GAYATRI VIDYA PARISHAD

COLLEGE FOR DEGREE AND PG COURSES (A)

(Affiliated to Andhra University)

Rushikonda, Visakhapatnam

Department of Computer Applications

VISION

"Creating human excellence for a better society"

MISSION

"Unfold into a world class organization with strong academic and research base, producing responsible citizens to cater to the changing needs of the society."

A CRYPTOGRAPHIC ALGORITHM BASED ON ASCII AND NUMBER SYSTEM CONVERSIONS ALONG WITH A CYCLIC MATHEMATICAL FUNCTION

A Project report submitted in

partial fulfillment of the

requirement for the Award of the

Degree of

Master of Computer Applications

Submitted By

B. SATYA PAVAN

(Regd. No: PG212202006)

Under the Esteemed Guidance of

B. DIVAKAR

Assistant Professor

Department of Computer Applications

GAYATRI VIDYA PARISHAD

COLLEGE FOR DEGREE AND PG COURSES(A)

(Affiliated to Andhra University)

Rushikonda, Visakhapatnam.

2021-2023

GAYATRI VIDYA PARISHAD

COLLEGE FOR DEGREE AND PG COURSES(A)

(Affiliated to Andhra University)

Rushikonda, Visakhapatnam.

Department of Computer Applications

CERTIFICATE

This is to certify that the project report titled "**A CRYPTOGRAPHIC ALGO- RITHM BASED ON ASCII AND NUMBER SYSTEM CONVERSIONS ALONG WITH A CYCLIC MATHEMATICAL FUNCTION"** is the bona-fide record of project work carried out by **Mr. B. SATYA PAVAN (Regs. No. PG212202006),** a student of this college, during the academic year 2021-2023, in partial fulfillment of the requirement for the award of the degree of Master of Computer Applications.

Project Guide
B.DIVAKAR
Assistant Professor

Director of MCA
Prof I. S. Pallavi,

External Examiner

DECLARATION

I, B. SATYA PAVAN hereby declares that the project report entitled "**A Cryptographic Algorithm Based on ASCII and Number System Conversions along with a Cyclic Mathematical Function",** is an original work done at **Gayatri Vidya Parishad College for Degree and PG .Courses (Autonomous), Visakhapatnam,** submitted in partial fulfillment of the requirements for the award of Master of Computer Applications, Gayatri Vidya Parishad College for Degree and PG Courses(A), affiliated to Andhra University. I assure that this project is not submitted by me in any other University or College.

B. SATYA PAVAN

ACKNOWLEGMENT

I consider this as a privilege to thank all those people who helped me a lot for successful completion of the project “A Cryptographic Algorithm Based on ASCII and Number System Conversions along with a Cyclic Mathematical Function”.

I would like to thank my project guide and our ever-accommodating **B.DIVAKAR , Assistant Professor**, who has obliged in responding to every request though she is busy with her hectic schedule of administration and teaching.

I would like to thank, **Prof. I. S. Pallavi, Director of MCA,** who had been a constant source of support and help during the tenure of completion of this project.

I would like to thank our ever-accommodating my project guide **Principal of Gayatri Vidya Parishad College for Degree and PG Courses (A), Prof. S. Rajani,** who has very obliged in responding to every request though he is busy with her hectic schedule of teaching.

I thank all the **Teaching and Non-teaching staff** who has been a constant source of support and encouragement during the study tenure.

B. SATYA PAVAN

A Cryptographic Algorithm Based on ASCII and Number System Conversions along with a Cyclic Mathematical Function

ABSTRACT

ABSTRACT

Data encryption and decryption in an efficient manner are the challenging aspects of modern information theory. In this algorithm, the plaintext to be encrypted is converted into unprintable characters. For encryption, a different technique is applied based on ASCII and number system conversions, which makes this algorithm different from others. First, each character of the plaintext and secret key is converted into its equivalent ASCII (decimal). Then, using some matrix manipulations on the decimal, representation of each character is transformed to 4 unprintable characters. After that, every unprintable character in the intermediate cipher text is further converted into a different unprintable character using a cyclic mathematical function. Performing three steps of processing, the final encrypted message is produced that gives higher level of security.

CONTENTS

1. INTRODUCTION

1.1Network security:

In an organization several computers are connected in network. This network may connect to internet. So, the message transmission is done between the computers. Here, we need the network security to protect our message transmissions. This is the criteria of providing network security.

Network security involves the authorization of access to data in a network, which is controlled by the network administrator. Users choose or assigned to username and password or other authenticating information that allows them access to information and programs within their authority. Network security covers a verity of computer networks, both public and private, that are used in everyday jobs conducting transactions and communications among business, government agencies and individuals. Networks can be private, such as within a company, and others which might be open to public access. Network security is involved in organizations, enterprises, and other types of institutions. It does as its title explains: It secures the network, as well as protecting and overseeing operations being done. The most common and simple way of protecting a network resource is by assigning it a unique name and a corresponding password.

The provision and policies adopted by a network administrator to prevent and monitor unauthorized access, misuse, modification, or denial of a computer network and network-accessible resources. Network security involves the authorization of access to data in a network, which is controlled by the network administrator. Uses can choose or are assigned and ID and password or other authenticating information that allows them access to information and program within their authority.

Network security covers a variety of computer networks, both public and private, that are used in everyday jobs conducting transaction and communications among business, government agencies and individuals. Network can be private, such as within a company, and others which might be open to public access. Network security is involved in organizations, enterprises, and other type of institutions.

The networks are computer networks, both public and private, that are used every day transactions and communications among businesses, government agencies and individuals. The networks are comprised of “nodes’, which are “client” terminals (individual user PCs), and one or more “servers” and/or “host” computers. They are linked by communication system, some of which might be private, such as within a company and others which might be open to public access. The obvious example of a network system that is open to public access is the Internet, but many private networks also utilize publicly-accessible communications.

Today, most companies host computers can be accessed by their employees whether in their offices over a private communications network, or from their homes or hotel rooms while on the road through normal telephone lines.

1.1.1. Principles in Network Security:

To provide adequate protection of network resources, the procedures and technologies that you deploy need to guarantee three things, sometimes referred to as the CIA triad: the following are some useful security services.

1.1.2. Confidentiality:

Confidentiality is roughly equivalent to privacy. Measures undertaken to ensure confidentiality are designed to prevent sensitive information from reaching the wrong people, while making sure that the right people can in fact get it: Access must be restricted to those authorized to view the data in question. It is common, as well, for data to be categorized according to the amount and type of damage that could be done should it fall into unintended hands. More or less stringent measures can then be implemented according to those categories.

Sometimes safeguarding data confidentiality may involve special training for that privy to such documents. Such training would typically include security risks that could threaten this information. Training can help familiarize authorized people with risk factors and password related best practices and information about social engineering methods, to prevent them from bending data handling rules with good intensions and potentially disastrous results.

1.1.3. Integrity:

Integrity involves maintaining the consistency, accuracy, and trustworthiness of data over its entire life cycle. Data must not be changed in transit, and steps must be taken to ensure that data cannot be altered by unauthorized people. These measures include file permissions and user access controls. Version control maybe used to prevent erroneous changes or accidental deletion by authorized users becoming a problem. In addition, some means must be in place to detect any changes in data that might occur as a result of non-human caused events such as an electromagnetic pulse or server crash. Some data might include checksums, even cryptographic checksums, for verification of integrity. Backups or redundancies must be available to restore that affected data to its state.

1.1.4. Availability:

Availability of information refers to ensuring that authorized parties can access the information when needed. Information only has value if the right people can access it at the right times. Denying access to information has become a very common attack nowadays. Almost every week you can find news about high profile website access to the resources of the website. Such downtime can be very costly. Other factors that could lead to lack of availability to valuable information may include accidents such as power outages or natural disasters such as floods.

1.1.5 Non-repudiation:

The creator/sender of the information cannot deny at alter stage his or her intentions in the creation or transmission of the information. In law, non-repudiation implies one's intention to fulfill their obligation to a contract.

It is important to note that while technology such as cryptographic system can assist in non- repudiation efforts, the concept is at its core legal concept transcending the realm of technology. It is not, for instance, sufficient to show that the message a digital signature signed with the sender's private key, and thus only the sender could have sent the message and nobody else could have altered it in transit. The alleged sender could in return demonstrate that the digital signature algorithm is vulnerable or flawed or prove that his signing key has been compromised.

1.1.6. Authentication:

The sender and receiver can confirm their identity and the origin/destination of the information. In computing, e-Business, and information security, it is necessary to ensure that the data, transaction, communication or documents (electronic or physical) are genuine. It is also important for authenticity to validate that both parties involved are who they claim to be. Some information security systems incorporate authentication features such as "digital signatures", which give evidence that the message data is genuine and was sent by someone possessing the proper signing key.

1.1.7. Network security tools:

Network security refers to the tools, technologies and processes that protect an organization's network and critical infrastructure from unauthorized use, cyberattacks, data loss and other security threats.

1.1.8. Antivirus software packages:

These packages counter most virus threats if regularly updated and correctly maintained.

1.1.9. Secure network infrastructure:

Switches and routers have hardware and software features that support secure connectivity, perimeter security, intrusion protection, identity services, and security management. Dedicated network security hardware and software-Tools such as firewalls and intrusion detection system provide protection for all areas of the network and enable secure connections.

Virtual private network: These networks provide access control and data encryption between two different computers on a network. This allows remote workers to connect to the network without the risk of a hacker or their activities and transactions on the network. Services include passwords, digital certificates, and digital authentication keys.

1.1.10. Security management:

This is the glue that holds together the other building blocks of a strong security solution none of these approaches alone will be sufficient to protect a network, but when they are layered together; they can be highly effective in keeping a network safe from attacks and other threats to security, In addition. Well-thought-out corporate policies are critical to determine and control access to various parts of the network.

1.2. Cryptography:

Cryptography was concerned totally with message encryption, i.e., the conversion of message from an intelligible form into unintelligible one and reverse again at the other end, rendering it unreadable by an unauthorized person without the knowledge of secret key (decryption key). In the modern age of technology cryptography is becoming a more and more central topic within computer science. As there is a need for more secure cryptographic schemes, the application of graph theory is going to in- crease for the development of secure encryption algorithms. R. Yadhu have proposed a selective encryption mechanism using message specific key and spanning tree concept of graph theory. The mechanism provides protection of privacy in communication as it avoids the formation of self-loops and parallel edges and key is exchanged only among the authenticated persons only. Graph theory has a great contribution in the development of various encryption techniques. In this paper we propose a scheme for secure communication using prime weighted graph.

Cryptography is the art and science of secure data communications over insecure channels. It is the study of method of sending messages in disguised form so that only the intended recipients can remove the disguise and interpret the message. Historically, the major consumers of cryptography were military and intelligence organizations. Today, however, cryptography is everywhere! Security mechanisms that rely on cryptography are an integral part of almost any computer system. Users rely on cryptography every time they access a secured website. Cryptographic methods are used to enforce access control in multi-user operating systems, and to prevent thieves from extracting trade secrets from stolen laptops. Software protection methods employ encryption, authentication, and other tools to prevent copying.

1.2.1. CRYPTOGRAPHIC SYSTEM

Cryptographic system is any computer system that involves cryptography. Such systems include for instance, a system for secure electronic mail which might include methods for digital signatures, cryptographic hash functions, key management techniques, and so on. Cryptographic systems are made up of cryptographic primitives, and are usually rather complex. Because of this, breaking a cryptosystem is not restricted to breaking the underlying cryptographic algorithms; usually it is far easier to break the system as a whole.

1.2.2. CRYPTOANALYSIS

Cryptanalysis refers to the art and science of analyzing information systems in order to study the hidden aspects of the systems. Cryptanalysis is used to breach cryptographic security systems and gain access to the contents of encrypted messages, even if the cryptographic key is unknown

1.2.3. BASIC SECURITY REQUIREMENTS:

To provide adequate protection of network resources, the procedures and technologies that you deploy need to guarantee three things, sometimes referred to as the CIA triad

Network Security Concerns Itself with The Following Four Objectives:

Confidentiality: Providing confidentiality of data, guarantees that only authorized users can view sensitive information. (The information cannot be understood by anyone for whom it was unintended)

Integrity: Providing integrity of data guarantees that only authorized users can change sensitive in- formation and provides a way to detect whether data has been tampered with during trars1nission; this might also guarantee the authenticity of data. (The information cannot be altered in storage or transit between sender and intended receiver without the alteration being detected)

Availability of systems and data: System and data availability provides uninterrupted access by authorized users to important computing resources and data.

Authentication: The sender and receiver can confirm each other's identity and the origin/destination of the infuriation.

2. LITERATURE SURVEY

2.1. Introduction:

Cryptography, Steganography and watermarking are three popular modern security offering techniques. Among them, the cryptography is older and mostly used technique as it is easy to implement and offers higher level of security. The cryptography is the study of mathematical techniques related to the aspects of information security such as confidentiality, data integrity, entity authentication and data origin authentication. Cryptography historically dealt with the construction and analysis of protocols that would prevent any third parties from reading a private communication between two parties. In the digital age, cryptography has evolved to address the encryption and decryption of private communications through the Internet and computer systems. Until modern times, cryptography referred almost exclusively to encryption, which is the process of converting original information (called plaintext) into unintelligible text (called cipher text). On the other hand, decryption is the reverse procedure that moves from the unintelligible cipher text back to plaintext.

2.2. History:

Information security begins with computer security. The main aim of data hiding technologies is to solve modern network security, quality of services control, and secure communications. Over years there are many security approaches have been developed to provide information security. Although these approaches help in managing security, there is a need for information security ap- proaches to provide a holistic modeling support which can be integrated into modern information development approaches Most of the modern approaches to information security employ cryptography.

In proposed system encryption and decryption is done by using asymmetric key. Key is generated using new Pythagorean triple algorithm. This key is used in encryption and decryption of information.

2.3. Current System:

In Existing system there are many cryptographic algorithms are introduced. One of the earliest and most used mechanisms in cryptography is Caesar's cipher which is also called a shift cipher. It is a replacement mechanism in which each

letter of the plain text is replaced by another letter which is certain places ahead of the letter and the process is repeated for all the letters in the plain text. The number of places ahead to be used is the key to the encryption. For example, if the key is 2, then a will be replaced by c which is two places ahead . In this project, a new cryptographic algorithm is proposed which involves ASCII, number system conversions and the cyclic mathematical used in.

2.4. Proposed System:

Day by day the level of security is going to be higher. Still now many researchers are working on cryptography and data hiding. A new cryptographic algorithm for the Real Time Application was in to improve the time for encryption and decryption of data of end-to-end delay and to provide higher level of security. In this paper, we proposed an improved algorithm which is different from the traditional symmetric-key cryptography, asymmetric-key cryptography or hashing function. A cryptographic algorithm based on ASCII conversion and a cyclic mathematical function was presented in, and which makes the cipher different from other algorithms.

2.5. REQUIREMENT ELICITATION:

Requirement is the feature the system must have a constraint that it must satisfy to be accepted by the clients. Requirements engineering aims at defining the requirements of the system under construction. It includes two main activities namely Requirements Elicitation and Analysis.

Requirements Elicitation focuses on describing the process of the system. The client, the developer and the users identify the problem. Such a definition is called Requirement Specification. This specification is structured and formulized during analysis to produce an Analysis Model. Requirements Elicitation and Analysis focuses only on the user's view of the system. Requirements Elicitation includes the following activities:

Identifying Actors:

During this activity, developers identify the different types of users the future system will support.

Identifying Scenarios:

During this activity, developers observe users and develop a set if detailed scenarios for typical functionality provided by the future system. Developers use these scenarios to communicate with the users and deepen their understanding.

Identifying Use Cases:

Once developers and users agree on the set of scenarios, a set of use cases that completely represent the future system.

Refining Use Cases:

During this activity, developers ensure that the requirements specifications completed by detailing each use case and describing the behavior of the system in the presence of errors and exceptional conditions.

Identifying Relationships among Use Cases:

During this activity, developers identify the dependencies among the use cases and also consolidate the use case model by factoring out common functionality.

Identifying Non-functional Requirements:

During this activity, developers, users and clients agree on aspects like performance of system, documentation, resources, security and its quality.

2.6. REQUIREMENT ELICITATION AND ANALYSIS:

2.6.1. Functional Requirements:

It describes the interactions between the system and its environment independent of its implementation.

The functional requirements are:

Encryption:

Input: Plaintext,Key

Output: Cipher text.

Decryption:

Input: Cipher text, Key

Output: Plain text

2.6.2. NON-FUNCTIONAL REQUIREMENTS:

Constraints on the services or functions offered by the system such as timing constraints, constraints on the development process, standards, etc. During this activity, developers, users and clients agree on aspects like performance of system, documentation, resources, security and its quality.

- ***Usability*:** The GUI of this system provides easy access to the user and user can get best results for the given input.
- ***Supportability*:** System is implemented using Python 3.8.3 IDLE
- ***Performance***: It display the correct result based on the output for all possible correct inputs at all times when the computer is improper condition
- ***Reliability*:** This system will perform its intended function adequately for a specific period of time..
- ***Security:*** This system provides security by a new approach to encryption using rail fence coding
- ***Accuracy: Julius Caesar's*** hard to decrypt the message.

- ***Portability:*** This system can work on windows operating system environment with minimal changes.

2.6.3. HARDWARE REQUIREMENTS:

The Minimum hardware requirements to run this system are:

1. **Processor** : Intel core i3 and above
2. **Ram** : 4.00GB
3. **System Type** : 64/32-bit Operating System
4. **Hard Disk** : 500GB
5. **Monitor** : Standard
6. **Keyboard** : Standard
7. **Mouse** : Standard

2.6.4. SOFTWAREREQUIREMENTS:

The Minimum software requirements to run this system are:

1. **Front end Design** **:** Python 3.10.3 IDLE
2. **Operating System** **:** Windows 10
3. **Back end Design** **:**Python Programming Language

2.7. INTRODUCTION TO PYTHON

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985- 1990. Like Perl, Python source code is also available under the GNU General Public License (GPL). This tutorial gives enough understanding on Python programming language.

2.7.1. Advantages of Python:

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

Python is a MUST for students and working professionals to become a great Software Engineer especially when they are working in Web Development Domain. I will list down some of the key advantages of learning Python:

- **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

2.8. Characteristics of Python

Following are important characteristics of Python Programming −

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large ap- plications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

2.8.1. New Approach for building window Software

The Python Framework simplifies Windows development. It provides developers with a single approach to build both desktop applications sometimes called smart client applications and Web- Based applications. It also developers to use the same tools and skills to develop software for a verity of system ranging from handled smart phones to large server installations.

2.8.2. Applications of Python

As mentioned before, Python is one of the most widely used language over the web. I'm going to list few of them here:

- **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
- **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** − Python can run on a wide variety of hardware platforms and has the same inter- face on all platforms.
- **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** − Python provides interfaces to all major commercial databases.
- **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** − Python provides a better structure and support for large programs than shell scripting.

2.8.3. Python - GUI Programming (Tkinter):

Python provides various options for developing graphical user interfaces (GUIs). Most important are listed below.

- **Tkinter** − Tkinter is the Python interface to the Tk GUI toolkit shipped with Python. We would look this option in this chapter.
- **wxPython** − This is an open-source Python interface for wxWindows http://wxpython.org.
- **JPython** − JPython is a Python port for Java which gives Python scripts seamless access to Java class libraries on the local machine http://www.jython.org.

There are many other interfaces available, which you can find them on the net.

2.9. Tkinter Programming

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps −

- Import the Tkinter module.
- Create the GUI application main window.
- Add one or more of the above-mentioned widgets to the GUI application.
- Enter the main event loop to take action against each event triggered by the user.

3.UML MODELING

3.1. Introduction to UML

UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software system. UML was created by the Object Management Group (OMG) and UML

1.0 specification drift was proposed to the OMG in January 1997.

OMG is continuously making efforts to create a truly industry standard.

- UML stands for Unified Modeling Language.
- UML is different from the other common programming language such as C++, java, COBOL, etc.
- UML is a pictorial language used to make software blueprints.
- UML can be described as a general purpose visual modeling language to visualize, specify, construct and document software system.
- Although UML is generally used to model software system, it is not limited within this boundary. It is generally used to model software system as well. For example, the process flows in a manufacturing unit, etc.

UML is not a programming language, but tools can be used to generate code in various language using UML diagrams. UML has a direct relation with object-oriented analysis and design. After some standardization, UML has become an OMG standard.

3.2. Goals of UML

A picture is worth a thousand words, this idiom absolutely fits describing UML. Object- oriented concepts were introduced much earlier than UML. At that point of time, there were no standard methodologies to organize and consolidate the Object-oriented development. It was then that UML came into picture.

There are number of goals for developing UML but the most important is to define some general-purpose modeling language, which all models can use and it also need to be made simple to understand and use.

UML diagram are not only made for developers but also for business users, common people, and anybody interested to understand the system. The system can be a software or non-software system. Thus, it must be clear that UML is not a development method rather it accompanies with processes to make it a successful system. In conclusion, the goal of UML can be defined as a simple modeling mechanism to model all possible practical system in today's complex environment.

3.3 UML standard diagrams:

The elements are like components which can be associated in diverse ways to make a complete UML picture, which is known as diagram. Thus, it is very important to understand the different diagrams to implement the knowledge in real-life system. Any complex system is best understood by making some kind of diagrams or pictures. These diagrams have a better impact on our understanding. If we look around, we will realize that the diagram is not a new concept but it is used widely in different forms in different industries. We prepare UML diagram to understand the system in a better and simple way. A single diagram is not enough to cover all the aspects of the system. UML defines various kinds of diagrams to cover most of the aspects of a system. You can also create your own set of diagrams to meet your requirements. Diagrams are generally made in an incremental and iterative way. There are two broad categories of diagram and they are again divided into subcategories:

- Structural Diagrams
- Behavioral Diagrams

3.3.1. Structural Diagrams

The structural diagram represents the static aspect of the system. These static aspects represent those parts of a diagram, which forms the main structure and are therefore stable. These static parts are represented by classes, interfaces, object, components, and nodes. The four structural dia- grams are:

- Class diagram
- Object diagram
- Component diagram
- Deployment diagram

3.3.2. Behavioral Diagrams

Any system can have two aspects, static and dynamic. So, a model is considered as complete when both the aspects are fully covered. Behavioral diagram captures the dynamic aspect of a system. Dynamic aspect can be further described as the changing/moving parts of a system. UML has the following five types of behavioral diagrams:

- Use case diagram
- Sequence diagram
- Collaboration diagram
- State chart diagram
- Activity diagram

3.4. UML Diagrams: The UML diagram is a general-purpose way to visualize the main concepts of object-oriented concepts used in the software.

3.5. Use Case Diagram

3.5.1. Introduction Use Case Diagram:

Use case describes the behavior of the system as seen from the actor's point of view. A use case diagram can portray then different types of users of a system and the many ways that they interact with system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well. Actors initiate the use cases for accessing system's functionality. When actors and use cases exchange information, they are said to Communicate. To describe a use case, we use a template composed of six fields:

Use Case Name **:** The name of the use case.

Participating Actors **:** The actors participating in the particular use case..

Entry Condition **:** Condition for initiating the use case.

Flow of events **:** Sequence of steps describing the

functioning of Use case.

Exit Condition **:** Condition for terminating the use case.

Quality Requirements **:** Requirements that do not belong to the use case but constraint the functionality of the system

Actors:

Actors are external entities that interact with the system. Actors typically include a user role or another system. They have a unique names and descriptions.

Figure 3.5.1: Actor

In this project the two actors are Sender and Receiver:

Actor	**Roles**
Sender --	Enter/browse plain text --Generate cipher text
Receiver	Receive cipher text --Generate plain text

3.5.2. Use Case Diagram

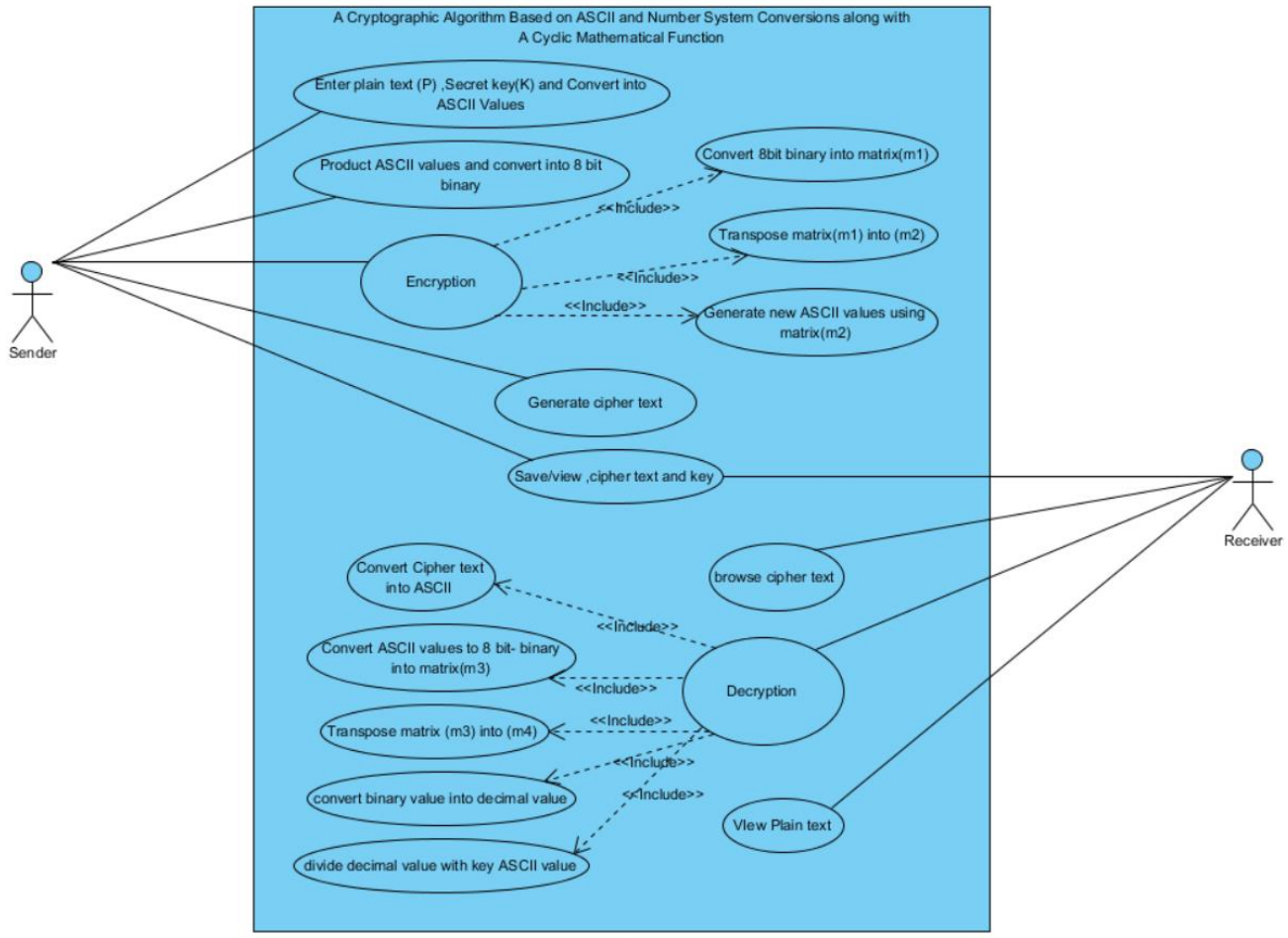


Figure: 3.5.2.1 Use case diagram (Sender and Receiver)

3.5.3. Description:

In this use case diagram, the sender will enter the plain text then encrypt with the key generated using Pythagorean triple algorithm then the cipher text is obtained. Now the receiver will choose the cipher text then decrypt by using the key generated at the sender side. Now plain text is generated at receiver side.

3.5.3.1. Use case for upload plain text

Use case Name	Sender and receiver
Participating Actors	Sender and receiver
Entry Condition	Sender Enter the plain text
Flow of Events	1. Enter the plain text 2. Display the plain text on the screen
Exit Condition	Display the plain text on the screen.

Table 3.5.3.1**. Sender Use case**

3.5.3.2. Use case for Key Generation

Use case Name	Key Generation
Participating Actors	Sender
Entry Condition	First value should be in equal length second value
Flow of Events	Calculating key value using the formulas
Exit Condition	Then key will be generated

Table 3.5.3.2. **Key generation Use case**

3.5.3.3. Use case for Encryption

Use case Name	Encryption
Participating Actors	Sender
Entry Condition	Enter the plain text
Flow of Events	1. Enter the plain text. 2. Convert plain text into numbers.

	3. Encrypt the message by pressing the encrypt button. 4. Save the Cipher text to the Receiver.
Exit Condition	Obtain the cipher text

Table 3.5.3.3 **Encryption Use case**

3.5.3.4. Use case for Decryption

Use case Name	Decryption
Participating Actors	Receiver
Entry Condition	Select the cipher text
Flow of Events	1. Select cipher text. 2. Convert cipher text into numbers. 3. Decrypt the cipher text by pressing the button. 4. We will obtain the plain text.
Exit Condition	Obtain the plain text

Table 3.5.3.4. **Decryption Use case**

3.6. Scenarios

A Scenario is an instance of a use case describing a concrete ser of actions. Scenarios are used as examples for illustration common cases: their focus is on understanding ability. To describe a scenario, we use a template composed of three fields:

Scenario Name : Name of the Scenario

Participating Actors : Instance of the actors participating in the Scenario

Flow of Events : Sequence of steps describing the event in scenario

Scenario Name	Encryption

Participating Actors	Sender

3.6.1. Encryption Scenario:

Flow of Events	1. Enter the plain text. 2. Plain text is converted into cipher text. 3. Send the cipher text to the receiver.

Table 3.6.1.1. Encryption Scenario

3.6.2. Decryption Scenario:

Scenario Name	Decryption
Participating Actors	Receiver
Flow of Events	1. Receiver will decrypt the cipher text. 2. Used the key send by the sender. 3. Get the plain text.

Table 3.6.2.1. Decryption Scenario

3.7. Sequence Diagram:

Interaction between object can be described by means of sequence diagrams. An object inter- acts with another object by sending messages. The reception of a message by an object triggers the execution of an operation, which in turn may send messages to other objects. Arguments may be passed along with a message and are bound to the parameters of the executing operation in the receiving object.

3.7.1. Sequence diagram for sender:

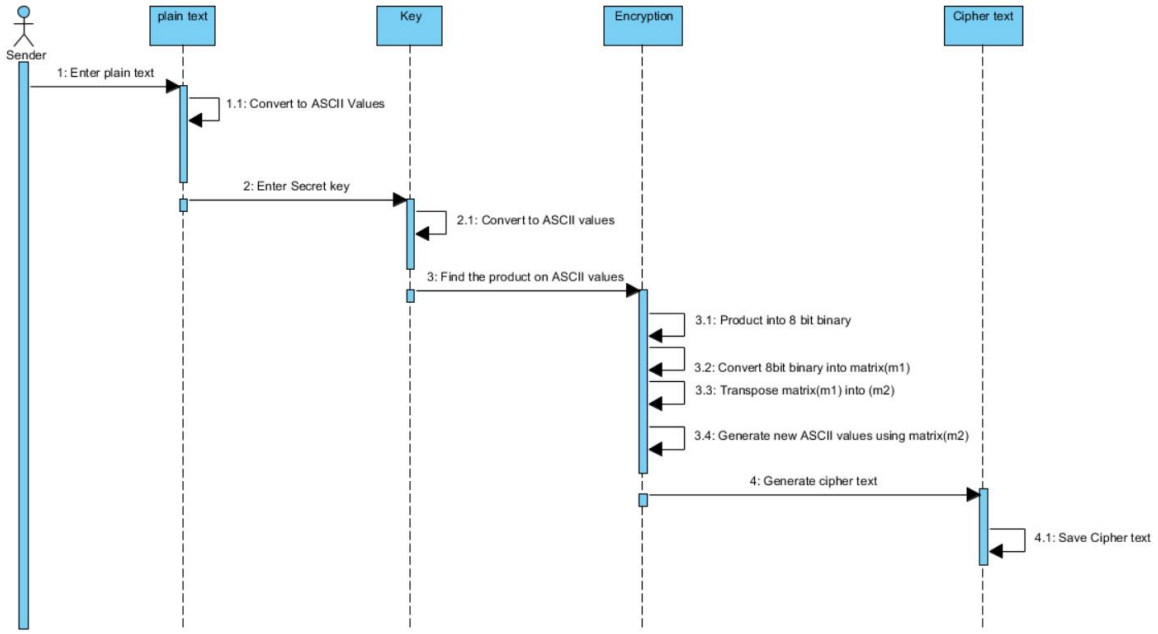


Figure 3.7.1.1 Senders Sequence Diagram

3.7.2. Description:

Sender enters the plain text and generates P and Q values for key generation. The Sender will encrypt the text by using the encryption algorithm and key. Save the cipher text, and the saved cipher text and key will be given to the receiver for decryption. Sender will exit from the system.

3.7.3. Sequence diagram for receiver:

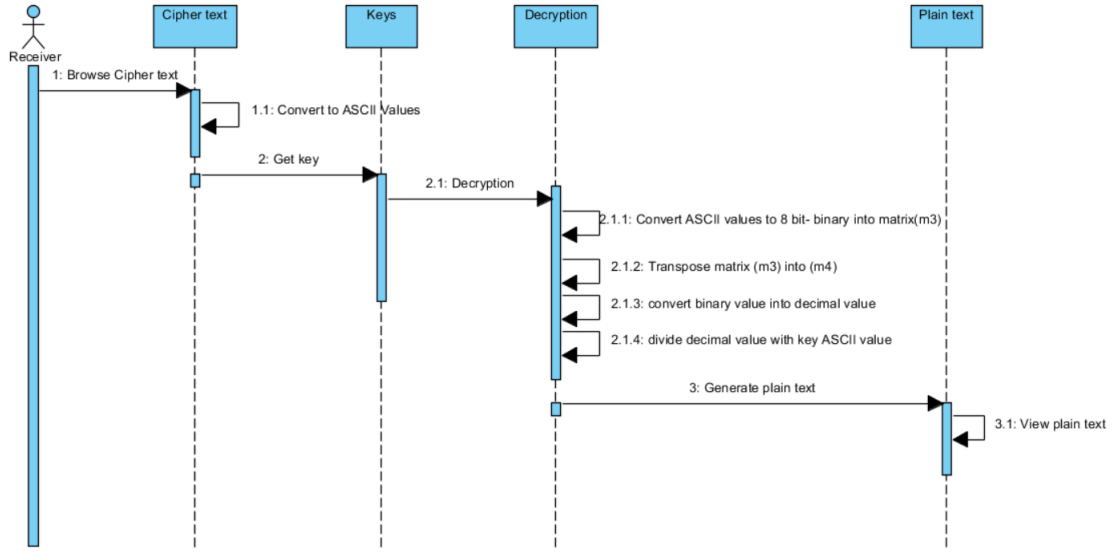


Figure 3.7.3.1 Receivers Sequence Diagram

3.7.4. Description:

The receiver gets the cipher text and key which are generated in encryption process. Then decrypts the cipher text using the key to get the plain text. The decryption process is a reverse process of the encryption process.

3.8. State-Chart Diagram:

State diagrams are used to describe the behavior of a system. State diagram describe all the possible state of an object as events occur. Each diagram usually represents objects of a single class and track the different state of its objects through the system. Not all classes will require a state diagram and state diagram are not useful for describing the collaboration of all objects in a use case. State diagram have very few elements.This is the state of the object when it is created. After the initial state the object begins changing states. State chart diagrams are also used for forward and reverse engineering of a system. But the main purpose is to model reactive system.

3.8.1. State-Chart Diagram:

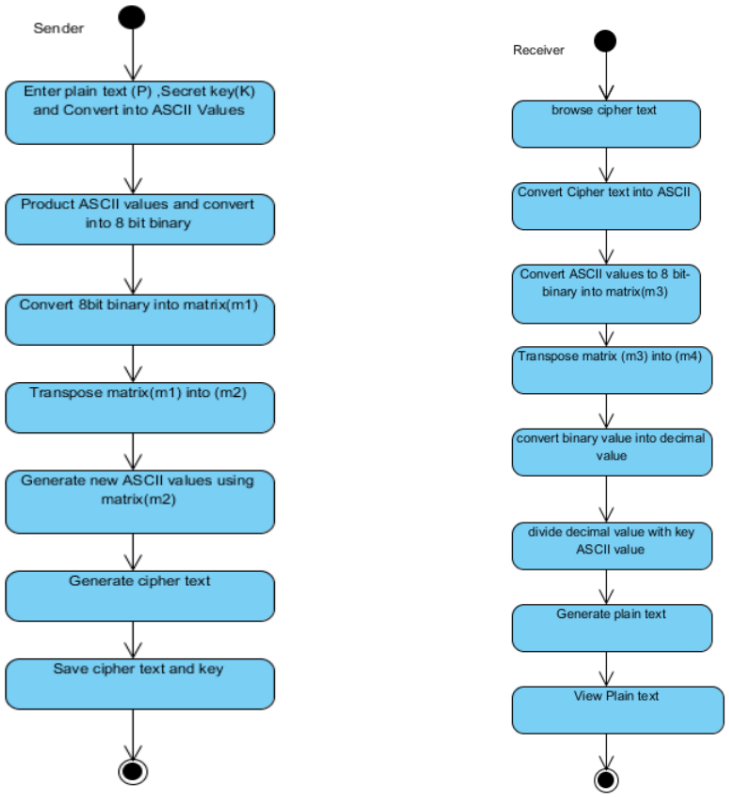


Figure 3.8.1.1. State-Chart Diagram

3.8.2. Description:

First Sender type the plain text from the required location. Sender generates the key. Using this key Sender encrypts the plain text to get cipher text. Then send the cipher text and key to the receiver. Now receiver selects the cipher and key to decrypt cipher text into recovered plain text.

4. DESIGN

4.1. Design and Description of the Algorithm:

Encryption is the process of transforming plaintext into the cipher text where plain text is the input to the encryption process and cipher text is the output of the encryption process. Decryption is the process of transforming cipher text into the plain text where cipher text is the input to the decryption and plain text is the output of the decryption process. There are various encryption algorithms exit classified as symmetric and asymmetric encryption algorithms. Here, in this project using an algorithm for data encryption and decryption which is based on ASCII of characters in the plaintext. This algorithm is used to encrypt data by using ASCII values of the data to be encrypted. Few values are created using Pythagorean triple algorithm, which are used as a key to encrypt and decrypt the data. So, it can be said it is a kind of symmetric encryption algorithm because it uses same key for encryption and decryption.

This system generates key based on the given P and Q values (P>Q), by using these values Pythagorean triples are generated. These triples are used as key for encryption and decryption, this is symmetric cryptographic algorithm.

4.2. Encryption and Decryption

Data that can be read and understood by anyone without any special knowledge about it is called plaintext or clear text. The method of disguising the plaintext in such a way as to hide the information is called encryption. Encrypting plaintext results in unreadable gibberish called cipher text. You use encryption to ensure that information is hidden from anyone for whom it is not intended, even those who can see the encrypted data. The process of reverting cipher text to its original plaintext is called decryption.

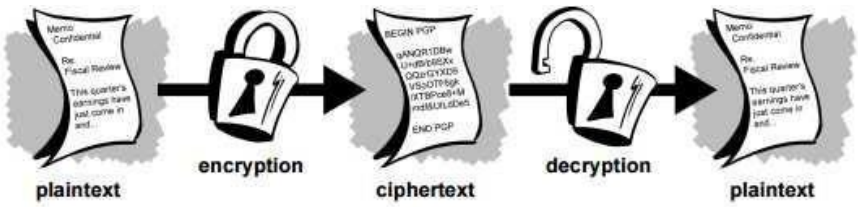


Fig. 4.2.1. Encryption and Decryption

4.3. Conventional Cryptography

In conventional cryptography, also called secret-key or symmetric-key encryption, one key is used to both encrypt and decrypt the data. The famous Caesar's Cipher is an example of this technique. Only the person who knew the "shift by 3" rule [2] could understand the message. While sending the encrypted data, the key is shared through another secure channel so as to make it possible for the receiver to decrypt the cipher text into original plaintext. Which makes it a bit insecure?

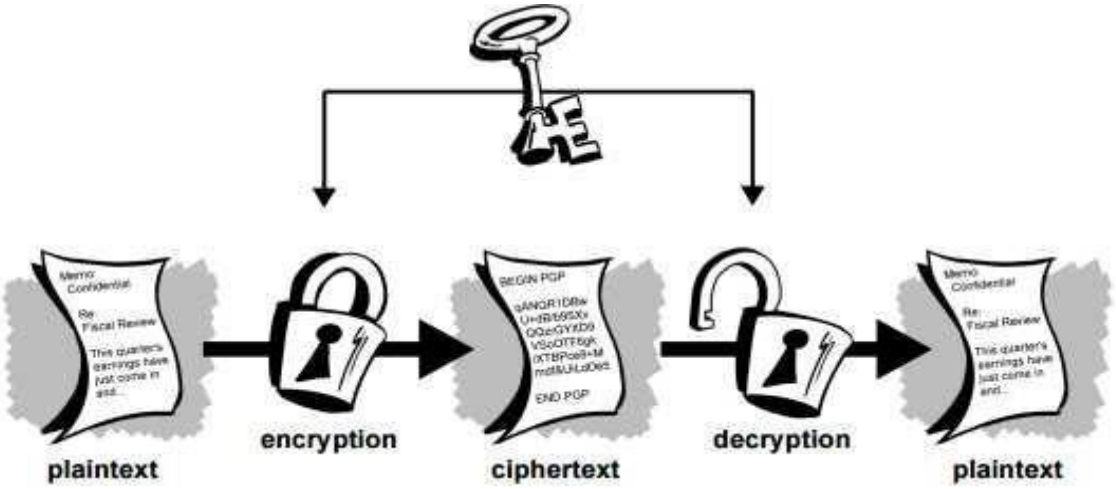


Fig. 4.3.1.. Conventional Cryptography

4.4. Public Key Cryptography

In this technique, there are two keys involved, one for encryption and other for decryption. The receiver already has a private key which is never used in any

communications. Only the public key is sent along with the cipher text. No one without having the private key can decrypt the code, thus making the communication safer.

4.5. Public Key Cryptography

In this technique, there are two keys involved, one for encryption and other for decryption. The receiver already has a private key which is never used in any communications. Only the public key is sent along with the cipher text. No one without having the private key can decrypt the code, thus making the communication safer.

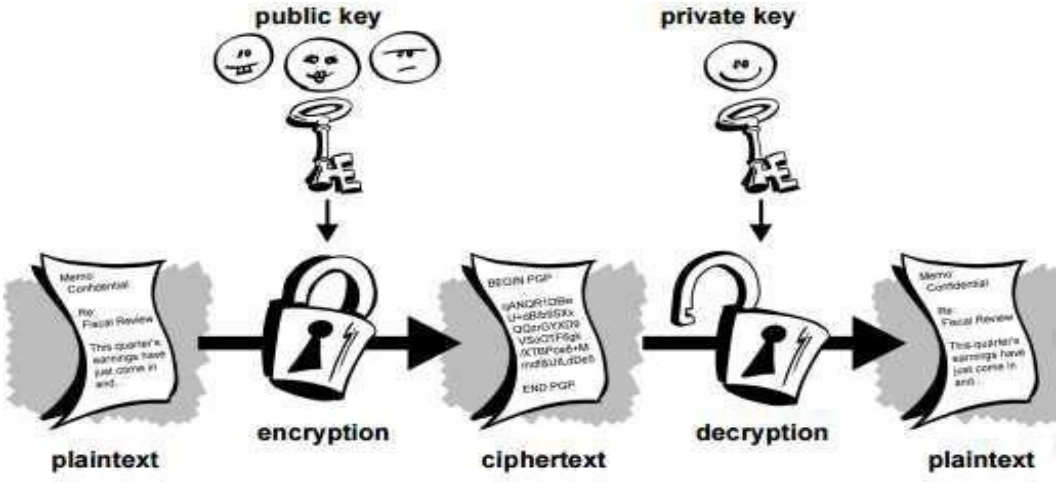


Fig. 4.4.1. Public Key Cryptography

4.6. THE PROPOSED APPROACH

4.6.1. Algorithm for encryption is as follows:

Steps in Algorithm

1. Input message.
2. Convert to ASCII values
3. Input Key1 message
4. Convert to ASCII values
5. Find the product
6. Convert the product into binary and put the bits on a 4×4 matrix from last to first.

7. Read the bits from each column and make decimal.
8. We get four decimal values in New_ASCII array.
9. Input Key2 value (m)
10. calculate Final_ASCII [i] = (New_ASCII +m) % 32, where,0<m<32
11. Convert the Final_ASCII[i] into its equivalent character.
12. End encryption.

4.6.2. <u>Algorithm for decryption is as follows:</u>

1) Browse Cipher text
2) Convert to values
3) Browse Key2 value
4) Calculate the formula

Final ASCII=(A-K+32) mod 32

5) Each value is convert to binary
6) Then all the binary values arrange in 4x4 matrix
7) Transpose the matrix
8) Each 4×4 matrix is convert to binary value
    - And Key1 from Sender is PAVAN
9) Key is Convert to ASCII values
10) Then Divide each Decimal value by Key values
11) Then values are Plain Text

4.7. Encryption Example

Mapping Table

0	1	2	3	4	5	6	7
!	‘	#	$	%	&	(	)
8	9	10	11	12	13	14	15
*	+	/	;	<	=	>	?
16	17	18	19	20	21	22	23
@	[	\	]	^	{	|	}
24	25	26	27	28	29	30	31
~	×	–	Ø	©	®	±	µ

Let’s take a message

“SATYA” And Key is

“PAVAN”

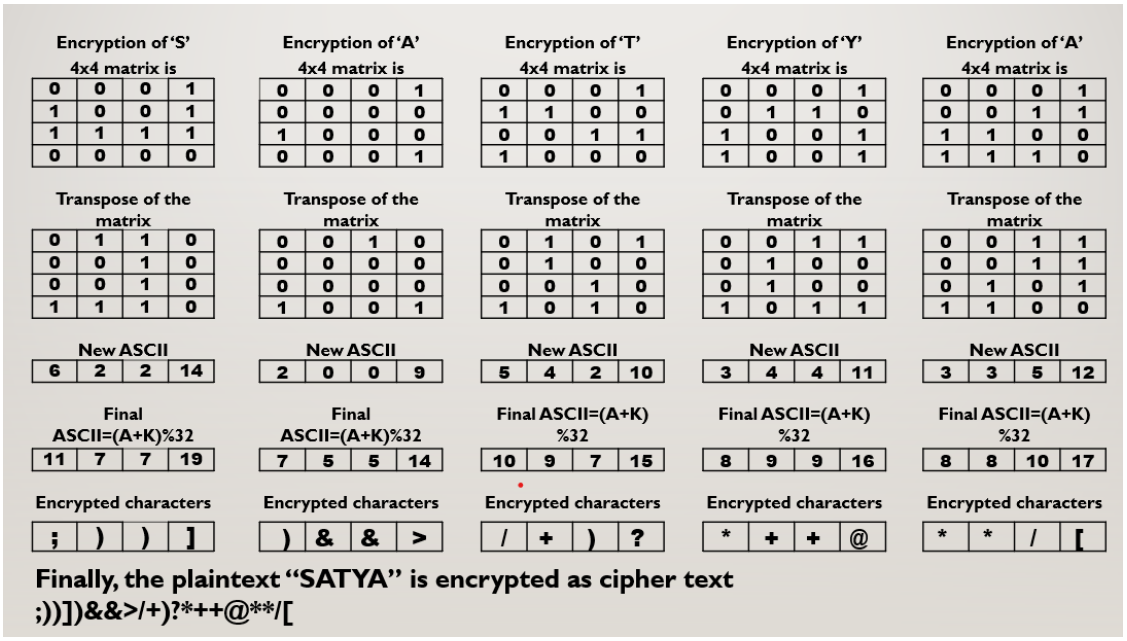
Both are converted to ASCII

values After Calculate Product

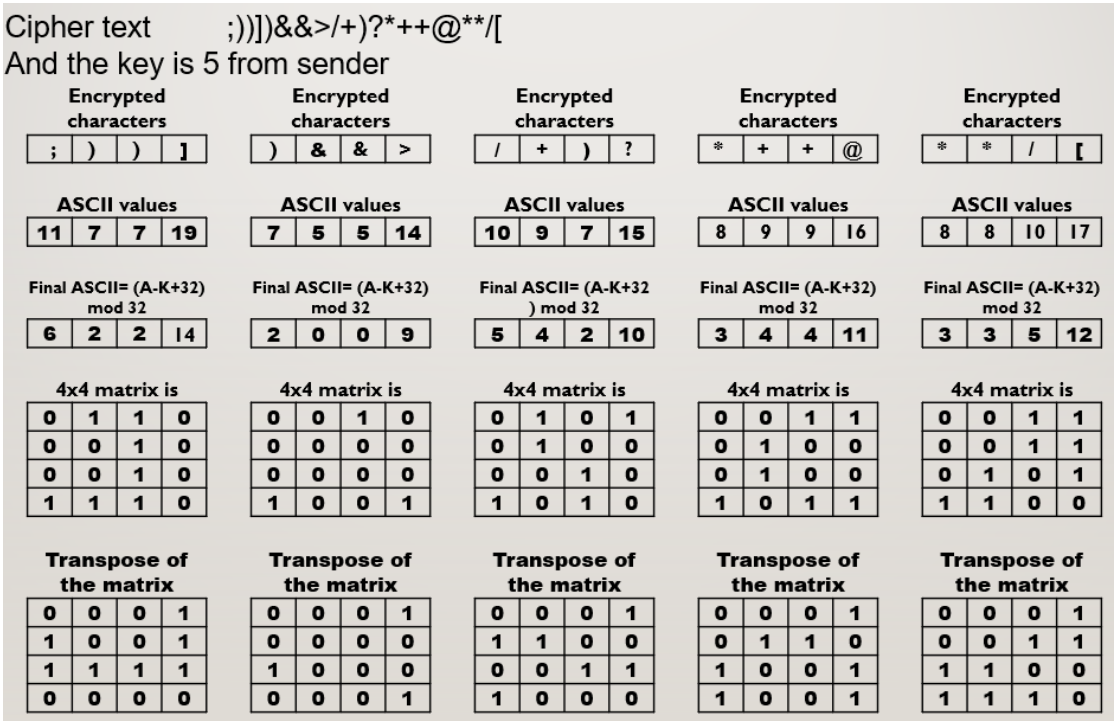
Then Convert to Binary values

Plain Text	ASCII	Key1	ASCII	Product	Binary
S	83	P	80	6640	1 1001 1111 0000
A	65	A	65	4225	1 0000 1000 0001
T	84	V	86	7224	1 1100 0011 1000
Y	89	A	65	5785	1 0110 1001 1001
A	65	N	78	5070	1 0011 1100 1110

Table 4.7.1. Encryption Example



4.3. Decryption Example



Each 4×4 matrix is convert to binary value

And Key1 from Sender is PAVAN

Key is Convert to ASCII values

Then Divide each Decimal value by Key values

Binary	Convert to Decimal	Key1	ASCII	Decimal ASCII	Plain Text
1 1001 1111 0000	6640	P	80	83	S
1 0000 1000 0001	4225	A	65	65	A
1 1100 0011 1000	7224	V	86	84	T
1 0110 1001 1001	5785	A	65	89	Y
1 0011 1100 1110	5070	N	78	65	A

Table 4.7.2 Decryption Example

The decryption procedure of the other characters from the cipher text is "SATYA"

5. CODING

The goal of coding or programming phase is to translate the design of the system produced during the phase into code in a given programming language, which can be executed by a computer and the performs the computation specified by the design.

The coding phase affects both testing and maintenance. The goal of coding is not to re- duce the implementation cost, but the goal should be to reduce the cost of later phase. In other words, the goal is not to simplify the job of programmer. Rather the goal should be to simplify the job of the tester and maintainer.

5.1. Coding Approach:

There are two major approaches for coding any software system. They are top-Down approach and bottom up approach.

Bottom-up approach can suit for developing the object-oriented systems. During system design phase of reduce the complexity. We decompose the system into appropriate number of subsystems, for which objects can be modelled independently. These objects exhibit the way the subsystems perform their operations.

Once object have been modeled they are implemented by means of coding. Even though related to the same system as the objects are implemented of each other the Bottom-Up approach is more suitable for coding these objects.

In this approach, we first do the coding of objects independently and then we integrate these modules into one system to which they belong. In this project, top-Down approach is followed. For registration and Login. User will clicks and stores the intruder information intensely.

5.2. Information Handling:

Any software system requires some amount of information during its operation selection of appropriate data structures can help us to produce the code so that objects of the system can better operate with the available information decreased complexity.

In this project, Encryption and decryption will not be possible if the image fields are vacant. System will not have any default values. User must specify each secret file name in encryption and locate all required operations in decryption.

5.3. Programming Style:

Programming style deals with act of rules that a programmer must follow so that the characteristics of coding such as Traceability, Understands the ability, Modifiability, and Extensibility can be satisfied. In the current system, we followed the coding rules for naming the variables and methods. The system is developed in a very interactive and users friendly manner.

5.4. Verification and Validation:

Verification is the process of checking the product built is right. Validation is the process of checking whether the right product is built. During the Development of the system coding for the object has been thoroughly verified from various aspects regarding their design, in the way they are integrated and etc. The various techniques that have been followed for validation discussed in testing the current system. Validations applied to the entire system at two levels:

Form level validation:

Validations of all the inputs given to the system at various points in the forms are validated while navigating to the next form. System raises appropriate custom and predefined exceptions to alert the user about the errors occurred or likely to occur.

Field level validation:

Validations at the level of individual controls are also applied whenever necessary. System pops up appropriate and sensuous dialogs whenever necessary.

5.5. Implementation of the system

```
from tkinter import *
from PIL import ImageTk,Image
from tkinter import messagebox
from tkinter.filedialog import asksaveasfile
from tkinter.filedialog import askopenfile
import random

def raise_frame(f):
    f.tkraise()

def finish():
    r.destroy()

def fr1():
    raise_frame(Fr1)
def fr2():
    raise_frame(Fr2)
def fr3():
    raise_frame(Fr3)
def fr4():
    raise_frame(Fr4)
def fr5():
    raise_frame(Fr5)
def fr6():
    raise_frame(Fr6)
def fr7():
    raise_frame(Fr7)
def fr8():
    raise_frame(Fr8)
def fr9():
    raise_frame(Fr9)
```

```
def fr10():
    raise_frame(Fr10)
def fr11():
    raise_frame(Fr11)
def fr12():
    raise_frame(Fr12)

#Encryption Calculations--------------------------------------------------------------------
tik=0
def fun1():
    global values1,tik
    values1 = ''
    if(tik==0):
        tik=1
        t1=e1.get()
        n=len(t1)
        if(n!=0):

            for i in range(len(t1)):
                z2=ord(t1[i])
                values1+=str(z2)+' '

        v1.set(values1)
    else:
        messagebox.showinfo('MessageBox','Please enter plain text  ')
def fun2():
    global values2,tik
    values2 = ''
    if(tik==1):
        tik=2
        t2=e2.get()
        n=len(t2)
        if(n!=0):
            for i in range(len(t2)):
```

```
            z2=ord(t2[i])
            values2+=str(z2)+' '
        v2.set(values2)
    else:
        messagebox.showinfo('MessageBox','Please enter key value ')
def fun3():
    global values1,values2,dec,tik
    if(tik==2):
        tik=3
        values1=values1.split(' ')
        values1=values1[:-1]
        values2=values2.split(' ')
        values2=values2[:-1]
        a=values1
        b=values2
        dec=''
        for i in range(len(a)):
            dec+= str(int(a[i])*int(b[i]))+' '
        v3.set(dec)
    else:
        messagebox.showinfo('MessageBox','Please hit the previous button ')
def D8B(a):
    bnr = bin(a).replace('0b','')
    x = bnr[::-1]
    while len(x) < 16:
        x += '0'
    bnr = x[::-1]
    return bnr
def fun4():
    global dec,binary1,tik
    if(tik==3):
        tik=4
        dec=dec.split(' ')
        dec=dec[:-1]
```

```
        binary1= ''
        for i in range(len(dec)):
            binary1+=str(D8B(int(dec[i])))+' '
        v4.set(binary1)
    else:
        messagebox.showinfo('MessageBox','Please hit the previous button ')
#====================Encryption 2 page Calculations ==========#
def MATRIX(a,x1,y1,n):

    a1=''
    for i in range(len(a)):
        a1+=a[i]+' '
        if((i+1)%4==0):
            a1+='\n'
    a1=a1[:-1]
    if((n+1)==1):
        v5.set(a1)
        l3= Label(Fr7,textvariable= v5, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==2):
        v6.set(a1)
        l3= Label(Fr7,textvariable= v6, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==3):
        v7.set(a1)
        l3= Label(Fr7,textvariable= v7, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==4):
        v8.set(a1)
        l3= Label(Fr7,textvariable= v8, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==5):
        v9.set(a1)
        l3= Label(Fr7,textvariable= v9, fg="black",bg="white",anchor="w", font =
```

```
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==6):
        v10.set(a1)
        l3= Label(Fr7,textvariable= v10, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==7):
        v11.set(a1)
        l3= Label(Fr7,textvariable= v11, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==8):
        v12.set(a1)
        l3= Label(Fr7,textvariable= v12, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==9):
        v13.set(a1)
        l3= Label(Fr7,textvariable= v13, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==10):
        v14.set(a1)
        l3= Label(Fr7,textvariable= v14, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)

def MATRIXD(a,x1,y1,n):

    a1="
    for i in range(len(a)):
        a1+=a[i]+' '
        if((i+1)%4==0):
            a1+='\n'
    a1=a1[:-1]
    if((n+1)==1):
        v5.set(a1)
        l3= Label(Fr10,textvariable= v5, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
```

```
    if((n+1)==2):
        v6.set(a1)
        l3= Label(Fr10,textvariable= v6, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==3):
        v7.set(a1)
        l3= Label(Fr10,textvariable= v7, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==4):
        v8.set(a1)
        l3= Label(Fr10,textvariable= v8, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==5):
        v9.set(a1)
        l3= Label(Fr10,textvariable= v9, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==6):
        v10.set(a1)
        l3= Label(Fr10,textvariable= v10, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==7):
        v11.set(a1)
        l3= Label(Fr10,textvariable= v11, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==8):
        v12.set(a1)
        l3= Label(Fr10,textvariable= v12, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==9):
        v13.set(a1)
        l3= Label(Fr10,textvariable= v13, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==10):
        v14.set(a1)
```

```
    l3= Label(Fr10,textvariable= v14, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)

def MT(a):
    t="

    a1=a[0]
    a2=a[1]
    a3=a[2]
    a4=a[3]
    a1=a1.replace(' ','')
    a2=a2.replace(' ','')
    a3=a3.replace(' ','')
    a4=a4.replace(' ','')


    for j in range(4):
        t+=str(a1[j]+a2[j]+a3[j]+a4[j])+'\n'


    #print(t)
    return  t

def B2D(binary):
    binary1 = binary
    decimal, i, n = 0, 0, 0
    while(binary != 0):
        dec = binary % 10
        decimal = decimal + dec * pow(2, i)
        binary = binary//10
        i += 1
    return(decimal)
D="
def TRANS(a,x1,y1,n):
```

```
    global D
    a1=''
    for i in range(len(a)):
        a1+=a[i]
        if((i+1)%4==0):
            a1+='\n'
    a1=a1[:-1]

    a=a1.split('\n')

    a1=MT(a)
    a1=a1[:-1]

    if((n+1)==1):
        v15.set(a1)
        A1=a1.split('\n')

        D+=str(B2D(int(A1[0])))+' '
        D+=str(B2D(int(A1[1])))+' '
        D+=str(B2D(int(A1[2])))+' '
        D+=str(B2D(int(A1[3])))+' '

        l3= Label(Fr7,textvariable= v15, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==2):
        v16.set(a1)
        A1=a1.split('\n')

        D+=str(B2D(int(A1[0])))+' '
        D+=str(B2D(int(A1[1])))+' '
        D+=str(B2D(int(A1[2])))+' '
        D+=str(B2D(int(A1[3])))+' '

        l3= Label(Fr7,textvariable= v16, fg="black",bg="white",anchor="w", font =
```

```
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==3):
        v17.set(a1)
        A1=a1.split('\n')

        D+=str(B2D(int(A1[0])))+' '
        D+=str(B2D(int(A1[1])))+' '
        D+=str(B2D(int(A1[2])))+' '
        D+=str(B2D(int(A1[3])))+' '

        l3= Label(Fr7,textvariable= v17, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==4):
        v18.set(a1)
        A1=a1.split('\n')

        D+=str(B2D(int(A1[0])))+' '
        D+=str(B2D(int(A1[1])))+' '
        D+=str(B2D(int(A1[2])))+' '
        D+=str(B2D(int(A1[3])))+' '

        l3= Label(Fr7,textvariable= v18, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==5):
        v19.set(a1)
        A1=a1.split('\n')

        D+=str(B2D(int(A1[0])))+' '
        D+=str(B2D(int(A1[1])))+' '
        D+=str(B2D(int(A1[2])))+' '
        D+=str(B2D(int(A1[3])))+' '

        l3= Label(Fr7,textvariable= v19, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
```

```
    if((n+1)==6):
        v20.set(a1)
        A1=a1.split('\n')

        D+=str(B2D(int(A1[0])))+' '
        D+=str(B2D(int(A1[1])))+' '
        D+=str(B2D(int(A1[2])))+' '
        D+=str(B2D(int(A1[3])))+' '

        l3= Label(Fr7,textvariable= v20, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==7):
        v21.set(a1)
        A1=a1.split('\n')

        D+=str(B2D(int(A1[0])))+' '
        D+=str(B2D(int(A1[1])))+' '
        D+=str(B2D(int(A1[2])))+' '
        D+=str(B2D(int(A1[3])))+' '

        l3= Label(Fr7,textvariable= v21, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==8):
        v22.set(a1)
        A1=a1.split('\n')

        D+=str(B2D(int(A1[0])))+' '
        D+=str(B2D(int(A1[1])))+' '
        D+=str(B2D(int(A1[2])))+' '
        D+=str(B2D(int(A1[3])))+' '

        l3= Label(Fr7,textvariable= v22, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==9):
```

```
        v23.set(a1)
        A1=a1.split('\n')

        D+=str(B2D(int(A1[0])))+' '
        D+=str(B2D(int(A1[1])))+' '
        D+=str(B2D(int(A1[2])))+' '
        D+=str(B2D(int(A1[3])))+' '
        l3= Label(Fr7,textvariable= v23, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==10):
        v24.set(a1)
        A1=a1.split('\n')

        D+=str(B2D(int(A1[0])))+' '
        D+=str(B2D(int(A1[1])))+' '
        D+=str(B2D(int(A1[2])))+' '
        D+=str(B2D(int(A1[3])))+' '
        l3= Label(Fr7,textvariable= v24, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
def TRANSD(a,x1,y1,n):
    global D
    a1="
    for i in range(len(a)):
        a1+=a[i]
        if((i+1)%4==0):
            a1+='\n'
    a1=a1[:-1]

    a=a1.split('\n')

    a1=MT(a)
    a1=a1[:-1]

    if((n+1)==1):
```

```
        v15.set(a1)
        A1=a1.split('\n')

        D+=str(B2D(int(A1[0]+A1[1]+A1[2]+A1[3])))+' '

        l3= Label(Fr10,textvariable= v15, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==2):
        v16.set(a1)
        A1=a1.split('\n')

        D+=str(B2D(int(A1[0]+A1[1]+A1[2]+A1[3])))+' '

        l3= Label(Fr10,textvariable= v16, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==3):
        v17.set(a1)
        A1=a1.split('\n')

        D+=str(B2D(int(A1[0]+A1[1]+A1[2]+A1[3])))+' '

        l3= Label(Fr10,textvariable= v17, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==4):
        v18.set(a1)
        A1=a1.split('\n')

        D+=str(B2D(int(A1[0]+A1[1]+A1[2]+A1[3])))+' '

        l3= Label(Fr10,textvariable= v18, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==5):
        v19.set(a1)
        A1=a1.split('\n')
```

```
        D+=str(B2D(int(A1[0]+A1[1]+A1[2]+A1[3])))+' '

        l3= Label(Fr10,textvariable= v19, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==6):
        v20.set(a1)
        A1=a1.split('\n')

        D+=str(B2D(int(A1[0]+A1[1]+A1[2]+A1[3])))+' '

        l3= Label(Fr10,textvariable= v20, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==7):
        v21.set(a1)
        A1=a1.split('\n')

        D+=str(B2D(int(A1[0]+A1[1]+A1[2]+A1[3])))+' '

        l3= Label(Fr10,textvariable= v21, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==8):
        v22.set(a1)
        A1=a1.split('\n')

        D+=str(B2D(int(A1[0]+A1[1]+A1[2]+A1[3])))+' '

        l3= Label(Fr10,textvariable= v22, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==9):
        v23.set(a1)
        A1=a1.split('\n')

        D+=str(B2D(int(A1[0]+A1[1]+A1[2]+A1[3])))+' '
```

```
        l3= Label(Fr10,textvariable= v23, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)
    if((n+1)==10):
        v24.set(a1)
        A1=a1.split('\n')

        D+=str(B2D(int(A1[0]+A1[1]+A1[2]+A1[3])))+' '


        l3= Label(Fr10,textvariable= v24, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=6,height=4,).place(x = x1,y=y1,)

def fun5():
    global binary1,binary2,tik
    if(tik==4):
        tik=5
        binary=binary1.split(' ')
        binary=binary[:-1]
        binary2=binary

        x=180
        y=130
        a="
        for i in range(len(binary)):
            a=binary[i]
            x+=100
            MATRIX(a,x,y,i)
    else:
        messagebox.showinfo('MessageBox','Please hit the previous button ')
def fun6():
    global binary2,tik
    if(tik==5):
        tik=6
```

```
        x=180
        y=280
        a=''
        for i in range(len(binary2)):
            a=binary2[i]
            x+=100
            TRANS(a,x,y,i)
    else:
        messagebox.showinfo('MessageBox','Please hit the previous button ')
def fun7():
    global a,tik,D
    if(tik==6):
        tik=7
        v25.set(D)
    else:
        messagebox.showinfo('MessageBox','Please hit the previous button ')
#==============Encryption 3 page Calculations
============================#
def fun8():
    global tik
    if(tik==7):
        tik=8
        L=len(e2.get())
        v26.set(str(L))
    else:
        messagebox.showinfo('MessageBox','Please hit the previous button ')

def fun9():
    global a,vk,tik,D,vk1,ct
    mapT=['!','0','#','$','%','&','(',')','*','+','/',';','?','@','[',':',']','^',
          '{','|','}','~','×','–','Ø','©','®','±','µ',']']
    if(tik==8):
        tik=9
        k=len(e2.get())
```

```
        vk=''
        vk1=''
        ct=''
        D=D.split(' ')

        for i in range(len(D)):
            if(D[i].isnumeric()):
                x=int(k)+int(D[i])
                vk+=str(x)+' '
                vk1+=str(int(x)%32)+' '
                y=int(x)%32
                ct+=str(mapT[int(y)])
        v27.set(vk)
        D=''
    else:
        messagebox.showinfo('MessageBox','Please hit the previous button ')
def fun10():
    global vk1,tik
    if(tik==9):
        tik=10
        v28.set(vk1)
    else:
        messagebox.showinfo('MessageBox','Please hit the previous button ')
def fun11():
    global ct,tik
    if(tik==10):
        tik=11
        v29.set(ct)
    else:
        messagebox.showinfo('MessageBox','Please hit the previous button ')



def SaveCT():
    global ct,tik
```

```
    if(tik==11):
        tik=12
        f = asksaveasfile(mode='w', defaultextension=".txt")
        if f is None:
            return
        f.write(ct)
        messagebox.showinfo("message dialog box", 'CIPHER text saved successfully')
        f.close()
    else:
        messagebox.showinfo('MessageBox','Please previous button ')

def Savekeys():
    global keys,tik
    if(tik==12):
        tik=13
        f = asksaveasfile(mode='w', defaultextension=".txt")
        if f is None:
            return
        f.write(e2.get())
        messagebox.showinfo("message dialog box", 'plain text saved successfully')
        f.close()
    else:
        messagebox.showinfo('MessageBox','Please previous button ')
#Decryption Calculations page-1-----------------------------------------------
kit=0
def OpenCode():
    global ct
    global kit
    if(kit==0):
        kit=1
        file=askopenfile(mode='r',filetypes=[('All files','*.txt')])
        if file is not None:
            ct=file.read()
        v30.set(ct)
```

```
    else:
        messagebox.showinfo('MessageBox','Please previous button ')
def fun12():
    global ct,kit,ctasci

    mapT=['!','0','#','$','%','&','(',')','*','+','/',':',';','?','@','[',':',']','^',
          '{','|','}','~','×','–','Ø','©','®','±','µ',']']
    if(kit==1):
        kit=2
        ctasci=''
        for i in range(len(ct)):
            ctasci+=str(mapT.index(ct[i]))+' '
        v31.set(ctasci)
    else:
        messagebox.showinfo('MessageBox','Please hit the previous button ')
def OpenKey():
    global key,kit
    if(kit==2):
        kit=3
        file=askopenfile(mode='r',filetypes=[('All files','*.txt')])
        if file is not None:
            key=file.read()
        v32.set(key)
    else:
        messagebox.showinfo('MessageBox','Please previous button ')


def fun13():
    global key,kit,k

    if(kit==3):
        kit=4
        k=len(key)
        v33.set(str(k))
    else:
```

```
        messagebox.showinfo('MessageBox','Please hit the previous button ')

def fun14():
    global k,pt1,ctasci,kit,bn1
    if(kit==4):
        kit=5
        ctasci=ctasci.split(' ')
        ct=ctasci[:-1]
        pt1=''
        bn1=''
        for i in range(len(ct)):
            m=str(int(ct[i])- int(k))
            pt1+=str(int(int(m)+32)%32)+' '
            bn1+=str(D4B(int(int(m)+32)%32))+' '
            #print(str(D8B(int(int(m)+32)%32)))
        v34.set(pt1)
    else:
        messagebox.showinfo('MessageBox','Please hit the previous button ')

def D4B(a):
    bnr = bin(a).replace('0b','')
    x = bnr[::-1]
    while len(x) < 4:
        x += '0'
    bnr = x[::-1]
    return bnr

def fun15():
    global kit,bn1
    if(kit==5):
        kit=6
        v35.set(bn1)
    else:
        messagebox.showinfo('MessageBox','Please hit the previous button ')
```

```
#Decryption Calculations page-2-------------------------------------------------------------

def fun16():
    global kit,bn1
    global binary1,binary2

    if(kit==6):
        kit=7
        binary1=bn1
        binary1=binary1.replace(' ','')
        b=''

        for i in range(len(binary1)):
            b+=binary1[i]
            if((i+1)%16==0):
                b+=' '
        binary1=b

        binary=binary1.split(' ')
        binary=binary[:-1]
        binary2=binary

        x=180
        y=130
        a=''
        for i in range(len(binary)):
            a=binary[i]
            x+=100
            MATRIXD(a,x,y,i)
    else:
        messagebox.showinfo('MessageBox','Please hit the previous button ')

def fun17():
```

```
    global kit,binary2
    if(kit==7):
        kit=8


        x=180
        y=280
        a=''
        for i in range(len(binary2)):
            a=binary2[i]
            x+=100
            TRANSD(a,x,y,i)
    else:
        messagebox.showinfo('MessageBox','Please hit the previous button ')
def fun18():
    global kit,D
    if(kit==8):
        kit=9
        v36.set(D)
    else:
        messagebox.showinfo('MessageBox','Please hit the previous button ')

#Decryption Calculations page-3----------------------------------------------------------------

def fun19():
    global kit,key
    if(kit==9):
        kit=10
        file=askopenfile(mode='r',filetypes=[('All files','*.txt')])
        if file is not None:
            key=file.read()
        v37.set(key)
    else:
        messagebox.showinfo('MessageBox','Please previous button ')
```

```
def B2D(binary):
    binary1 = binary
    decimal, i, n = 0, 0, 0
    while(binary != 0):
        dec = binary % 10
        decimal = decimal + dec * pow(2, i)
        binary = binary//10
        i += 1
    return(decimal)

def fun20():
    global k1,kit,key,kasc
    if(kit==10):
        kit=11
        kasc=''
        for i in range(len(key)):
            kasc+=str(ord(key[i]))+' '
        v38.set(kasc)
    else:
        messagebox.showinfo('MessageBox','Please hit the previous button ')
def fun21():
    global kit,D
    if(kit==11):
        kit=12
        v39.set(D)
    else:
        messagebox.showinfo('MessageBox','Please hit the previous button ')

def fun22():
    global kasc,kit,D,pt
    if(kit==12):
        kit=13
        kasc=kasc.split(' ')
        kasc=kasc[:-1]
```

```
        D=D.split(' ')
        D=D[:-1]
        dec=''
        pt=''
        for i in range(len(D)):
            k=int(int(D[i])/int(kasc[i]))
            dec+=str(k)+' '
            pt+=chr(k)

        v40.set(dec)
    else:
        messagebox.showinfo('MessageBox','Please hit the previous button ')
def fun23():
    global kit,pt
    if(kit==13):
        kit=14
        v41.set(pt)
    else:
        messagebox.showinfo('MessageBox','Please hit the previous button ')
#=====================================================
def clearAll():
    global tik,kit
    clearE1()
    clearE2()
    clearE3()
    clearD1()
    clearD2()
    tik=0
    kit=0
def clearE1():
    global tik
    tik=0
    e1.delete(0,END)
    e2.delete(0,END)
```

```
    v1.set('')
    v2.set('')
    v3.set('')
    v4.set('')
def clearE2():
    global tik
    tik=0
    v5.set('')
    v6.set('')
    v7.set('')
    v8.set('')
    v9.set('')
    v10.set('')
    v11.set('')
    v12.set('')
    v13.set('')
    v14.set('')
    v15.set('')
    v16.set('')
    v17.set('')
    v18.set('')
    v19.set('')
    v20.set('')
    v21.set('')
    v22.set('')
    v23.set('')
    v24.set('')
    v25.set('')
def clearE3():
    global tik
    tik=0
    v26.set('')
    v27.set('')
    v28.set('')
```

```
def clearD1():
    global kit
    kit=0
    v29.set('')
    v30.set('')
    v31.set('')
    v32.set('')
    v33.set('')
    v34.set('')
    v35.set('')
    v36.set('')
def clearD2():
    global kit
    kit=0
    v32.set('')
    v33.set('')
    v34.set('')
    v35.set('')
    v36.set('')
    v37.set('')
    v38.set('')
    v39.set('')
    v40.set('')
def clearD3():
    global kit
    kit=0
    v37.set('')
    v38.set('')
    v39.set('')
    v40.set('')
    v41.set('')
    v42.set('')
def clearD4():
    global kit
```

```
    kit=0
    v43.set('')
    v44.set('')
    v45.set('')
    v46.set('')

r = Tk()
Fr0 = Frame(r)
Fr1 = Frame(r)
Fr2 = Frame(r)
Fr3 = Frame(r)
Fr4 = Frame(r)
Fr5 = Frame(r)
Fr6 = Frame(r)
Fr7 = Frame(r)
Fr8 = Frame(r)
Fr9 = Frame(r)
Fr10 = Frame(r)
Fr11 = Frame(r)
Fr12 = Frame(r)


Fr0.place(x = 0,y = 0,height=157, width=1300)
Fr1.place(x = 0,y = 157,height=680, width=1300)
Fr2.place(x = 0,y = 157,height=680, width=1300)
Fr3.place(x = 0,y = 157,height=680, width=1300)
Fr4.place(x = 0,y = 157,height=680, width=1300)
Fr5.place(x = 0,y = 157,height=680, width=1300)
Fr6.place(x = 0,y = 157,height=680, width=1300)
Fr7.place(x = 0,y = 157,height=680, width=1300)
Fr8.place(x = 0,y = 157,height=680, width=1300)
Fr9.place(x = 0,y = 157,height=680, width=1300)
Fr10.place(x = 0,y = 157,height=680, width=1300)
Fr11.place(x = 0,y = 157,height=680, width=1300)
```

```
Fr12.place(x = 0,y = 157,height=680, width=1300)


Fr1.config(bg='white')
Fr2.config(bg='white')
Fr3.config(bg='white')
Fr4.config(bg='white')
Fr5.config(bg='white')
Fr6.config(bg='white')
Fr7.config(bg='white')
Fr8.config(bg='white')
Fr9.config(bg='white')
Fr10.config(bg='white')
Fr11.config(bg='white')
Fr12.config(bg='white')

#Variable Declaration----------------------------------------------------------------------
v1=StringVar()
v2=StringVar()
v3=StringVar()
v4=StringVar()
v5=StringVar()
v6=StringVar()
v7=StringVar()
v8=StringVar()
v9=StringVar()
v10=StringVar()
v11=StringVar()
v12=StringVar()
v121=StringVar()
v13=StringVar()
v14=StringVar()
v15=StringVar()
v16=StringVar()
```

```
v17=StringVar()
v18=StringVar()
v19=StringVar()
v20=StringVar()
v21=StringVar()
v22=StringVar()
v23=StringVar()
v24=StringVar()
v25=StringVar()
v26=StringVar()
v27=StringVar()
v28=StringVar()
v29=StringVar()
v30=StringVar()
v31=StringVar()
v32=StringVar()
v33=StringVar()
v34=StringVar()
v35=StringVar()
v36=StringVar()
v37=StringVar()
v38=StringVar()
v39=StringVar()
v40=StringVar()
v41=StringVar()
v42=StringVar()
v43=StringVar()
v44=StringVar()
v45=StringVar()

#HeadingPage
ph1=ImageTk.PhotoImage(Image.open("logo.png"))
lab2 = Label(Fr0,image=ph1).place(x = -40, y = 0)
```

```
#----------------Home Page Fr-1 ------------------------------------------------------------------

ph2=ImageTk.PhotoImage(Image.open("bg5.jpg"))
lab2 = Label(Fr1,image=ph2).place(x = 0, y = 0)

lab2 = Label(Fr1, justify="left",text = '''A Cryptographic Algorithm Based on ASCII and Number System
    Conversions along with A Cyclic Mathematical Function
''',
            fg="black",bg='white', font = "TimesNewRoman 22 bold",height=3).place(x = 140,y=100)

lab2 = Label(Fr1, justify="left",text = '''
Project By  MCA  4th semester
Name : Bendi Satya Pavan
Rno :  PG212202006
''',
            fg="black",bg='white', font = "TimesNewRoman 11 bold",height=6).place(x = 180,y=350)

lab2 = Label(Fr1, justify="left",text = '''
Project Guide By
Sir. B.Divakar
Assistant Professor
''',
            fg="black",bg='white', font = "TimesNewRoman 14 bold",height=5).place(x = 720,y=350)

b1 = Button(Fr1, text = "Proceed",fg="black",bg="white",bd=5, font = "TimesNewRoman 14 bold",width=8,command = fr5).place(x = 500, y = 270)
b2 = Button(Fr1, text = "Close",fg="black",bg="white",bd=5, font = "TimesNewRoman 14 bold",width=8,command=finish).place(x = 1050, y = 500)
```

```
#--------------------Abstract Page   ----Fr-2----------------------------------------------------

ph3=ImageTk.PhotoImage(Image.open("bg5.jpg"))
lab4 = Label(Fr2,image=ph3).place(x = 0, y = 0)

lab2 = Label(Fr2, justify="left",text = "'A Cryptographic Algorithm Based on ASCII and
Number System
    Conversions along with A Cyclic Mathematical Function
'",
            fg="black",bg='white', font = "TimesNewRoman 18 bold",height=3).place(x =
220,y=10)

l1 = Label(Fr2, text = "Abstract",fg="black",font = "TimesNewRoman 16
bold",bg='white',height=1,width=10).place(x = 500,y=120)


l3= Label(Fr2, text="'

Data encryption and decryption in an efficient manner are the challenging aspects of
modern
information theory. In this algorithm, the plaintext to be encrypted is converted into
unprintable characters. For encryption, a different technique is applied based on ASCII
and
number system conversions, which makes this algorithm different from others. First,
each
character of the plaintext is converted into its equivalent ASCII (decimal). Then, using
some
matrix manipulations on the decimal, representation of each character is transformed to 5
unprintable characters. After that, every unprintable character in the intermediate cipher
text
is further converted into a different unprintable character using a cyclic mathematical
```

function. Performing three steps of processing, the final encrypted message is produced that
gives higher level of security.
'",

fg="black",bg="white",justify=LEFT, font = "TimesNewRoman 12 bold",height=12).place(x = 250,y=180)

b8 = Button(Fr2, text = "Prev",fg="black",bg="white",bd=5, font = "TimesNewRoman 10 bold",width=6,command = fr1).place(x = 50, y=500)
b7 = Button(Fr2, text = "Next",fg="black",bg="white",bd=5, font = "TimesNewRoman 10 bold",width=6,command=fr3).place(x = 1050, y=500)

#--------------------Current System Page ----Fr-3----------

ph4=ImageTk.PhotoImage(Image.open("bg5.jpg"))
lab4 = Label(Fr3,image=ph4).place(x = 0, y = 0)

lab2 = Label(Fr3, justify="left",text = "'A Cryptographic Algorithm Based on ASCII and Number System
Conversions along with A Cyclic Mathematical Function
'",
fg="black",bg='white', font = "TimesNewRoman 18 bold",height=3).place(x = 220,y=10)
l1 = Label(Fr3, text = "Current System",fg="black",font = "TimesNewRoman 16 bold",bg='white',height=1,width=14).place(x = 450,y=120)

l3= Label(Fr3, text="'
In Existing system there are many cryptographic algorithms are introduced.
One of the earliest and most used mechanisms in cryptography is Caesar's cipher which is also called a shift cipher.

It is a replacement mechanism in which each letter of the plain text is replaced by another letter which is certain places ahead of the letter and the process is repeated for all the letters in the plain text.
The number of places ahead to be used is the key to the encryption. For example, if the key is 2, then a will be replaced by c which is two places ahead of a.

'",
fg="black",bg="white",justify=LEFT, font = "TimesNewRoman 12 bold",height=10).place(x = 250,y=180)

b8 = Button(Fr3, text = "Prev",fg="black",bg="white",bd=5, font = "TimesNewRoman 10 bold",width=6,command = fr2).place(x = 50, y=500)
b7 = Button(Fr3, text = "Next",fg="black",bg="white",bd=5, font = "TimesNewRoman 10 bold",width=6,command=fr4).place(x = 1050, y=500)

#--------------------Proposed System Page ----Fr-4----------

ph5=ImageTk.PhotoImage(Image.open("bg5.jpg"))
lab4 = Label(Fr4,image=ph5).place(x = 0, y = 0)

lab2 = Label(Fr4, justify="left",text = '"A Cryptographic Algorithm Based on ASCII and Number System
Conversions along with A Cyclic Mathematical Function
'",
fg="black",bg='white', font = "TimesNewRoman 18 bold",height=3).place(x = 220,y=10)
l1 = Label(Fr4, text = "Proposed System",fg="black",font = "TimesNewRoman 16 bold",bg='white',height=1,width=18).place(x = 450,y=120)

l3= Label(Fr4, text="'
Day by day the level of security is going to be higher. Still now many researchers are working on cryptography and data hiding.A new cryptographic algorithm for the Real

Time
Application was in to improve the time for encryption and decryption of data of end-to-end
delay and to provide higher level of security. In this paper, we proposed an improved algorithmwhich is different from the traditional symmetric-key cryptography, asymmetric-key
cryptographyor hashing function.A cryptographic algorithm based on ASCII conversion and a cyclic
mathematicalfunction was presented in, and which makes the cipher differentfrom other algorithms.

'",

fg="black",bg="white",justify=LEFT, font = "TimesNewRoman 12 bold",height=15,).place(x = 180,y=180)

b8 = Button(Fr4, text = "Prev",fg="black",bg="white",bd=5, font = "TimesNewRoman 10 bold",width=6,command = fr3).place(x = 50, y=500)
b7 = Button(Fr4, text = "Next",fg="black",bg="white",bd=5, font = "TimesNewRoman 10 bold",width=6,command=fr5).place(x = 1050, y=500)

#----------Menu Page Fr-5 ----------

ph6=ImageTk.PhotoImage(Image.open("bg5.jpg"))
lab2 = Label(Fr5,image=ph6).place(x = 0, y = 0)

ll = Label(Fr5, text = "Menu",justify=LEFT,fg="black",bg="white",bd=5, font = "TimesNewRoman 14 bold",width=18,).place(x = 430, y =60)
b1 = Button(Fr5, text = "Home",justify=LEFT,fg="black",bg="white",bd=5, font = "TimesNewRoman 12 bold",height=1,width=18,command=fr1).place(x = 450, y =130)
b1 = Button(Fr5, text = "Abstract",justify=LEFT,fg="black",bg="white",bd=5, font = "TimesNewRoman 12 bold",height=1,width=18,command=fr2).place(x = 450, y =180)
b1 = Button(Fr5, text = "Current System",justify=LEFT,fg="black",bg="white",bd=5,

```
font = "TimesNewRoman 12 bold",height=1,width=18,command=fr3).place(x = 450, y =230)
b1 = Button(Fr5, text = "Proposed System",justify=LEFT,fg="black",bg="white",bd=5, font = "TimesNewRoman 12 bold",height=1,width=18,command=fr4).place(x = 450, y =280)
b1 = Button(Fr5, text = "Encryption",justify=LEFT,fg="black",bg="white",bd=5, font = "TimesNewRoman 12 bold",height=1,width=18,command=fr6).place(x = 450, y =330)
b1 = Button(Fr5, text = "Decryption",justify=LEFT,fg="black",bg="white",bd=5, font = "TimesNewRoman 12 bold",height=1,width=18,command=fr9).place(x = 450, y =380)

b1 = Button(Fr5, text = "Close",justify=LEFT,fg="black",bg="white",bd=5, font = "TimesNewRoman 12 bold",height=1,width=18,command=finish).place(x = 450, y =430)



#--------------------Encryption page-1   ----Fr-6----------------------------------------------

ph7=ImageTk.PhotoImage(Image.open("bg5.jpg"))
lab4 = Label(Fr6,image=ph7).place(x = 0, y = 0)
l22 = Label(Fr6, text = "ENCRYPTION",fg="black",font = "TimesNewRoman 20 bold",bg='white',height=1,width=11).place(x = 500,y=70)
l3= Label(Fr6, text = "Plain Text",fg="black",bg="white",bd=5, font = "TimesNewRoman 16 bold",height=1,width=14).place(x = 150, y =130)
e1=Entry(Fr6,bg="White",fg="black", font = "TimesNewRoman 16 bold",width=16)
e1.place(x = 400,y=130)

b1 = Button(Fr6, text = "Convert to ASCII",justify=LEFT,fg="black",bg="white",bd=5, font = "TimesNewRoman 12 bold",height=1,width=18,command=fun1).place(x = 150, y =180)
l3= Label(Fr6,textvariable= v1, fg="black",bg="white",anchor="w", font = "TimesNewRoman 16 bold",width=42).place(x = 400,y=180)

l3= Label(Fr6, text = "Enter Key",fg="black",bg="white",bd=5, font =
```

```
"TimesNewRoman 16 bold",height=1,width=12).place(x = 150, y =230)
e2=Entry(Fr6,bg="White",fg="black", font = "TimesNewRoman 16 bold",width=42)
e2.place(x = 400,y=230)

b1 = Button(Fr6, text = "Convert to ASCII",justify=LEFT,fg="black",bg="white",bd=5,
font = "TimesNewRoman 12 bold",height=1,width=18,command=fun2).place(x = 150, y
=280)
l3= Label(Fr6,textvariable= v2, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=42).place(x = 400,y=280)

b1 = Button(Fr6, text = "Product",fg="black",bg="white",bd=5, font =
"TimesNewRoman 12 bold",height=1,width=18,command=fun3).place(x = 150, y =330)
l3= Label(Fr6,textvariable= v3, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=42).place(x = 400,y=330)

b1 = Button(Fr6, text = "Product Binary",fg="black",bg="white",bd=5, font =
"TimesNewRoman 12 bold",height=1,width=18,command=fun4).place(x = 150, y =380)
l3= Label(Fr6,textvariable= v4, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=42).place(x = 400,y=380)
b8 = Button(Fr6, text = "Prev",fg="black",bg="white",bd=5, font = "TimesNewRoman
10 bold",width=6,command = fr5).place(x = 50, y=500)
b8 = Button(Fr6, text = "Home",fg="black",bg="white",bd=5, font = "TimesNewRoman
10 bold",width=6,command = fr1).place(x = 400, y=500)
b8 = Button(Fr6, text = "Clear",fg="black",bg="white",bd=5, font = "TimesNewRoman
10 bold",width=6,command = clearE1).place(x = 500, y=500)

b7 = Button(Fr6, text = "Next",fg="black",bg="white",bd=5, font = "TimesNewRoman
10 bold",width=6,command=fr7).place(x = 1050, y=500)


#--------------------Encryption page-2   ----Fr-7-----------------------------------------

ph8=ImageTk.PhotoImage(Image.open("bg5.jpg"))
lab4 = Label(Fr7,image=ph8).place(x = 0, y = 0)
```

```
b1 = Button(Fr7, text = "Matrix",fg="black",bg="white",bd=5, font = "TimesNewRoman
12 bold",height=1,width=18,command=fun5).place(x = 70, y =130)
b1 = Button(Fr7, text = "Transpose",fg="black",bg="white",bd=5, font =
"TimesNewRoman 12 bold",height=1,width=18,command=fun6).place(x = 70, y =280)

b1 = Button(Fr7, text = "Convert to Values(CT1)",fg="black",bg="white",bd=5, font =
"TimesNewRoman 12 bold",height=1,width=18,command=fun7).place(x = 70, y =450)
l3= Label(Fr7,textvariable= v25, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=42).place(x = 350,y=450)


b8 = Button(Fr7, text = "Prev",fg="black",bg="white",bd=5, font = "TimesNewRoman
10 bold",width=6,command = fr6).place(x = 50, y=500)
b8 = Button(Fr7, text = "Home",fg="black",bg="white",bd=5, font = "TimesNewRoman
10 bold",width=6,command = fr1).place(x = 400, y=500)
b8 = Button(Fr7, text = "Clear",fg="black",bg="white",bd=5, font = "TimesNewRoman
10 bold",width=6,command = clearE2).place(x = 500, y=500)

b7 = Button(Fr7, text = "Next",fg="black",bg="white",bd=5, font = "TimesNewRoman
10 bold",width=6,command=fr8).place(x = 1050, y=500)

#--------------------Encryption page-3   ----Fr-8-----------------------------------------------

ph9=ImageTk.PhotoImage(Image.open("bg5.jpg"))
lab4 = Label(Fr8,image=ph9).place(x = 0, y = 0)

ph09=ImageTk.PhotoImage(Image.open("map1.jpg"))
lab4 = Label(Fr8,image=ph09).place(x = 700, y = 50)

l3= Button(Fr8, text = "Key Length KL",fg="black",bg="white",bd=5, font =
"TimesNewRoman 12 bold",height=1,width=14,command=fun8).place(x = 150, y =130)
l=Label(Fr8,textvariable= v26,bg="White",fg="black", font = "TimesNewRoman 16
bold",width=16).place(x = 400,y=130)
```

```
b1 = Button(Fr8, text = "CT1 + KL ",fg="black",bg="white",bd=5, font =
"TimesNewRoman 12 bold",height=1,width=18,command=fun9).place(x = 150, y =200)
l3= Label(Fr8,textvariable= v27, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=42).place(x = 400,y=200)

b1 = Button(Fr8, text = "(CT1+KL) % 32",fg="black",bg="white",bd=5, font =
"TimesNewRoman 12 bold",height=1,width=18,command=fun10).place(x = 150, y
=250)
l3= Label(Fr8,textvariable= v28, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=42).place(x = 400,y=250)

b1 = Button(Fr8, text = "Cipher Text",fg="black",bg="white",bd=5, font =
"TimesNewRoman 12 bold",height=1,width=18,command=fun11).place(x = 150, y
=300)
l3= Label(Fr8,textvariable= v29, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=42).place(x = 400,y=300)

b8 = Button(Fr8, text = "Save Key",fg="black",bg="white",bd=5, font =
"TimesNewRoman 10 bold",command = Savekeys).place(x = 350, y=500)
b8 = Button(Fr8, text = "Save CipherText",fg="black",bg="white",bd=5, font =
"TimesNewRoman 10 bold",command = SaveCT).place(x = 450, y=500)

b8 = Button(Fr8, text = "Prev",fg="black",bg="white",bd=5, font = "TimesNewRoman
10 bold",width=6,command = fr7).place(x = 50, y=500)
b8 = Button(Fr8, text = "Home",fg="black",bg="white",bd=5, font = "TimesNewRoman
10 bold",width=6,command = fr1).place(x = 600, y=500)
b8 = Button(Fr8, text = "Clear",fg="black",bg="white",bd=5, font = "TimesNewRoman
10 bold",width=6,command = clearE3).place(x = 700, y=500)
b7 = Button(Fr8, text = "Next",fg="black",bg="white",bd=5, font = "TimesNewRoman
10 bold",width=6,command=fr9).place(x = 1050, y=500)
```

```
#--------------------Decryption page-1   ----Fr-9-----------------------------------------

ph10=ImageTk.PhotoImage(Image.open("bg5.jpg"))
lab4 = Label(Fr9,image=ph10).place(x = 0, y = 0)
l22 = Label(Fr9, text = "DECRYPTION",fg="black",font = "TimesNewRoman 20
bold",bg='white',height=1,width=11).place(x = 500,y=70)
b1 = Button(Fr9, text = "Browse Cipher Text",fg="black",bg="white",bd=5, font =
"TimesNewRoman 12 bold",height=1,width=18,command=OpenCode).place(x = 150, y
=130)
l3= Label(Fr9,textvariable= v30, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=42).place(x = 400,y=130)

b1 = Button(Fr9, text = "Conver to ASCII",fg="black",bg="white",bd=5, font =
"TimesNewRoman 12 bold",height=1,width=18,command=fun12).place(x = 150, y
=180)
l3= Label(Fr9,textvariable= v31, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=42).place(x = 400,y=180)

b1 = Button(Fr9, text = "Browse Key",fg="black",bg="white",bd=5, font =
"TimesNewRoman 12 bold",height=1,width=18,command=OpenKey).place(x = 150, y
=230)
l3= Label(Fr9,textvariable= v32, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=42).place(x = 400,y=230)

b1 = Button(Fr9, text = "Key Values",fg="black",bg="white",bd=5, font =
"TimesNewRoman 12 bold",height=1,width=18,command=fun13).place(x = 150, y
=270)
l3= Label(Fr9,textvariable= v33, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=42).place(x = 400,y=270)

b1 = Button(Fr9, text = "(A-k+32)%32",fg="black",bg="white",bd=5, font =
"TimesNewRoman 12 bold",height=1,width=18,command=fun14).place(x = 150, y
=320)
l3= Label(Fr9,textvariable= v34, fg="black",bg="white",anchor="w", font =
```

"TimesNewRoman 16 bold",width=42).place(x = 400,y=320)

b1 = Button(Fr9, text = "Binary",fg="black",bg="white",bd=5, font = "TimesNewRoman 12 bold",height=1,width=18,command=fun15).place(x = 150, y =370)
l3= Label(Fr9,textvariable= v35, fg="black",bg="white",anchor="w", font = "TimesNewRoman 16 bold",width=42).place(x = 400,y=370)

b8 = Button(Fr9, text = "Prev",fg="black",bg="white",bd=5, font = "TimesNewRoman 10 bold",width=6,command = fr8).place(x = 50, y=500)
b8 = Button(Fr9, text = "next ",fg="black",bg="white",bd=5, font = "TimesNewRoman 10 bold",width=6,command=fr10).place(x = 1050, y=500)

b7 = Button(Fr9, text = "Clear",fg="black",bg="white",bd=5, font = "TimesNewRoman 10 bold",width=6,command = clearD1).place(x = 560, y=500)

#--------------------Decryption Page-2 ----Fr-10----------

ph11=ImageTk.PhotoImage(Image.open("bg5.jpg"))
lab4 = Label(Fr10,image=ph11).place(x = 0, y = 0)

b1 = Button(Fr10, text = "4x4 matrix",fg="black",bg="white",bd=5, font = "TimesNewRoman 12 bold",height=1,width=12,command=fun16).place(x = 100, y =130)
b1 = Button(Fr10, text = "Transpose of",fg="black",bg="white",bd=5, font = "TimesNewRoman 12 bold",height=1,width=12,command=fun17).place(x = 100, y =250)

b1 = Button(Fr10, text = "Convert to Values",fg="black",bg="white",bd=5, font = "TimesNewRoman 12 bold",height=1,width=18,command=fun18).place(x = 100, y =400)
l3= Label(Fr10,textvariable= v36, fg="black",bg="white",anchor="w", font =

```
"TimesNewRoman 16 bold",width=42).place(x = 400,y=400)

b8 = Button(Fr10, text = "Prev",fg="black",bg="white",bd=5, font = "TimesNewRoman
10 bold",width=6,command = fr9).place(x = 50, y=500)

b7 = Button(Fr10, text = "Clear",fg="black",bg="white",bd=5, font = "TimesNewRoman
10 bold",width=6,command = clearD2).place(x = 560, y=500)
b7 = Button(Fr10, text = "Next",fg="black",bg="white",bd=5, font = "TimesNewRoman
10 bold",width=6,command=fr11).place(x = 1050, y=500)



#--------------------Decryption Page-3   ----Fr-11----------------------------------------

ph12=ImageTk.PhotoImage(Image.open("bg5.jpg"))
lab4 = Label(Fr11,image=ph12).place(x = 0, y = 0)

b1 = Button(Fr11, text = "Browse key 1",fg="black",bg="white",bd=5, font =
"TimesNewRoman 12 bold",height=1,width=18,command=fun19).place(x = 150, y
=130)
l3= Label(Fr11,textvariable= v37, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=42).place(x = 400,y=130)

b1 = Button(Fr11, text = "Convert to ASCII",fg="black",bg="white",bd=5, font =
"TimesNewRoman 12 bold",height=1,width=18,command=fun20).place(x = 150, y
=180)
l3= Label(Fr11,textvariable= v38, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=42).place(x = 400,y=180)

b1 = Button(Fr11, text = "Get Products",fg="black",bg="white",bd=5, font =
"TimesNewRoman 12 bold",height=1,width=18,command=fun21).place(x = 150, y
=230)
l3= Label(Fr11,textvariable= v39, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=42).place(x = 400,y=230)
```

```
b1 = Button(Fr11, text = "Divide with Key",fg="black",bg="white",bd=5, font =
"TimesNewRoman 12 bold",height=1,width=18,command=fun22).place(x = 150, y
=280)
l3= Label(Fr11,textvariable= v40, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=42).place(x = 400,y=280)

b1 = Button(Fr11, text = "Plain Text",fg="black",bg="white",bd=5, font =
"TimesNewRoman 12 bold",height=1,width=18,command=fun23).place(x = 150, y
=330)
l3= Label(Fr11,textvariable= v41, fg="black",bg="white",anchor="w", font =
"TimesNewRoman 16 bold",width=42).place(x = 400,y=330)


b8 = Button(Fr11, text = "Prev",fg="black",bg="white",bd=5, font = "TimesNewRoman
10 bold",width=6,command = fr10).place(x = 50, y=500)

b7 = Button(Fr11, text = "Clear",fg="black",bg="white",bd=5, font = "TimesNewRoman
10 bold",width=6,command=clearD3).place(x = 560, y=500)
b7 = Button(Fr11, text = "Next",fg="black",bg="white",bd=5, font = "TimesNewRoman
10 bold",width=6,command=fr12).place(x = 1050, y=500)


#======CONCLUSION--Fr-12============================
ph13=ImageTk.PhotoImage(Image.open("bg5.jpg"))
lab4 = Label(Fr12,image=ph13).place(x = 0, y = 0)

lab2 = Label(Fr12, justify="left",text = '''A Cryptographic Algorithm Based on ASCII
and Number System
    Conversions along with A Cyclic Mathematical Function
''',
            fg="black",bg='white', font = "TimesNewRoman 18 bold",height=3).place(x =
220,y=10)
l1 = Label(Fr12, text = "Conclusion",fg="black",font = "TimesNewRoman 16
```

```
bold",bg='white',height=1,width=10).place(x = 530,y=120)


l3= Label(Fr12, text="'
To ensure higher security and to hide data in effective way the proposed algorithm contributes greatly. Here, we

present an algorithm which is based on ASCII conversion and number system conversion and a cyclic mathematical

function. This algorithm not only encrypts the data but also hides the data which gives more security. In future

we will try to increase the security technique and implement some real time security system and try to add

Steganography with the system.
'",

        fg="black",bg="white",justify=LEFT, font = "TimesNewRoman 12 bold",height=14).place(x = 150,y=180)

b8 = Button(Fr12, text = "Prev",fg="black",bg="white",bd=5, font = "TimesNewRoman 10 bold",width=6,command = fr11).place(x = 50, y=500)
b7 = Button(Fr12, text = "Next",fg="black",bg="white",bd=5, font = "TimesNewRoman 10 bold",width=6,command=fr1).place(x = 960, y=500)
#============================================================
raise_frame(Fr1)
r.resizable(0,0)
r.geometry("1200x730+100+0")
r.title("A Cryptographic Algorithm Based on ASCII and Number System Conversions along with A Cyclic Mathematical Function")
r.mainloop()
```

6. TESTING

Testing is the process of finding differences between the expected behavior specified by system models and the observed behavior of the system. Testing is a critical role in quality assurance and ensuring the reliability of development and these errors will be reflected in the codes the application should be thoroughly tested and validated.

Unit testing finds the differences between the object design model and its corresponding components. Structural testing finds differences between the system design model and a subset of integrated subsystems. Functional testing finds differences between the use case model and the system.

Finally, performance testing, finds differences between non-functional requirements and actual system performance. Form modeling point of view, testing is the attempt of falsification of the system with respect to the system models. The goal of testing is to design tests that exercise defects in the system and to reveal problems.

6.1. Testing Activities

Testing a large system is a complex system is a complex activity and like any complex activity. It must be breaking into smaller activities. Thus, incremental testing was performed on the project i.e., components and subsystems of the system were tested separately before integrating them to from the subsystem for system testing.

6.2 Unit Testing

Unit testing focuses on the building blocks of the software system that is the objects and subsystem. There are three motivations behind focusing on components. First unit testing reduces the complexity of overall test activities allowing focus on smaller units of the system, second unit testing makes it easier to pinpoint and correct faults given that few components are involved in the rest. Third unit testing allows parallelism in the testing activities, that is each component are involved in the test. Third unit testing allows parallelism in the testing activities that is each component can be tested independently of one another.

In this system each module of segment display, encryption and decryption are treated as individual units are tested individually. The following are some unit testing techniques.

Equivalence testing:

It is a black box testing technique that minimizes the number of test cases. The possible inputs are partitioned into equivalence classes and a test case is selected for each class.

Boundary testing:

It is a special case of equivalence testing and focuses on the conditions at the boundary of the equivalence classes. Boundary testing requires that the elements be selected from the edge of the equivalence classes.

Path testing:

It is a white box testing technique that identifies faults in the implementation of the component the assumption here is that exercising all possible paths through the code at least once. Most faults will trigger failure. This acquires knowledge of source code.

In this project, there are two screens one represents sender form and another for receiver. Sender can browse/type the plain text that should be converted into ASCII values and perform the modulus operation on it by taking the random number we can generate the key, this key for using both encryption and decryption. In the same manner, receiver browse the cipher and key discard the key from the cipher text then finally get the plain text.

6.3. Integration Testing

Integrated testing defects faults that have not been detected. During unit testing by focusing on small group on components two or more components are integrated and tested and once tests do not reveal any new faults, additional components are added to the group. This procedure allows testing of increasing more complex parts on the system while keeping the location of potential faults relatively small. I have used the following approach to implements and integrated testing.

Top-down testing strategy unit tests the components of the top layer and then integrated the components of the next layer down. When all components of the next layer have been tested together, the next layer is selected. This was repeated until all layers are combined and involved in the test. In this project, we first perform individual testing on the modules; Encryption and decryption which are individually verified are integrated for making a perfect project.

6.4. Validation Testing

The system completely assembled as package, the interfacing have been uncovered and corrected, and a final series of software tests are validation testing. The validation testing is nothing but validation success when system functions in a manner that can be reasonably expected by the customer. The system validation had done by series of Block-box test methods.

In this project, validation is performed on each individual control. In encryption and decryption, if the receiver browses the cipher text and the appropriate key and the cipher text are not match then the system will generate the wrong plaintext. If the key and the cipher text are matched, then system will generate the plain text.

6.5. System Testing

System testing of software is a testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic. The following are some system testing activities.

Usability testing:

It is technique used to evaluate a product by testing it on users. It finds differences between the functional requirements and the system. With minimum knowledge of network security is enough for to use our project.

Performance testing:

It covers a broad range of engineering or functional evaluations to meet measurable performance characteristics. The coding of the system involves a simple but

effective method that can perform well even the user submits text with high security.

Installation testing

It is a kind of quality assurance work in the software industry that focuses on what customers will need to do to install and set up the new software successfully. The system is installed in the target environment.

Finally, we test whether the application is able to encrypt and successfully decrypt the data. This involves testing all the individual system of the application. This is achieved through System testing.

6.6. Testing Types

Unit testing:

It finds the differences between the object design model and its corresponding components. In this test each component is tested independent of the other thus allowing parallelism in testing activity.

Ex: individual units like selecting the plain text and the key are not match each other then the system will not generate the appropriate result/plaintext.

Structural testing:

It finds difference between the system design model and a subset of integrated subsystems.

Functional testing:

It finds differences between the use case model and the system.

Performance testing:

It finds difference between non-functional requirements and actual system performance.

Testing Plain

Testing accounts for 45-75% of the typical project effort. It is also one of the most commonly underestimated activities on a project. A test plan is a document that answers the basic questions about your testing effort. It needs to be initiated during the requirements gathering phase of your project and should evolve into a roadmap for the testing phase.

Test Planning enables a more reliable estimate of the testing effort up front. It allows the project team to consider ways to reduce the testing effort without being under time pressure. Test Plan helps to identify problem areas and focuses the testing team's attention on the critical paths. Test plan reduces the probability of implementing non-tested components.

6.7. Test Case Report

Test Case-1:

Encryption

Process Test

Case id: 01

Test Case Name: Encryption Test

Test Case Type: Black Box Testing

Description	Expected Value	Observed value	Result
Enter the plain text for encryption process.	The plain text should be displayed on the screen.	Plain text is displayed.	Plain text
User clicks on encryption	Encryption process is done will get the Cipher text.	Will get the cipher text.	Cipher text.

Table 6.7.1: Encryption Process

Test Case-2:

Decryption

Process Test

Case id: 02

Test Case Name: Decryption Test

Test Case Type: Black Box Testing

Decryption	Expected Value	Observed Value	Result
User select the Cipher text.	Retrieve cipher text from the destination.	Will get the Cipher text.	Cipher text.
User clicks on decryption.	Decryption	Will get the Plain text.	Plain text.

Table 6.7.2: Decryption Process

RESULTS

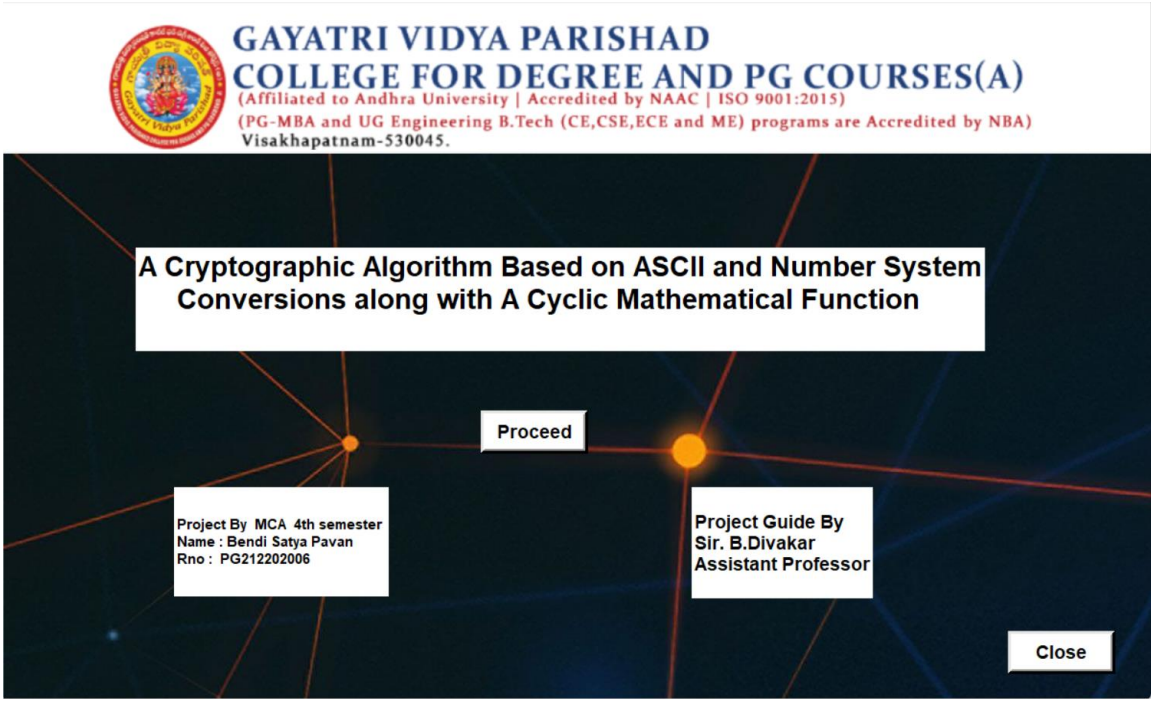


Figure 7.1 Home page

Project Execution starts from Home Page having the title of the actual project. If the Sender clicks on "**Proceed**" then the Project execution will begin.

Figure 7.2. Menu Page

Click on **Proceed** button it navigates to Menu page.

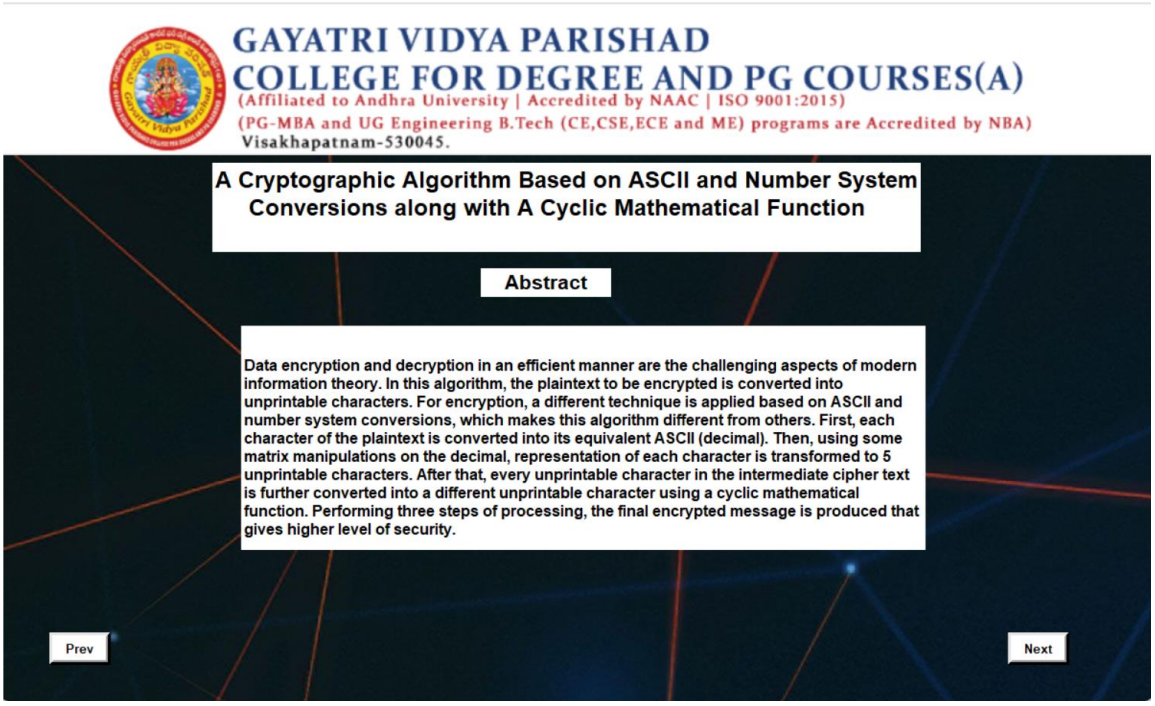


Figure 7.3. Abstract Page

click on **Abstract** Button it will display Abstract of the project in detail.

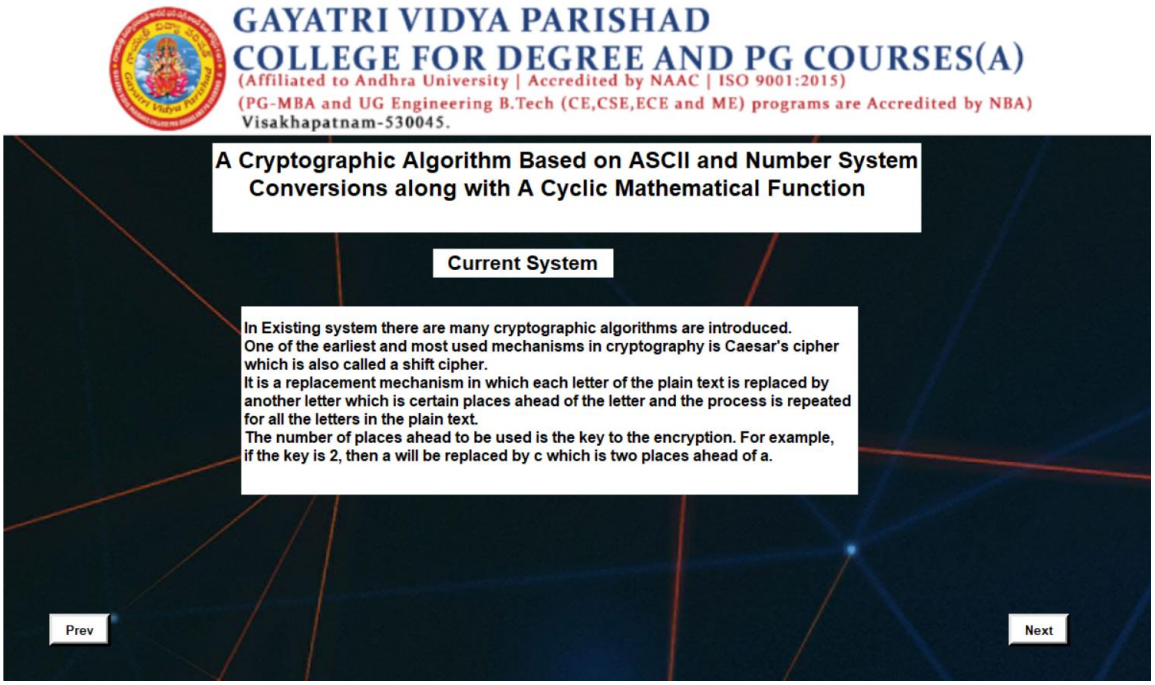


Figure 7.4. Current System Page

Click on **Current System** Button in menu page it navigates to Current system page.

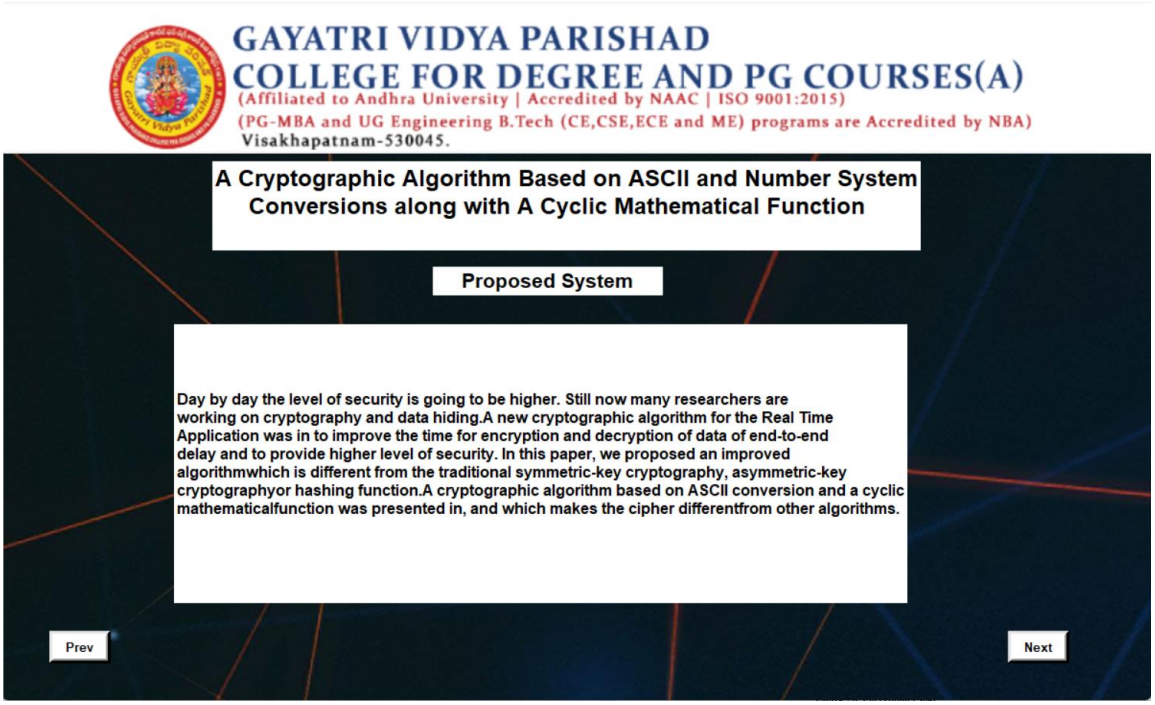


Figure 7.5. Proposed System Page

Click on **Proposed System** Button in menu page it navigates to Proposed System page.

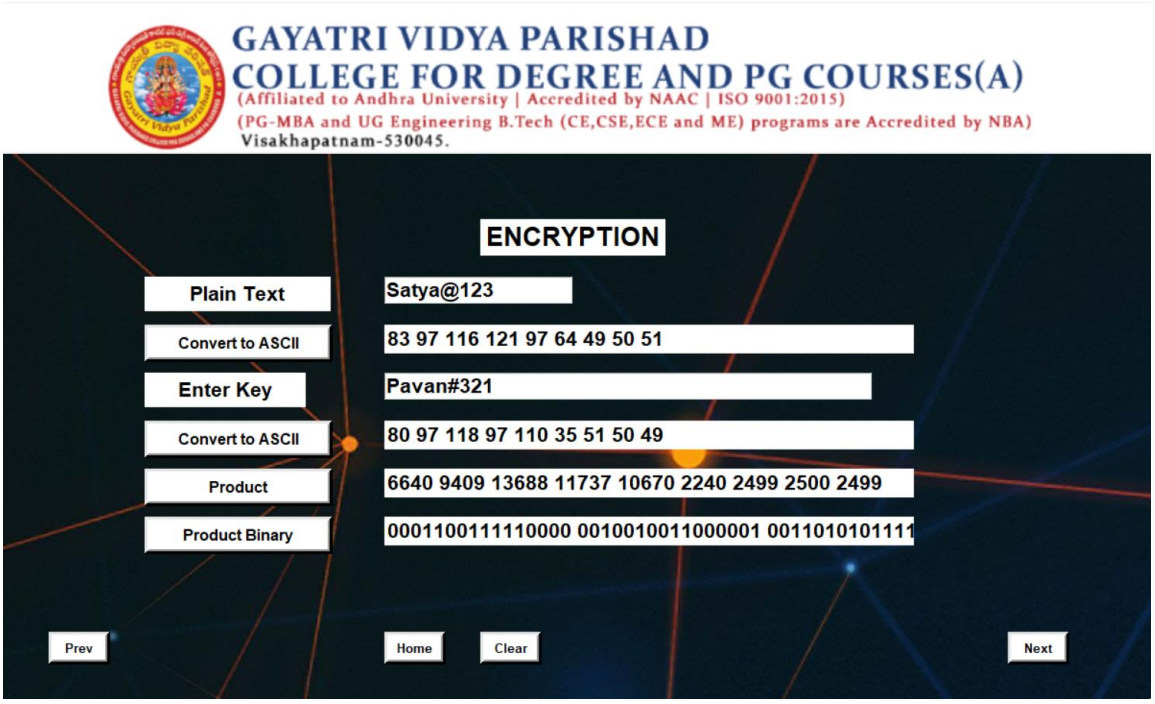


Figure 7.6. Encryption Page1

If the sender click on encryption button in menu page it navigates to the E**NCRYPTION** page. Here is the process for encryption. Sender need to enter the Plain Text and Enter key and click on convert to ASCII button, Click on Product button to product ASCII values. To covert the product value into binary value click on Product Binary

Button.Click on Next button for further Proccess.

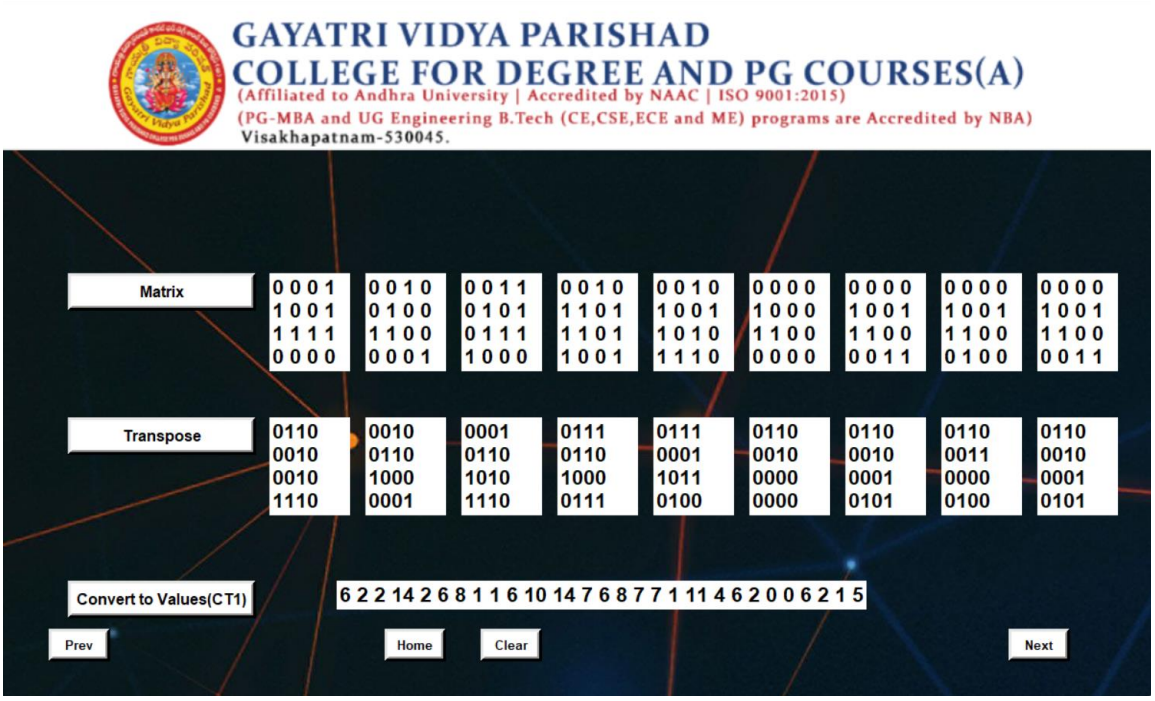


Figure 7.7. Encryption Page2

Convert binary value to matrix click on Matrix button and transpose the matrix by clicking on Trans- pose. In order to convert to New ASCII value click on Convert to Values (CT1).

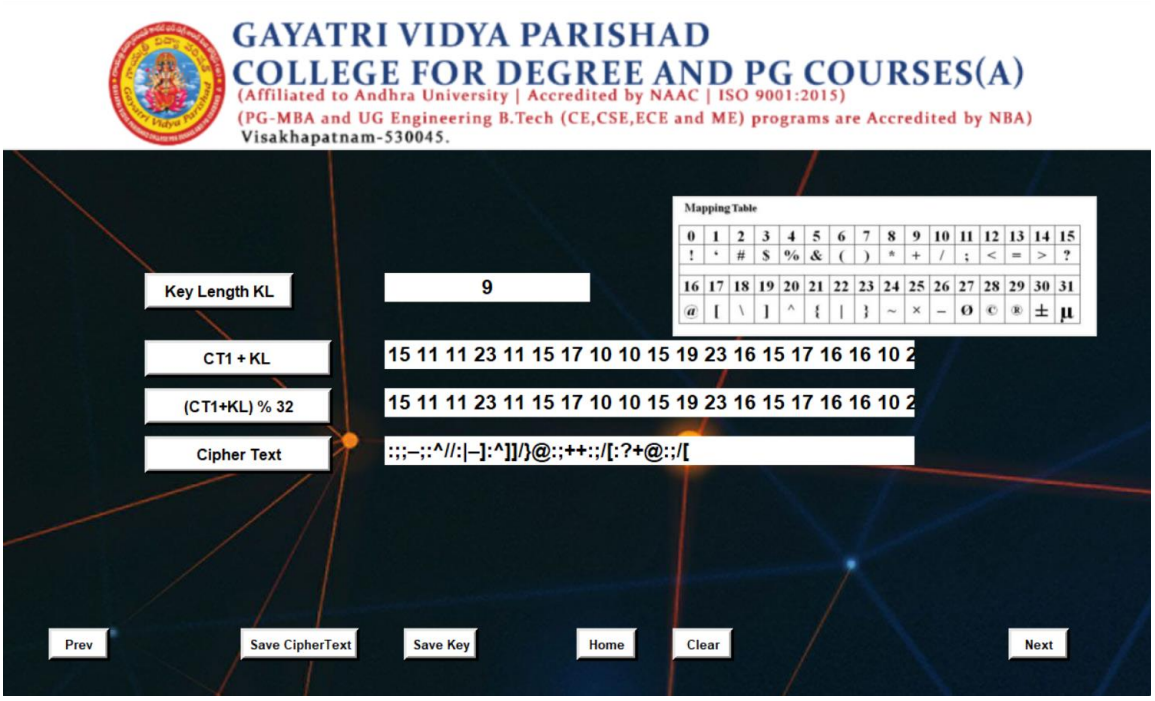


Figure 7.8. Encryption Page3

Click on Key Length Kl button it selects the value according to the length of Plain

text and KeyFinally sender click on **Cipher text** button it generate Cipher Text using Cyclic Mathematical Function. Save Cipher Text and Key

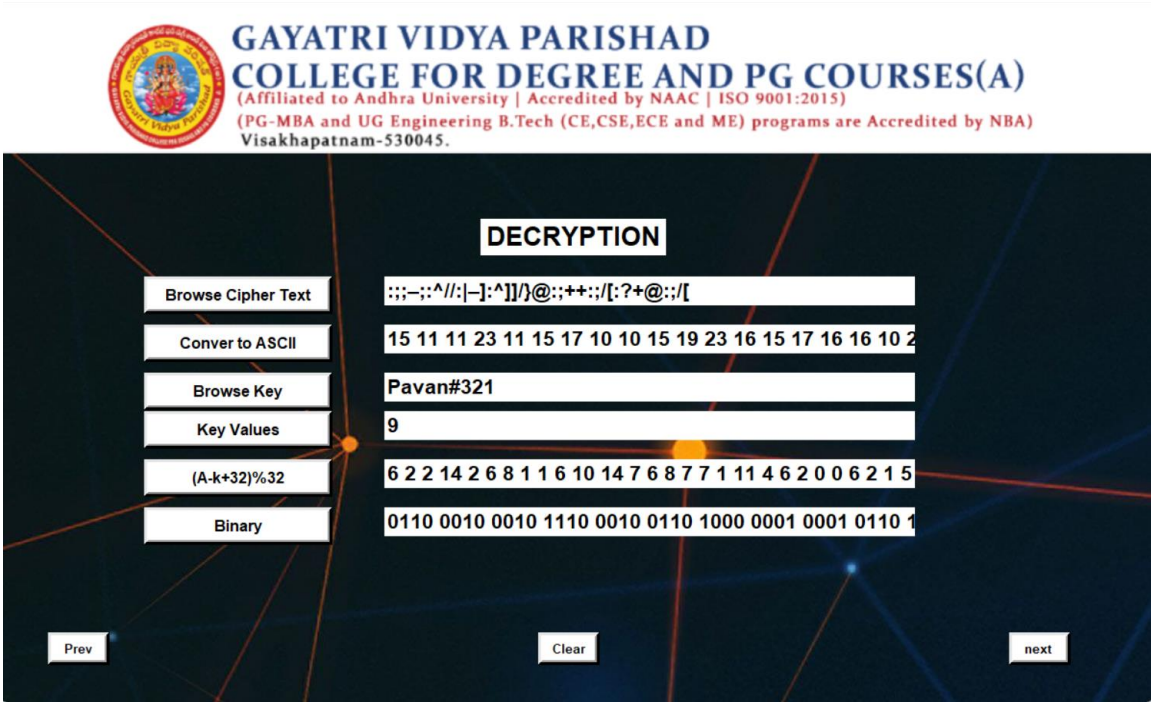


Figure 7.9. Decryption Page1

Here is Process to Decrypt the Message by the Receiver. Receiver need to browse the cipher text. and convert the cipher text to ASCII value by clicking on Convert to ASCII button and Receiver alsoneed to browse the key using Browse Key button and calculate cyclic mathematical function and generated binary value.

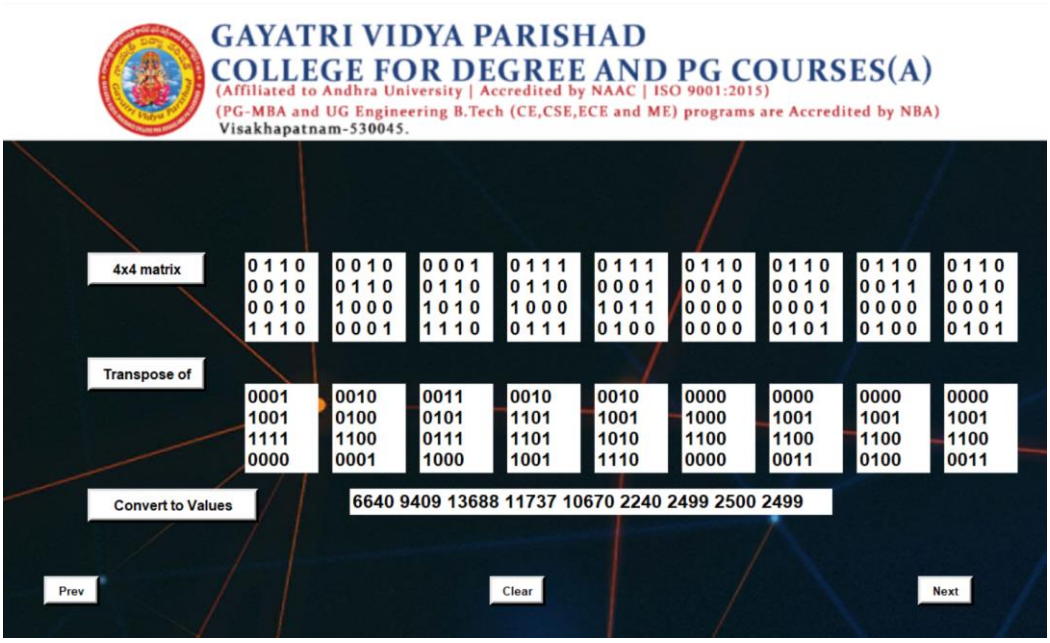


Figure 7.10. Decryption Page2

Click on 4*4 Matrix Button to change binary value to matrix form and also click on Transpose of but- ton to transpose the matrix and get the binary value.

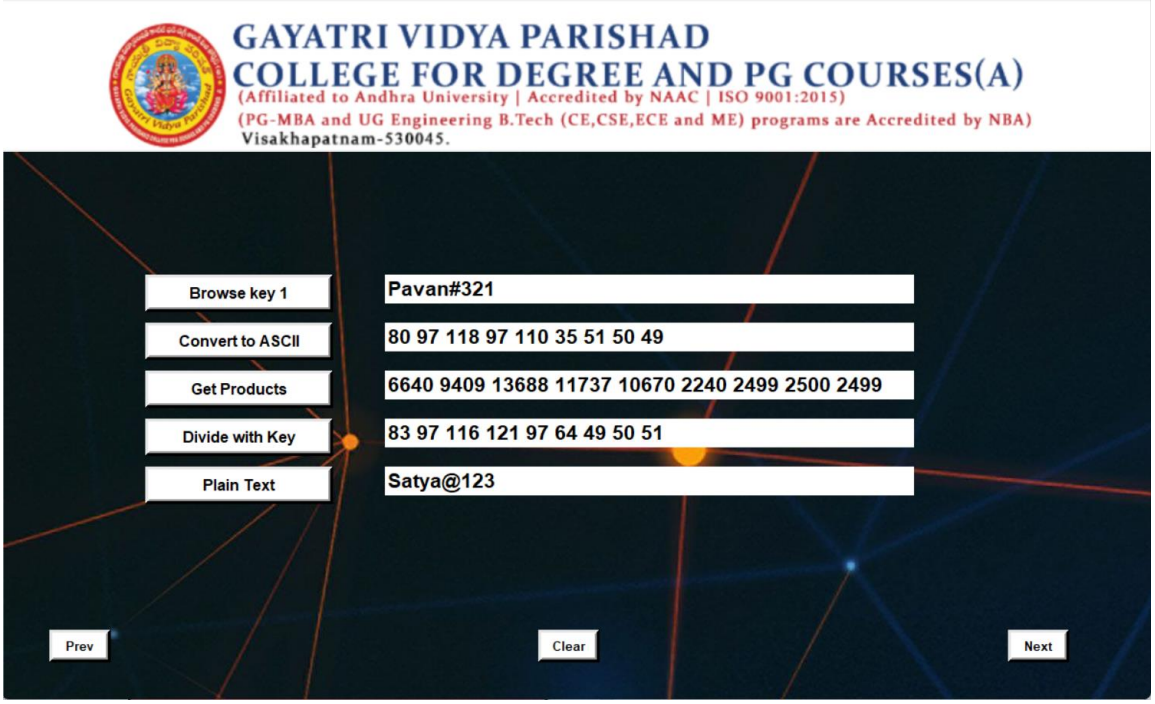


Figure 7.11. Decryption Page3

Receiver need to browse the key and convert to ASCII value by clicking on Convert to ASCII button. Product the key value with key by clicking on Get Products and divide decimal value with key value using Divide with Key button and Generate Plain text from the cipher text by Selecting Plain text button.

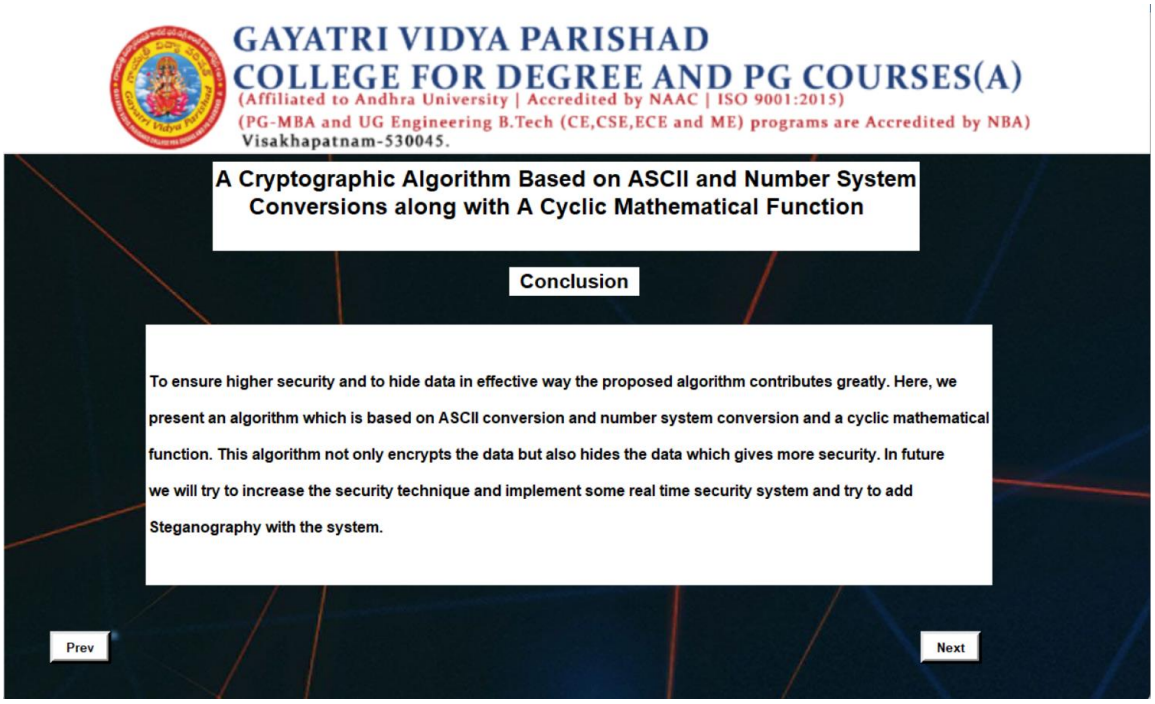


Figure 7.12. Conclusion Page

8.CONCLUSION

To ensure higher security and to hide data in effective way the proposed algorithm contributes greatly. Here, we present an algorithm which is based on ASCII conversion and number system conversion and a cyclic mathematical function. This algorithm not only encrypts the data but also hides the data which gives more security. In future we will try to increase the security technique and implement some real time security system and try to add Steganography with the system

9.REFERENCES

[1] C. E. Shannon, “Communication theory of secrecy systems∗ ,” Bell System Technical Journal, vol. 28, no. 4, pp. 656–715, 1949.

[2] K. H. Rosen, Elementary Number Theory and its Applications, Addison-Wesley, Boston, MA, USA, 5th edition, 2005.

[3] D. R. Stinson, Cryptography: Theory and Practice, Chapman and Hall/CRC, Boca Raton, FL, USA, 4th edition, 2018.

[4] D. B. West, Introduction to Graph Theory, Pearson, London, UK, 2nd edition, 2001.

[5] R. Frucht and F. Harary, “On the corona of two graphs,” Aequationes Math, vol. 4, pp. 322–325, 1970.

[6] R. L. Rivest, A. Shamir, and L. Adleman, “A method for obtaining digital signatures and public- key cryptosystems,” Communications of the ACM, vol. 21, no. 2, pp. 120–126, 1978.

[7] V. A. Ustimenko, “On graph-based cryptography and symbolic computations,” Serdica Journal of Computing, vol. 1, pp. 131–156, 2007.

[8] D. X. Charles, K. E. Lauter, and E. Z. Goren, “Cryptographic hash functions from expander graphs,” Journal of Cryptology , vol. 22, no. 1, pp. 93–113, 2009.

[9] P. L. K. Priyadarsini, “A survey on some applications of graph theory in cryptography,” Journal of Discrete Mathematical Sciences and Cryptography, vol. 18, no. 3, pp. 209–217, 2015.

[10] R. Selvakumar and N. Gupta, “Fundamental circuits and cutsets used in cryptography,” Journal of Discrete Mathematical Sciences and Cryptography, vol. 15, no. 4-5, pp. 287–301, 2012.

[11] P. Kedia and S. Agrawal, “Encryption using Venn-diagrams and graph,” International Journal of Advanced Computer Technology, vol. 4, no. 01, pp. 94–99, 2015.

[12] M. Yamuna and A. Elakkiya, “Data transfer using fundamental circuits,” International Journal of Computer and Modern Technology, vol. 2, no. 01, 2015.

[13] M. Yamuna and K. Karthika, “Data transfer using bipartite graphs,” International Journal of Advance Research in Science and Engineering, vol. 4, no. 02, pp. 128–131, 2015.

[14] W. Mahmoud and A. Etaiwi, “Encryption algorithm using graph theory,” Journal of Scientific Research and Reports , vol. 3, no. 19, pp. 2519–2527, 2014.

[15] B. R. Arunkumar, “Applications of Bipartite Graph in diverse fields including cloud computing,” International Journal of Modern Engineering Research, vol. 5, no. 7, p. 7, 2015.

[16] D. Sinha and A. Sethi, “Encryption using network and matrices through signed graphs,” International Journal of Computer Applications (0975-8887), vol. 138, no. 4, pp. 6–13, 2016.

[17] J. Hu, J. Liang, and S. Dong, “A bipartite graph propagation approach for mobile advertising fraud detection,” Mobile Information Systems, vol. 2017, p. 12, Article ID 6412521, 2017.

[18] A. Razaq, M. Awais Yousaf, U. Shuaib, N. Siddiqui, A. Ullah, and A. Waheed, “A novel construction of substitution box involving coset diagram and a bijective map,” Security and Communication Networks, vol. 2017, p. 16, Article ID 5101934, 2017.

[19] A. Razaq, H. Alolaiyan, M. Ahmad et al., “A novel method for generation of strong substitution- boxes based on coset graphs and symmetric groups,” IEEE Access, vol. 8, pp. 75473–75490, 2020.

[20] G. A Selim, “How to encrypt a graph,” International Journal of Parallel, Emergent and Distrib- uted Systems, vol. 35, no. 6, pp. 668–681, 2020

10.APPENDIX

10.1.LIST OF TABLES

10.2.List of Figures