(2024, 1006) = 11 23 503 = 1018072**

1. Apply above result(s) in to find 5 integer solution pairs (x,y) of this equation: 2024x + (1000 + m)y = gcd(2024, 1000 + m)

We have: 2024x + 1006y = 2

2 = 12 – 10 1 = 12 + 10 ()

= 12 + (1006 12 83) () = 1006 () + 12 84

= 1006 () + (2024 1006 2) 84 = 2024 84 + 1006 ()

🡺 Thus: 2 = 2024 84 + 1006 ()

🡺 x = 84, y = , d = 2

(x , y ), where k is any integer.

⬄ (84 , )

* k = 0: (84 , ) = (84, )
* k = 1: (84 , ) = (587, )
* k = : (84 , ) = (, )
* k = : (84 , ) = (, )
* k = : (84 , ) = (, )

🡺 The 5 pairs of integer solutions (x,y) of the equation are:

(84, ), (587, ), (, ), (, ), (, ).

CHAPTER – 2: RECURRENCE RELATION

Solve this recurrence relation.

= 8. 15.

with = 5 and = m, where m is the last 2 digits of your student ID. We have 522H0006 then = 6.

* Hence:

* Explicit formula:

* We have:
* Thus:

CHAPTER – 3: SET

1. Create a set Γ of characters from your case-insensitive non-diacritical full name.

Student code 522H0006 has full name Dang Thanh Nhan then Γ =

1. Find the union, intersect, non-symmetric difference, and symmetric difference of Γ and Δ, where Γ and Δ are from question 3a.

Γ =

Δ =

* Union (Γ ∪ Δ): The union of two sets contains all unique elements from both sets.

Γ ∪ Δ = {A, D, N, G, T, H, C, O, U}

* Intersection (Γ ∩ Δ): The intersection of two sets contains elements that are common to both sets.

Γ ∩ Δ = {A, D, N, G, T, H}

* Non-symmetric difference: The non-symmetric difference between two sets contains elements that are in one set but not in the other.

Γ Δ Δ = {C, O, U}

* Symmetric difference (Γ Δ Δ): The symmetric difference between two sets contains elements that are in one set or the other, but not in both.

Γ Δ Δ = {C, O, U}

CHAPTER – 4: RELATIONS

Student ID is 5220006 then the valid binary relation is

∀𝑎, 𝑏 ∈ N (aRb↔06|(𝑎.𝑏))

**Reflexive:**

A relation 𝑅 R on a set is reflexive if every element is related to itself. That is, we need to verify if:

∀ 𝑎 ∈ 𝑁 , 𝑎 𝑅 𝑎 ⟺ 06 ∣ ( 𝑎 ⋅ 𝑎 ) ∀a∈N,aRa⟺06∣(a⋅a)

Since 06 = 2 ⋅ 3 06=2⋅3, 𝑎 𝑅 𝑎 aRa holds if and only if 𝑎 2 a 2 is divisible by both 2 and 3, which implies 𝑎 2 a 2 is divisible by 6. However, 𝑎 2 a 2 is divisible by 6 only if 𝑎 a itself is divisible by both 2 and 3 (i.e., 𝑎 a is divisible by 6). This is not true for all natural numbers 𝑎 a. For example, 𝑎 = 1 a=1 results in 1 2 = 1 1 2 =1 which is not divisible by 6.

Conclusion: The relation 𝑅 R is not reflexive.

**Symmetric:**

A relation 𝑅 R on a set is symmetric if for all pairs ( 𝑎 , 𝑏 ) (a,b), whenever 𝑎 𝑅 𝑏 aRb, then 𝑏 𝑅 𝑎 bRa also holds. Based on the definition:

𝑎 𝑅 𝑏 ⟺ 06 ∣ ( 𝑎 ⋅ 𝑏 ) ⟹ 𝑏 𝑅 𝑎 ⟺ 06 ∣ ( 𝑏 ⋅ 𝑎 )

Since multiplication is commutative ( 𝑎 ⋅ 𝑏 = 𝑏 ⋅ 𝑎 a⋅b=b⋅a), the relation is symmetric. Conclusion: The relation 𝑅 R is symmetric.

**Antisymmetric**:

A relation 𝑅 R is antisymmetric if for all pairs ( 𝑎 , 𝑏 ) (a,b), whenever 𝑎 𝑅 𝑏 aRb and 𝑏 𝑅 𝑎 bRa both hold, it must be that 𝑎 = 𝑏 a=b. However, since 𝑎 𝑅 𝑏 aRb depends solely on the divisibility of their product by 6, different numbers can satisfy this condition without being equal. For example, 6 𝑅 12 6R12 and 12 𝑅 6 12R6 both hold because 72 72 and 144 144 are divisible by 6, but 6 ≠ 12.

Conclusion: The relation 𝑅 R is not antisymmetric.

**Transitive:**

A relation 𝑅 R is transitive if whenever 𝑎 𝑅 𝑏 aRb and 𝑏 𝑅 𝑐 bRc hold, then 𝑎 𝑅 𝑐 aRc also holds. Considering 𝑎 𝑅 𝑏 aRb implies 06 ∣ ( 𝑎 ⋅ 𝑏 ) 06∣(a⋅b) and 𝑏 𝑅 𝑐 bRc implies 06 ∣ ( 𝑏 ⋅ 𝑐 ) 06∣(b⋅c), it does not necessarily imply 06 ∣ ( 𝑎 ⋅ 𝑐 ) 06∣(a⋅c). For example, if 𝑎 = 6 a=6, 𝑏 = 10 b=10, and 𝑐 = 15 c=15, then 𝑎 𝑅 𝑏 aRb and 𝑏 𝑅 𝑐 bRc hold as 60 60 and 150 150 are divisible by 6, but 6 ⋅ 15 = 90 6⋅15=90 is not divisible by 6, hence 𝑎 𝑅 𝑐 aRc does not hold.

Conclusion: The relation 𝑅 R is not transitive.

* Summarily, the relation 𝑅 R defined by 06 ∣ ( 𝑎 ⋅ 𝑏 ) 06∣(a⋅b) is symmetric but not reflexive, not antisymmetric, and not transitive.

CHAPTER – 5: KRUSKAL’S ALGORITHM

Kruskal's algorithm is a popular method used in graph theory to find the minimum spanning tree (MST) for a connected, weighted graph. The algorithm works by sorting all the edges of the graph by their weights in non-decreasing order and then selecting edges for the MST, ensuring that no cycles are formed.

Circuit (Cycle) Checking

The key challenge in Kruskal's algorithm is ensuring that the selected edges do not form a cycle. If a cycle is formed, the set of edges no longer constitutes a tree.

Solutions for Circuit Checking

There are primarily two methods commonly used to check for cycles when implementing Kruskal’s algorithm:

1. Union-Find Data Structure (Disjoint Set Union - DSU)
2. Depth-First Search (DFS) or Breadth-First Search (BFS)

Union-Find Data Structure (Recommended)

Union-Find is an efficient data structure to keep track of partitions of sets and is highly suited for checking connectivity and detecting cycles in a graph during the execution of Kruskal's algorithm.

Operations needed:

Find: Determine which subset a particular element is in. This can be used for determining if two elements are in the same subset.

Union: Join two subsets into a single subset.

Cycle Checking Using Union-Find:

1. Initialization: Each vertex of the graph is initially considered as a separate subset.
2. Edge Selection: For each edge in the sorted list, check the subsets to which the two vertices of the edge belong using the Find operation:

* If the vertices belong to the same subset, including this edge would form a cycle, so skip it.
* If the vertices belong to different subsets, include this edge in the MST and combine the subsets using the Union operation.

CHAPTER – 6: EULERIAN CIRCUIT

1. Does the following graph have an Eulerian circuit or Eulerian path? Why?

A grid with letters and numbers

Description automatically generated

To determine whether a graph has an Eulerian circuit or an Eulerian path, we need to examine the degrees (number of edges connected) of each vertex in the graph. Here are the key properties to consider:

* Eulerian Circuit: An Eulerian circuit is a cycle in a graph that traverses each edge exactly once and returns to the starting vertex. And the graph must be connected, and all vertices in the graph must have an even degree.
* Eulerian Path: An Eulerian path is a path in a graph that traverses each edge exactly once but does not necessarily return to the starting vertex. And The graph must be connected and must have exactly two vertices of odd degree (all other vertices have an even degree).

Analysis of the Given Graph

The provided graph appears as a mesh grid enclosed within a polygon, with vertices labeled from A to n. To assess the presence of an Eulerian circuit or path, we'll evaluate the degrees of each vertex, focusing particularly on the boundary vertices, as the inner vertices clearly have a degree of 4 (even).

Boundary Vertices Degrees:

* Corners (A, C, G, D): Each corner vertex is connected to three edges.
* Edge vertices (excluding corners, like E, F, B, H, etc.): Each of these vertices is connected to four edges (even).

Since the corner vertices have an odd degree (3), and there are four such vertices, we can deduce:

* The graph cannot have an Eulerian circuit because there are more than two vertices with an odd degree.
* The graph cannot have an Eulerian path either, because there are more than two vertices with an odd degree.

Conclusion

Based on the degree of vertices, the graph does not support an Eulerian path or an Eulerian circuit due to the presence of four vertices with an odd degree.

1. Study and present your knowledge about Hierholzer’s algorithm to find an Eulerian circuit.

Hierholzer's algorithm is a classical method used for finding an Eulerian circuit in a graph that has one. It's important to note that the algorithm is applicable only if the graph has all vertices with even degree and is connected. Here's a step-by-step explanation of how Hierholzer's algorithm works:

Conditions for Eulerian Circuit

* All vertices with non-zero degree are connected.
* Every vertex has an even degree.

Steps of Hierholzer’s Algorithm

1. Choose a starting vertex:

* Start at any vertex 𝑣 v that has a non-zero degree.

1. Form a cycle/circuit:

* From vertex 𝑣 v, follow edges to form a circuit that returns to 𝑣 v. This initial cycle might not include all the edges.
* During this process, ensure that you choose unused edges at each step. If you reach a vertex and no unused outgoing edges are left, the algorithm stops prematurely. This happens only if the graph is not Eulerian.

1. Check if all edges are used:

* After creating the initial cycle, check if there are any unused edges. If all edges have been used, you have found your Eulerian circuit, and the algorithm ends.

1. Extend the cycle:

* If there are unused edges, select a new starting vertex on the existing cycle which has unused edges.
* Form a new cycle from this new start vertex, again ensuring that you return to the starting point using unused edges.
* Merge this cycle into the previous cycle, maintaining the cycle property, to form a new, extended cycle.

1. Repeat the extension:

* Repeat the extension process as many times as necessary until all edges in the graph are used.

1. Complete the circuit:

* The final cycle formed after all edges are included once and only once is the Eulerian circuit.
* Key Points
* Hierholzer’s algorithm efficiently finds an Eulerian circuit by incrementally extending cycles until all edges are included.
* The algorithm runs in 𝑂 ( 𝐸 ) O(E) time, where 𝐸 E is the number of edges, because each edge is used exactly once and each time an edge is added to the cycle, it is processed in constant time.
* This algorithm is practical for computational applications because of its straightforward implementation and efficient execution.
* Hierholzer’s method is elegant due to its simplicity and effectiveness in constructing Eulerian circuits, provided the graph meets the necessary conditions. It is widely used in applications that require the traversal of every edge of a graph exactly once, such as in the design of efficient routes and network cabling.

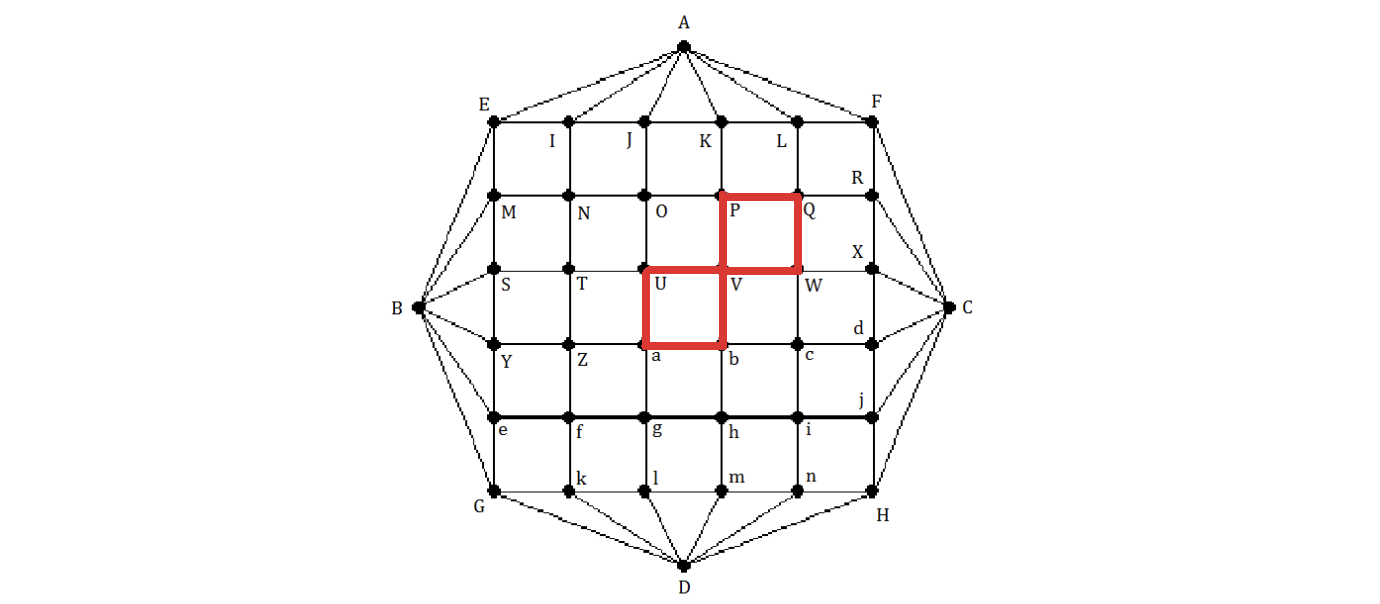
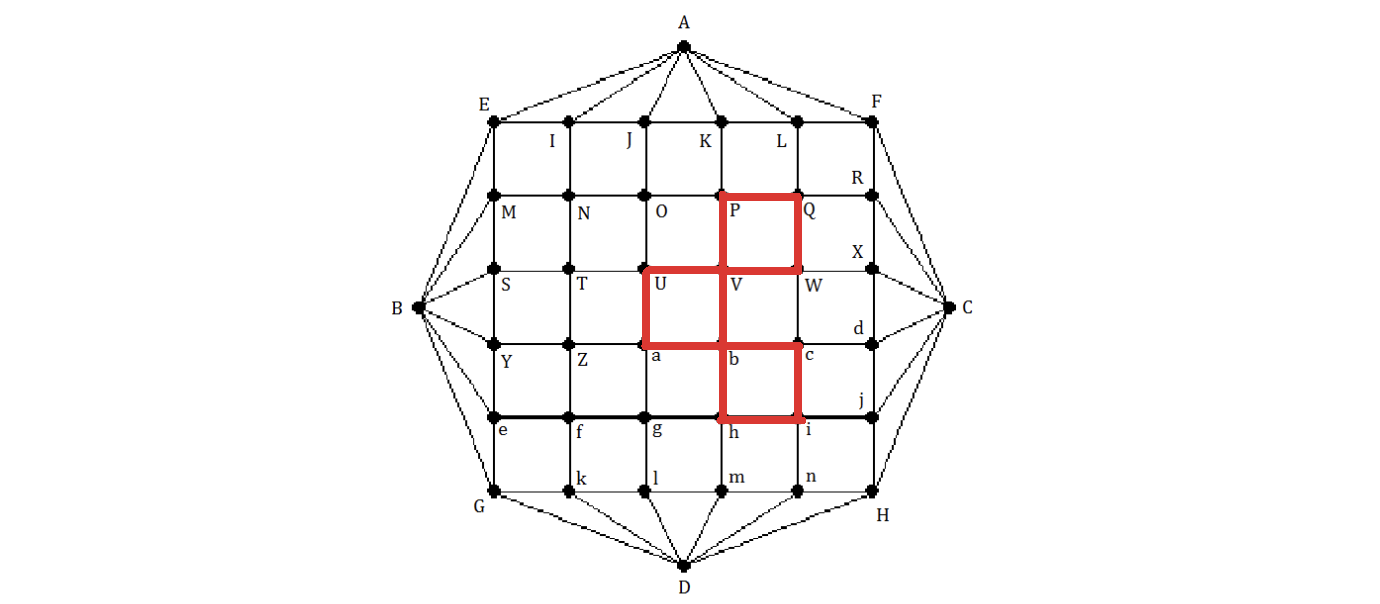
1. If the graph has an Eulerian circuit, use Hierholzer's algorithm to find an Eulerian circuit of that graph when the initial circuit R1 is:

Student ID 5220006, % 4 = 2. Then, R1 is UVbaU.

* is UVbaU

A grid with letters and numbers

Description automatically generated

* Step1: = VPQWV
  + = UVPQWVbaU
* Step2: = bcihb
  + = UVPQWVbcihbaU
* A red square with black lines and dots

  Description automatically generatedStep3: = aZfga
  + = UVPQWVbcihbaZfgaU
* A diagram of a grid with letters and numbers

  Description automatically generatedStep4: = QRFLQ
  + = UVPQRFLQWVbcihbaZfgaU
* A grid with letters and numbers

  Description automatically generatedStep5: = ijHni
  + = UVPQRFLQWVbcijHnihbaZfgaU
* A diagram of a grid with letters and numbers

  Description automatically generatedStep6: = fkGef
  + = UVPQRFLQWVbcijHnihbaZfkGefgaU
* A diagram of a grid with letters and numbers

  Description automatically generatedStep7: = WXdcW
  + = UVPQRFLQWXdcWVbcijHnihbaZfkGefgaU
* A grid of red squares

  Description automatically generatedStep8: = hmlgh
  + = UVPQRFLQWXdcWVbcijHnihmlghbaZfkGefgaU
* A diagram of a grid with letters and numbers

  Description automatically generatedStep9: = ZYSTZ
  + = UVPQRFLQWXdcWVbcijHnihmlghbaZYSTZfkGefgaU
* A diagram of a grid with letters and numbers

  Description automatically generatedStep10: = POJKP
  + = UVPOJKPQRFLQWXdcWVbcijHnihmlghbaZYSTZfkGefgaU
* A diagram of a grid with letters and numbers

  Description automatically generatedStep11: = ONTUO
  + = UVPONTUOJKPQRFLQWXdcWVbcijHnihmlghbaZYSTZfkGefgaU
* A grid of red squares

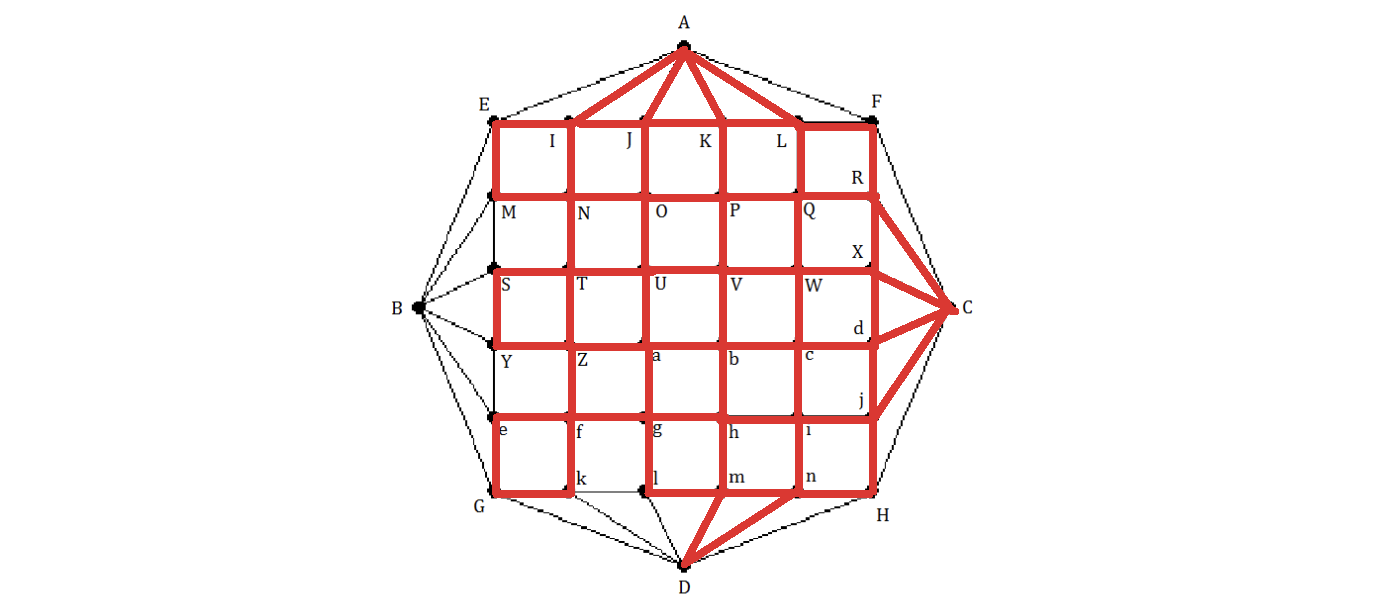
  Description automatically generatedStep12: = NIEMN
  + =UVPONIEMNTUOJKPQRFLQWXdcWVbcijHnihmlghbaZYSTZfkGefgaU
* A diagram of a grid

  Description automatically generatedStep13: = IAJI
  + =UVPONIAJIEMNTUOJKPQRFLQWXdcWVbcijHnihmlghbaZYSTZfkGefgaU
* A diagram of a hexagon with red squares

  Description automatically generatedStep14: = LAKL
  + =UVPONIAJIEMNTUOJKPQRFLAKLQWXdcWVbcijHnihmlghbaZYSTZfkGefgaU
* A red grid with black lines and letters

  Description automatically generatedStep15: = RCXR
  + =UVPONIAJIEMNTUOJKPQRCXRFLAKLQWXdcWVbcijHnihmlghbaZYSTZfkGefgaU
* A red grid with black lines and letters

  Description automatically generatedStep16: = jCdj
  + =UVPONIAJIEMNTUOJKPQRCXRFLAKLQWXdcWVbcijCdjHnihmlghbaZYSTZfkGefgaU
* Step17: = nDmn



* + =UVPONIAJIEMNTUOJKPQRCXRFLAKLQWXdcWVbcijCdjHnDmnihmlghbaZYSTZfkGefgaU
* A red grid with white letters

  Description automatically generatedStep18: = kDlk
  + =UVPONIAJIEMNTUOJKPQRCXRFLAKLQWXdcWVbcijCdjHnDmnihmlghbaZYSTZfkDlkGefgaU
* Step19: = eBYe

A red grid with letters and numbers

Description automatically generated

* + =UVPONIAJIEMNTUOJKPQRCXRFLAKLQWXdcWVbcijCdjHnDmnihmlghbaZYSTZfkDlkGeBYefgaU
* A red grid with letters and numbers

  Description automatically generatedStep20: = MBSM

* + =UVPONIAJIEMBSMNTUOJKPQRCXRFLAKLQWXdcWVbcijCdjHnDmnihmlghbaZYSTZfkDlkGeBYefgaU
* A red grid with white letters

  Description automatically generatedStep21: = AFCHDGBEA

* + =UVPONIAFCHDGBEAJIEMBSMNTUOJKPQRCXRFLAKLQWXdcWVbcijCdjHnDmnihmlghbaZYSTZfkDlkGeBYefgaU
  + So, the Eulerian circuit of that graph: UVPONIAFCHDGBEAJIEMBSMNTUOJKPQRCXRFLAKLQWXdcWVbcijCdjHnDmnihmlghbaZYSTZfkDlkGeBYefgaU

CHAPTER – 7: MAP COLORING

1. Modeling this map by a graph.
2. Color the map (graph) with a minimum number of colors. Present your solution step by step.

CHAPTER – 8: FINDING AN INVERSE MODULO N

* 1. Theorical research
     1. Inverse modulo

In modular arithmetic, the inverse modulo is a crucial concept that allows us to perform various mathematical operations efficiently. Given a number a and a modulo n, the inverse modulo of a modulo n is find number x that satisfies the equation ax ≡ 1 (mod n). This means that the remainder of ax when divided by 𝑛 equals 1.

The inverse modulo only exists if a and n are coprime, that is, if their greatest common divisor is 1. If a and n are not coprime, then there is no inverse modulo. The inverse modulo is an important concept in number theory and has various applications in cryptography, computer science, and other fields.

Finding the inverse modulo n using the extended Euclidean algorithm is an important concept in number theory that has various applications in cryptography, computer science, and other fields. In this context, the extended Euclidean algorithm is used to calculate the greatest common divisor (GCD) of two numbers a and b, as well as to find two integers x and y such that ax + by = GCD(a, b).

* + 1. The Extended Euclidean algorithm

To find the inverse modulo n using the extended Euclidean algorithm, we need to find two integers x and y such that ax + by = 1, where a is the number to find the inverse and b is the modulo. Once we have found x, it is the inverse of a modulo b.

1.13 Example

Let's find the inverse of 19 modulo 47. We can use the extended Euclidean algorithm to find x and y such that 19x +47y = 1. Starting with a = 19 and b = 47, we can use the following steps:

* Step1: Divide 47 by 19 to get a quotient of 2 and a remainder of 9.
* This means 47 = 2\*19+9.
* Step2: Divide 19 by 9 to get a quotient of 2 and a remainder of 1.
* This means 19 = 2\*(9)+1.

Now we can work backwards to express 1 as a linear combination of 20 and 23:

1 = 19 – 2\*(9) = 19 – 2\*(47 – 2\*(19)) = 5\*(19) – 2\*(47)

So x = 5 and y = –2, which means the inverse of 19 modulo 47 is 5.

* 1. Implement extended Euclidean algorithm

To implement the extended Euclidean algorithm in Python, we can define a function that takes two integers a and b as arguments and returns the greatest common divisor (GCD) of a and b, as well as x and y such that ax + by = GCD(a, b).

We can then define another function that takes two integers a and n as arguments and uses the extended Euclidean algorithm to find the inverse of a modulo.

Here is the Python code to implement the extended Euclidean algorithm to find the inverse modulo:

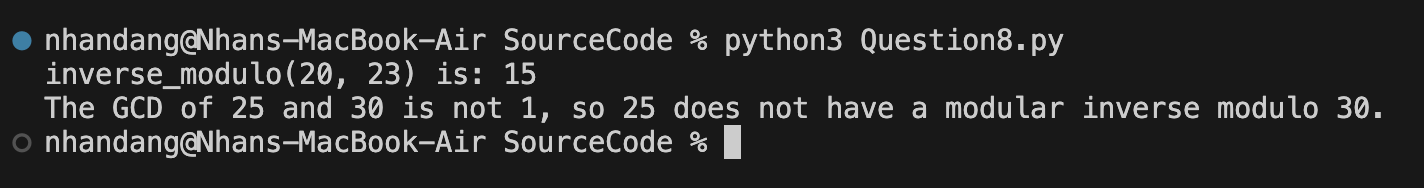
A computer screen shot of a black screen with white text

Description automatically generated

* The function **'extended\_euclid(a, b)**' implements the extended Euclidean algorithm to calculate the greatest common divisor (GCD) of two numbers 'a' and 'b', as well as to find two integers 'x' and 'y' such that ax + by = GCD(a, b).
* Then, the function **'inverse\_modulo(a, b)**' uses the **'extended\_euclid()**' function to find the GCD and two integers 'x' and 'y' such that ax + by = GCD(a, b). If the GCD is not equal to 1, it means that 'a' does not have a modular inverse modulo 'b'. Otherwise, the function returns the value of 'x' modulo 'b', which is the inverse modulo of 'a' modulo b.
  1. Test the implemented program
* **Test case:**

A black background with white text

Description automatically generated

* Result:
* Explain:
  + Test case 1: Find inverse\_modulo (20,23)
* First, I find the greatest common divisor (GCD) of 20 and 23 using the Euclidean algorithm:

23 = 1 \* 20 + 3

20 = 6 \* 3 + 2

3 = 1 \* 2 + 1

* The last non-zero remainder is 1, so the GCD(20, 23) = 1. Since the GCD is 1, we know that 20 has a modular inverse modulo 23.
* Next, we work backwards to express the GCD as a linear combination of 20 and 23:

1 = 3 - 1 \* 2

1 = 3 - 1 \* (20 - 6 \* 3)

1 = -1 \* 20 + 7 \* 3

1 = -1 \* 20 + 7 \* (23 - 1 \* 20)

1 = -8 \* 20 + 7 \* 23

* From this, we can see that -8 is a solution to the equation 20x ≡ 1 (mod 23). To find the smallest positive solution, we can add 23 to -8 until we get a positive number: -8 + 23 = 15. So, 15 is the modular inverse of 20 modulo 23.
  + Therefore, **inverse\_modulo (20, 23) = 15**
* Test case 2: Find inverse\_modulo (15, 20)
* First, we need to find the greatest common divisor (gcd) of 15 and 20 using the Euclidean algorithm:

20 = 1 \* 15 + 5

15 = 3 \* 5 + 0

* The last non-zero remainder is 5, so gcd(15,20) = 5. Since the gcd is not equal to 1, we know that 15 does not have a modular inverse modulo 20.
  + Therefore, **inverse\_modulo(15,20)** does not exist.

CHAPTER – 9: RSA CRYPTOSYSTEM

* 1. Theory research
* **RSA** is a public-key cryptosystem that is widely used for secure data transmission. It was developed by **Ron Rivest**, **Adi Shamir**, and **Leonard Adleman** in 1977, and is named after their initials. The security of RSA is based on the difficulty of factoring large integers into their prime factors, which is currently believed to be an intractable problem for classical computers.
* The following mathematical ideas form the foundation of the RSA cryptosystem:
* **Generation of keys:**
* **Step1**: Choose two large prime numbers of the same length: p, q
* **Step2**: Compute n and 𝜙 (𝑛):
  + n = p \* q
  + 𝜙 (𝑛) = (p−1) \* (q−1).
* **Step3**: Choose the **Public Exponent** e:
  + Choose e, which has a range greater than 1 and smaller than 𝜙 (𝑛). And coprime with 𝜙 (𝑛). We often utilize the value of e as (2\*k+1) for numbers like 3, 17, 19, 65537.
* **Step4:** Compute the **Private Exponent** d:
  + Determine d such that d × e ≡ 1 mod   𝜙 (𝑛)
* The public key consists of (n, e), and the private key consists of (n, d). The primes p and q should be kept secret.
* **Encryption algorithm:** Given a known plaintext x (0 ≤ x < n), divide x into character blocks. Then calculate the ciphertext C = mod n.
* **Decryption algorithm:** Given a known ciphertext c and the private key (n, d) the plaintext x can be calculated as x = (mod n).
* **Security of the RSA algorithm:** The security of the RSA algorithm relies on the difficulty of factoring the large number 𝑛 n into its prime factors p and q. If an attacker could factor n, they could compute 𝜙 (𝑛) and subsequently determine d from e. This problem, known as integer factorization, is considered computationally infeasible for sufficiently large n (e.g., 2048-bit keys).
* **Example:**
* **Step 1: Initialize Parameters**

+ Choose two prime numbers: p=17 and q=11.

+ Calculate 𝑁 = 𝑝 × 𝑞 = 17 × 11 = 187.

+ Calculate 𝜙 (𝑁) = (𝑝 − 1) × (𝑞 − 1) = 16 × 10 = 160.

+ Choose 𝑒 = 7, with 𝑒 and 𝜙 (𝑁) being coprime.

+ Find d such that 𝑒 × 𝑑 ≡ 1 m o d 𝜙 (𝑁). Using the extended Euclidean algorithm, we find 𝑑 = 23 because 7 × 23 ≡ 1 m o d 160.

* The public key is (𝑁, 𝑒) = (187, 7).
* The private key is (𝑁, 𝑑) = (187, 23)
* **Step 2: Encrypt the Message**

- Assume the message is "HELLO":

+ Use the ASCII table to convert the string to a sequence of numbers:

H = 72, E = 69, L = 76, L = 76, O = 79

- Since each character's encoded value must be smaller than 𝑁, we will

encrypt each character individually:

+ Encrypt H: mod 187 = 30

+ Encrypt E: mod 187 = 86

+ Encrypt L: mod 187 = 32

+ Encrypt L: mod 187 = 32

+ Encrypt O: mod 187 = 139

* The encrypted sequence is: 𝐶 = 30, 55, 72, 72, 44.

* **Step 3: Decrypt the Message**

- The recipient uses the private key (𝑁, 𝑑) = (187, 23) to decrypt the message:

+ Decrypt 30: mod 187 = 72

+ Decrypt 86: mod 187 = 69

+ Decrypt 32: mod 187 = 76

+ Decrypt 32: mod 187 = 76

+ Decrypt 139: mod 187 = 79

- Convert the numbers back to ASCII characters: 72 = H, 69 = E, 76 = L, 76 = L, 79 = O

* Decryption result: “HELLO”.
  1. Implement a Python program

We can either implement manually or use Python libraries to support cryptography. In this case, we have alots options: the Crypto library (cryptography, cryptodome) and the rsa module. For this tutorial, I will be using the rsa library because I just need rsa algorithm and it result return big int number while cryptography or cryptodome return base64.To run this code you need install rsa library (python version 3.10.6). Here is the code applied to RSA encryption:

A computer screen shot of text

Description automatically generated

* 1. Test the implemented program
* **Plain text:**
* A screenshot of a computer screen

  Description automatically generated**Result:**

The terminal show that decrypt message equal original message. The public key and private key store value I mention above. The last line I compare plain text and decrypted text, it returns true meaning successful decrypt.

* 1. Analyze the efficiency and security of the implemented RSA cryptosystem
     1. Efficiency
* RSA encryption is often used in combination with other encryption schemes and for digital signatures that can verify the authenticity and integrity of a message. It is usually not used to encrypt entire messages or files because it is less efficient and more resource-intensive than symmetric key encryption.
* To make things more efficient, a file is usually encrypted using a symmetric key algorithm. Then the symmetric key is encrypted using RSA encryption. In this process, only the person with access to the RSA private key can decrypt the symmetric key. If the symmetric key cannot be accessed, then the original file cannot be decrypted. This method can be used to secure messages and files without taking up too much time and resources.
* RSA encryption can be used in various systems. It can operate in OpenSSL, wolfCrypt, cryptlib, and other cryptographic libraries. Traditionally, it has been used in TLS and was also the initial algorithm used in PGP encryption. RSA is still seen in a variety of web browsers, email, VPNs, chat, and other communication channels.
* RSA is also commonly used to create secure connections between VPN clients and VPN servers. In protocols such as OpenVPN, TLS can use the RSA algorithm to exchange keys and establish a secure channel.
  + 1. Security
* Secure communication is when two entities are communicating and do not want a third party to listen in. For this to be the case, the entities need to communicate in a way that is unsusceptible to eavesdropping or interception. There are several requirements for secure communication which rsa have, including:
  + - **Confidentiality:** The content of the communication should be kept secret from anyone who is not authorized to access it. (Has been encrypt)
    - **Integrity:** The content of the communication should not be altered in transit without being detected. (Encrypt message equal original message)
    - **Authentication:** The identity of the sender and receiver should be verified to ensure that they are who they claim to be. (Has private key)
    - **Non-repudiation:** The sender should not be able to deny sending the message, and the receiver should not be able to deny receiving it. (Ensures that the sender of a message cannot deny sending it and the receiver cannot deny receiving it. In the RSA (Rivest-Shamir-Adleman) algorithm, non-repudiation is achieved using digital signatures. A digital signature is a mathematical scheme that allows a sender to prove the authenticity and integrity of a message to a receiver. When the sender signs a message with their private key, the receiver can verify the signature using the sender's public key. If the verification process is successful, the receiver can be assured that the message was indeed sent by the sender and has not been tampered with in transit. This makes it difficult for either party to deny their involvement in the communication.
* Security Basis: RSA's security relies on the difficulty of factoring large integers. Security experts and researchers continuously analyze RSA to identify vulnerabilities and enhance the algorithm. Key security analyses include:
  + - **Mathematical analysis** of RSA involves analyzing the properties of prime numbers, mathematical functions used in the algorithm, and the properties of modular arithmetic.
    - **Encryption and decryption analysis** of RSA involves analyzing the encryption and decryption processes and the properties of the keys used.
    - **Vulnerability testing** of RSA involves testing the algorithm for potential vulnerabilities and weaknesses.
    - **Detecting potential attacks** involves analyzing the different methods attackers could use to exploit RSA's vulnerabilities.
  1. Discuss the potential security threats and limitations of the RSA cryptosystem

1.5.1 The potential security threats

* **Mathematical Analysis Threats** 
  + **Prime Number Factorization:** The security of RSA relies heavily on the difficulty of factoring large prime numbers. Advances in factorization algorithms, such as the General Number Field Sieve (GNFS), could potentially reduce the time required to factorize the large composite numbers used in RSA keys.
  + **Quantum Computing:** Quantum algorithms, particularly Shor's algorithm, pose a significant threat to RSA. Quantum computers could theoretically factorize large integers exponentially faster than classical computers, rendering RSA insecure.
* **Encryption and Decryption Analysis Threats** 
  + **Weak Key Generation:** Improper generation of RSA keys, such as using small primes or predictable patterns, can make the system vulnerable to attacks. Ensuring that the primes are large and randomly generated is crucial for security.
  + **Side-Channel Attacks:** These attacks exploit information leaked during the encryption and decryption processes. Timing attacks, power analysis, and electromagnetic leaks can potentially reveal the private key without directly attacking the mathematical structure of RSA.
* **Vulnerability Testing Threats** 
  + **Implementation Flaws:** Vulnerabilities can arise from incorrect or insecure implementations of the RSA algorithm. For example, improper padding schemes (e.g., PKCS#1 v1.5) have led to attacks such as the Bleichenbacher attack, which can decrypt messages without the private key.
  + **Backdoors in Software:** Malicious backdoors intentionally placed in software that implements RSA can allow attackers to bypass encryption entirely. Ensuring the integrity and security of cryptographic software is essential.
* **Potential Attack Methods** 
  + **Chosen Ciphertext Attacks (CCA):** In these attacks, the attacker can choose a ciphertext and obtain its decryption under an unknown key. Adaptive chosen-ciphertext attacks (CCA2) can be particularly effective against systems that do not employ proper padding schemes.
  + **Timing Attacks:** By measuring the time it takes for certain operations, attackers can gain information about the private key. Variations in timing due to different computations can provide clues to an attacker.
  + **Mathematical Attacks:** Advanced mathematical techniques, such as lattice-based attacks, can be used to solve problems related to RSA's security assumptions. These attacks become more feasible as computational power increases.

1.5.2 The limitations

* Performance: RSA encryption and decryption operations can be slow, especially when dealing with large amounts of data. This can be a problem in real-time applications where speed is critical.
* Time limits: I conducted an experiment (100 times per case) by measuring the execution time of the RSA algorithm and the combined RSA and AES algorithm on the same piece of text and obtained the key bit sizes as shown in the table below. I then used the matplotlib library to graph the results as shown in table and the figure below: (Note that running speed may depend on computer hardware, Python version, and randomly generated keys, so the algorithm running time is different for each program execution.)

|  |  |  |  |
| --- | --- | --- | --- |
| Number Bits of key | 2048 | 3072 | 4096 |
| RSA time | 0.026126 | 0.052841 | 0.122883 |
| RSA & AES time | 0.008894 | 0.026709 | 0.062173 |

A graph with a blue line and orange line

Description automatically generated

* **Comment:**
* After conduct experiment, I see that RSA runtime longer than RSA mixed AES ((Advanced Encryption Standard) is a symmetric encryption algorithm, which means that the same key is used for both encryption and decryption. It was introduced as a replacement for the outdated DES (Data Encryption Standard) algorithm).
* Time complete depend on number of bit and length of plain text. Key size: RSA requires large key sizes to ensure security, which can be a problem in certain applications where the available storage space is limited.
* This experiment does not conclude that the RSA algorithm is inferior to other algorithms in terms of speed and security, but rather shows that the RSA algorithm can increase operational efficiency if combined with other algorithms.
  1. Recommendations
* RSA is still widely used and considered a secure cryptosystem when used with sufficiently large keys. To enhance the security of RSA, it is crucial to use large keys, implement the algorithm carefully, and update the keys regularly. Additionally, using block cipher algorithms such as AES to encrypt data and then using RSA to encrypt the AES key can further enhance the security of RSA, making it more robust against potential vulnerabilities and attacks.
* In conclusion, analyzing the security and limitations of the RSA algorithm is essential to ensure the safety of information encryption in real-world applications. Security experts and researchers should continue to update, research, and develop new methods to enhance the security of RSA against potential vulnerabilities and attacks. By continuously improving the security of RSA, we can ensure that it remains a reliable and secure method for data encryption.