KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY

(AN AUTONOMOUS INSTITUTION)

Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH,

Narayanaguda, Hyderabad–500029.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

LAB MANUAL

NAME OF THE LAB: Cryptography and Network Security

B.Tech IV YEAR I SEM (KR21)

ACADEMIC YEAR 2024-25

KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY

(AN AUTONOMOUS INSTITUTE)

Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH, Hyderabad

Certificate

This is to certify that following is a Bonafide Record of the workbook task done by

________________________________bearing Roll No____________________of

___________________Branch of________________year B.Tech Course in the

___________________Subject during the Academic year____________&__________
under our supervision.

Number of experiments completed:______________________________________

Signature of Staff Member Incharge
Mr B Pandya Naik

Signature of Head of the Dept.
Mr Para Upendar

Signature of Internal Examiner

Signature of External Examiner

KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY

(AN AUTONOMOUS INSTITUTE)

Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH, Hyderabad

Daily Laboratory Assessment Sheet

Name of the Lab: Name of the Student:

Class: HT.No:

S.No.	Name of the Experiment	Date	Observation Marks (3M)	Record Marks (4M)	Viva Voice Marks (3M)	Total Marks (10M)	Signature of Faculty
	TOTAL						

CONTENT

SYLLABUS

1.Write a Java program that contains a string (char pointer) with a value 'Hello World'. The program should XOR each character in this string with 0 and displays the result.

2. Write a java program that contains a string (char pointer) with a value 'Hello World'. The program should AND or and XOR each character in this string with 127 and displays the result.

3.Write a Java program to perform encryption and decryption using the following algorithms

a. Ceaser cipher

b. Substitution cipher

c. Hill Cipher

4.Write a JAVA program to implement the DES algorithm logic.

5.Write a JAVA program to implement the Blow fish algorithm logic.

6.Write a JAVA program to implement the Rijndael algorithm logic.

7.Write the RC4 logic in Java Using Java cryptography; encrypt the text “Hello world” using Blow fish. Create your own key using Java key tool.

8. Write a Java program to implement RSA algorithm.

9.Write a program to Calculate the message digest of a text using the SHA-1 algorithm.

10.Write a program to Calculate the message digest of a text using the MD5 algorithm.

TEXTBOOKS:

- Cryptography and Network Security-Principles and Practice: William Stallings, Pearson Education,7thEdition 2017.
- Cryptography and Network Security: Atul Kahate, Mc Graw Hill, 4thEdition 2019.

REFERENCE BOOKS:

- Cryptography and Network Security: CK Shyamala, N Harini, Dr.TR Padmanabhan, Wiley India, 2ndEdition 2011.
- Cryptography and Network Security: Forouzan Mukhopadhyay, Mc Graw Hill, 3rdEdition 2015.
- Information Security, Principles, and Practice: Mark Stamp, Wiley India. 2021.

.•Principles of Computer Security: WM. Arthur Conklin, Greg White, TMH. 4th Edition 2016.

- Introduction to Network Security: Neal Krawetz, CENGAGE Learning 2007
- Network Security and Cryptography: Bernard Menezes, CENGAGE Learning 2010.

Course Objectives: The course will help to

- Understand various cryptographic algorithms.
- Understand Key Exchange mechanism

Course Out comes: After learning the concepts of the course, the student is able to

- Student will be able to understand basic cryptographic algorithms, message and web authentication and securityissues.
- Ability to identify information system requirements for both of them such as client and server.

Software to beUsed:

OS, GCC Compiler

KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY

(AN AUTONOMOUS INSTITUTE)

Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH,
Narayanguda, Hyderabad – 500029

Department of Computer Science and Engineering

Vision of the Institution:

To be the fountain head of latest technologies, producing highly skilled, globally competent engineers.

Mission of the Institution:

- To provide a learning environment that inculcates problem solving skills, professional, ethical responsibilities, lifelong learning through multi modal platforms and prepare students to become successful professionals.
- To establish Industry Institute Interaction to make students ready for the industry.
- To provide exposure to students on latest hardware and software tools.
- To promote research based projects/activities in the emerging areas of technology convergence.
- To encourage and enable students to not merely seek jobs from the industry but also to create new enterprises
- To induce a spirit of nationalism which will enable the student to develop, understand India's challenges and to encourage them to develop effective solutions.
- To support the faculty to accelerate their learning curve to deliver excellent service to students

KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY

(AN AUTONOMOUS INSTITUTE)

Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH,
Narayanguda, Hyderabad – 500029

Department of Computer Science and Engineering

Vision of the Department:

To be among the region's premier teaching and research Computer Science and Engineering departments producing globally competent and socially responsible graduates in the most conducive academic environment.

Mission of the Department:

- To provide faculty with state-of-the-art facilities for continuous professional development and research, both in foundational aspects and of relevance to emerging computing trends.
- To impart skills that transform students to develop technical solutions for societal needs and inculcate entrepreneurial talents.
- To inculcate an ability in students to pursue the advancement of knowledge in various specializations of Computer Science and Engineering and make them industry-ready.
- To engage in collaborative research with academia and industry and generate adequate resources for research activities for seamless transfer of knowledge resulting in sponsored projects and consultancy.
- To cultivate responsibility through sharing of knowledge and innovative computing solutions that benefits the society-at-large.
- To collaborate with academia, industry and community to set high standards in academic excellence and in fulfilling societal responsibilities.

KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY
(AN AUTONOMOUS INSTITUTE)
Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH,
Narayanguda, Hyderabad – 500029

Department of Computer Science and Engineering

PROGRAM OUTCOMES (POs):

PO1: Engineering Knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineeringproblems.

PO2: Problem Analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics,natural sciences, and engineering sciences.

PO3: Design/Development of Solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4: Conduct Investigations of Complex Problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5: Modern Tool Usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6: The Engineer and Society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7: Environment and Sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9: Individual and Team Work: Function effectively as an individual, and as a member orleader in diverse teams, and in multidisciplinary settings.

PO10: Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11: Project Management and Finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12: Life-long Learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO1: An ability to analyze the common business functions to design and develop appropriate Computer Science solutions for social upliftments.

PSO2: Shall have expertise on the evolving technologies like Python, Machine Learning, Deep Learning, Internet of Things (IOT), Data Science, Full stack development, Social Networks, CyberSecurity, Big Data, Mobile Apps, CRM, ERP etc.

KESHAV MEMORIAL INSTITUTE OF TECHNOLOG
(AN AUTONOMOUS INSTITUTE)
Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH,
Narayanguda, Hyderabad – 500029

Department of Computer Science and Engineering

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PEO1: Graduates will have successful careers in computer related engineering fields or will beable to successfully pursue advanced higher education degrees.

PEO2: Graduates will try and provide solutions to challenging problems in their profession by applying computer engineering principles.

PEO3: Graduates will engage in life-long learning and professional development by rapidly adapting changing work environment.

PEO4: Graduates will communicate effectively, work collaboratively and exhibit high levels of professionalism and ethical responsibility.

PROGRAMS

Week1.

Write a Java program that contains a string (char pointer) with a value 'Hello World'. The program should XOR each character in this string with 0 and displays the result.

Program:

```
public class XORWithZero
 {
 public static void main(String[] args)
 {
 // Define the string
String text = "Hello World";
// Display the original string
System.out.println("Original String: " + text);
 // Perform XOR operation with 0 and display the result System.out.print("XOR with 0: ");
for (int i = 0; i<text.length(); i++)
 {
char c = text.charAt(i);
char xorResult = (char)(c ^ 0);
 // XOR with 0 System.out.print(xorResult);
 }
System.out.println();
 }
 }
```

Output:

```
Original String: Hello World
XOR with 0: Hello World
```

Week2

Write a java program that contains a string (char pointer) with a value 'Hello World'. The program should AND or and XOR each character in this string with 127 and displays the result.

Program:

```
public class BitwiseOperations
{
 public static void main(String[] args)
 {
 // Define the string
String text = "Hello World";
// Display the original string
System.out.println("Original String: " + text);
 // Perform AND operation with 127 and display the result System.out.print("AND with 127: ");
for (int i = 0; i<text.length(); i++)
 {
 char c = text.charAt(i);
char andResult = (char)(c & 127);
System.out.print(andResult);
 }
System.out.println();
// Perform XOR operation with 127 and display the result System.out.print("XOR with 127: ");
 for (int i = 0; i<text.length(); i++)
 {
 char c = text.charAt(i);
 char xorResult = (char)(c ^ 127);
System.out.print(xorResult);
 }
System.out.println();
 }
 }
```

Output:

```
Original String: Hello World
AND with 127: Hello World XOR with 127: 7xqq~?pq~q
```

Week3

WriteaJavaprogramtoperformencryptionanddecryptionusingthefollowingalgorithms:

a) **CeaserCipher**

b) **SubstitutionCipher**

c) **HillCipher**

a) CeaserCipher

Program:

```
import java.util.Scanner;

public class CaesarCipher {
    // Method to encrypt the message using Caesar Cipher
    public static String encrypt(String message, int shift) {
        StringBuilder result = new StringBuilder();

        for (int i = 0; i<message.length(); i++) {
            char ch = message.charAt(i);

            // Encrypt uppercase letters
            if (Character.isUpperCase(ch)) {
                char c = (char) (((int) ch + shift - 65) % 26 + 65);
result.append(c);
            }
            // Encrypt lowercase letters
            else if (Character.isLowerCase(ch)) {
                char c = (char) (((int) ch + shift - 97) % 26 + 97);
result.append(c);
            }
            // Keep non-alphabetic characters as they are
            else {
result.append(ch);
            }
        }
        return result.toString();
    }

    // Method to decrypt the message using Caesar Cipher
    public static String decrypt(String message, int shift) {
        return encrypt(message, 26 - shift); // Decrypt is reverse of encrypt with 26 - shift
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        // Input the message and shift value
System.out.print("Enter the message: ");
        String message = scanner.nextLine();

System.out.print("Enter the shift value (1-25): ");
        int shift = scanner.nextInt();

        // Input validation for shift value
        if (shift < 1 || shift > 25) {
System.out.println("Invalid shift value. Please enter a number between 1 and 25.");
            return;
        }
        // Encrypt the message
        String encryptedMessage = encrypt(message, shift);
System.out.println("Encrypted Message: " + encryptedMessage);
        // Decrypt the message
        String decryptedMessage = decrypt(encryptedMessage, shift);
System.out.println("Decrypted Message: " + decryptedMessage);

scanner.close();
    }
}
```

Output:

```
Enter the message: Hello World
Enter the shift value (1-25): 1
Encrypted Message: IfmmpXpsme
Decrypted Message: Hello World
```

b) SubstitutionCipher

Program:

```
import java.util.Scanner;


 public class SubstitutionCipher {
     // Alphabet used for reference
    private static final String ALPHABET = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";

    // Method to encrypt the message using Substitution Cipher
    public static String encrypt(String message, String key) {
        StringBuilder encryptedMessage = new StringBuilder();
        message = message.toUpperCase();
        for (int i = 0; i<message.length(); i++) {
            char currentChar = message.charAt(i);

            // If character is an alphabetic letter
            if (Character.isLetter(currentChar)) {
                int indexInAlphabet = ALPHABET.indexOf(currentChar);
                char encryptedChar = key.charAt(indexInAlphabet);
encryptedMessage.append(encryptedChar);
            } else {
                // Non-alphabet characters are added as-is
encryptedMessage.append(currentChar);
            }
        }
        return encryptedMessage.toString();
    }

    // Method to decrypt the message using Substitution Cipher
    public static String decrypt(String encryptedMessage, String key) {
        StringBuilder decryptedMessage = new StringBuilder();
encryptedMessage = encryptedMessage.toUpperCase();

        for (int i = 0; i<encryptedMessage.length(); i++) {
            char currentChar = encryptedMessage.charAt(i);   xi

            // If character is an alphabetic letter
            if (Character.isLetter(currentChar)) {
                int indexInKey = key.indexOf(currentChar);
                char decryptedChar = ALPHABET.charAt(indexInKey);
decryptedMessage.append(decryptedChar);
            } else {
                // Non-alphabet characters are added as-is
decryptedMessage.append(currentChar);
            }
        }
        return decryptedMessage.toString();
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Define the substitution key (26 unique uppercase letters)
        String key = "QWERTYUIOPLKJHGFDSAZXCVBNM";  // Example key, can be any permutation of 26 letters
System.out.println("Using substitution key: " + key);

        // Input the message to encrypt
System.out.print("Enter the message to encrypt: ");
        String message = scanner.nextLine();
        // Encrypt the message
        String encryptedMessage = encrypt(message, key);
System.out.println("Encrypted Message: " + encryptedMessage);

        // Decrypt the message
        String decryptedMessage = decrypt(encryptedMessage, key);
System.out.println("Decrypted Message: " + decryptedMessage);
        scanner.close();
    }
}
```

Output:

```
Using substitution key: QWERTYUIOPLKJHGFDSAZXCVBNM
Enter the message to encrypt: RAMA
Encrypted Message: SQJQ
Decrypted Message: RAMA
```

C) HillCipher

Program:

```
import java.util.Scanner;
public class HillCipher
 {
    // Function to perform matrix multiplication
    public static int[] matrixMultiply(int[][] keyMatrix, int[] messageVector) {
int[] result = new int[messageVector.length];
        for (int i = 0; i<keyMatrix.length; i++) {
            result[i] = 0;
            for (int j = 0; j <keyMatrix[i].length; j++) {
                result[i] += keyMatrix[i][j] * messageVector[j];
            }
            result[i] = result[i] % 26; // Perform modulo 26 operation
        }
        return result;
    }

    // Function to find the modular inverse of a number
    public static int modInverse(int a, int m) {
        a = a % m;
        for (int x = 1; x < m; x++) {
            if ((a * x) % m == 1) {
                return x;
            }
        }
        return 1;
    }

    // Function to calculate the inverse of a 2x2 matrix
    public static int[][] inverseKeyMatrix(int[][] keyMatrix) {
        int determinant = (keyMatrix[0][0] * keyMatrix[1][1] - keyMatrix[0][1] * keyMatrix[1][0]) % 26;
        determinant = (determinant + 26) % 26;
        int inverseDeterminant = modInverse(determinant, 26);
int[][] inverseMatrix = new int[2][2];
inverseMatrix[0][0] = (keyMatrix[1][1] * inverseDeterminant) % 26;
inverseMatrix[1][1] = (keyMatrix[0][0] * inverseDeterminant) % 26;
inverseMatrix[0][1] = (-keyMatrix[0][1] * inverseDeterminant + 26) % 26;
inverseMatrix[1][0] = (-keyMatrix[1][0] * inverseDeterminant + 26) % 26;

        return inverseMatrix;
    }
    // Function to convert a string into an integer vector
    public static int[] stringToVector(String text) {
int[] vector = new int[text.length()];
        for (int i = 0; i<text.length(); i++) {
            vector[i] = text.charAt(i) - 'A';
        }
        return vector;
    }

    // Function to convert an integer vector into a string
    public static String vectorToString(int[] vector) {
        StringBuilder text = new StringBuilder();
        for (int i : vector) {
text.append((char) (i + 'A'));
        }
        return text.toString();
    }

    // Function to encrypt the plaintext
    public static String encrypt(String plaintext, int[][] keyMatrix) {
int[] messageVector = stringToVector(plaintext);
int[] encryptedVector = matrixMultiply(keyMatrix, messageVector);
        return vectorToString(encryptedVector);
    }

    // Function to decrypt the ciphertext
    public static String decrypt(String ciphertext, int[][] keyMatrix) {
int[][] inverseMatrix = inverseKeyMatrix(keyMatrix);
int[] messageVector = stringToVector(ciphertext);
int[] decryptedVector = matrixMultiply(inverseMatrix, messageVector);
        return vectorToString(decryptedVector);
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Input: 2x2 key matrix
int[][] keyMatrix = new int[2][2];
System.out.println("Enter the 2x2 key matrix (values between 0 and 25):");
        for (int i = 0; i< 2; i++) {
            for (int j = 0; j < 2; j++) {
keyMatrix[i][j] = scanner.nextInt();
            }
        }

        // Input: plaintext (must be of length 2 for simplicity)
System.out.println("Enter the plaintext (length 2, uppercase letters only):");
        String plaintext = scanner.next().toUpperCase();

        // Encrypt the plaintext
        String ciphertext = encrypt(plaintext, keyMatrix);
System.out.println("Encrypted Text: " + ciphertext);

        // Decrypt the ciphertext
        String decryptedText = decrypt(ciphertext, keyMatrix);
System.out.println("Decrypted Text: " + decryptedText);

scanner.close();
    }
}
```

Output:

```
1 2
3 4
Enter the plaintext (length 2, uppercase letters only):
AB
Encrypted Text: CE
Decrypted Text: AY
```

Week4

Write a java program to implement the DES algorithm logic?

Program:

```
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;
import java.util.Base64;

public class DESExample {

    // Method to generate a secret key for DES
    public static SecretKeygenerateKey() throws Exception {
KeyGeneratorkeyGenerator = KeyGenerator.getInstance("DES");
keyGenerator.init(56); // DES uses a 56-bit key size
        return keyGenerator.generateKey();
    }

    // Method to encrypt data using the DES algorithm
    public static String encrypt(String plaintext, SecretKey key) throws Exception {
        Cipher cipher = Cipher.getInstance("DES");
cipher.init(Cipher.ENCRYPT_MODE,  key);
byte[] encryptedBytes = cipher.doFinal(plaintext.getBytes());
        return Base64.getEncoder().encodeToString(encryptedBytes);
    }

    // Method to decrypt data using the DES algorithm
    public static String decrypt(String ciphertext, SecretKey key) throws Exception {
        Cipher cipher = Cipher.getInstance("DES");
cipher.init(Cipher.DECRYPT_MODE,  key);
byte[] decryptedBytes = cipher.doFinal(Base64.getDecoder().decode(ciphertext));
        return new String(decryptedBytes);
    }
    public static void main(String[] args) {
        try {
            // Generate a secret key for DES
SecretKeysecretKey = generateKey();

            // Plain text to be encrypted
            String plaintext = "Hello, World!";
System.out.println("Original Text: " + plaintext);

            // Encrypt the plain text
            String encryptedText = encrypt(plaintext, secretKey);
System.out.println("Encrypted Text: " + encryptedText);

            // Decrypt the encrypted text
            String decryptedText = decrypt(encryptedText, secretKey);
System.out.println("Decrypted Text: " + decryptedText);

        } catch (Exception e) {
e.printStackTrace();
        }
    }
}
```

Output:

```
Original Text: Hello, World!
Encrypted Text: wEm+7nd6ij+aOwmOMdQORQ==
Decrypted Text: Hello, World!
```

Week5

Write a java program to implement the Blowfish algorithm logic?

Program:

```
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;
import java.util.Base64;

public class BlowfishExample {

    // Method to generate a secret key for Blowfish
    public static SecretKeygenerateKey(int keySize) throws Exception {
KeyGeneratorkeyGenerator = KeyGenerator.getInstance("Blowfish");
keyGenerator.init(keySize); // keySize can be between 32 and 448 bits
        return keyGenerator.generateKey();
    }

    // Method to encrypt data using the Blowfish algorithm
    public static String encrypt(String plaintext, SecretKey key) throws Exception {
        Cipher cipher = Cipher.getInstance("Blowfish");
cipher.init(Cipher.ENCRYPT_MODE,  key);
byte[] encryptedBytes = cipher.doFinal(plaintext.getBytes());
        return Base64.getEncoder().encodeToString(encryptedBytes);
    }

    // Method to decrypt data using the Blowfish algorithm
    public static String decrypt(String ciphertext, SecretKey key) throws Exception {
        Cipher cipher = Cipher.getInstance("Blowfish");
cipher.init(Cipher.DECRYPT_MODE,  key);
byte[] decryptedBytes = cipher.doFinal(Base64.getDecoder().decode(ciphertext));
        return new String(decryptedBytes);
    }

    public static void main(String[] args) {
        try {
            // Generate a secret key for Blowfish
SecretKeysecretKey = generateKey(128); // You can specify a key size between 32 and 448 bits
           // Plain text to be encrypted
            String plaintext = "Hello, World!";
System.out.println("Original Text: " + plaintext);

            // Encrypt the plain text
            String encryptedText = encrypt(plaintext, secretKey);
System.out.println("Encrypted Text: " + encryptedText);

            // Decrypt the encrypted text
            String decryptedText = decrypt(encryptedText, secretKey);
System.out.println("Decrypted Text: " + decryptedText);

        } catch (Exception e) {
e.printStackTrace();
        }
    }
}
```

Out Put:

```
•   Original Text: Hello, World!
•   Encrypted Text: XNcjWiCOqfEnr6Fjc8GViw==

    Decrypted Text: Hello, World!
```

Week6

Write a java program to implement the Rijndael algorithm logic?

Program:

```
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;
import java.util.Base64;

public class AESExample {

    // Method to generate a secret key
    public static SecretKeygenerateKey(int n) throws Exception {
KeyGeneratorkeyGenerator = KeyGenerator.getInstance("AES");
keyGenerator.init(n);
        return keyGenerator.generateKey();
    }

    // Method to encrypt data using the AES algorithm
    public static String encrypt(String plaintext, SecretKey key) throws Exception {
        Cipher cipher = Cipher.getInstance("AES");
cipher.init(Cipher.ENCRYPT_MODE,  key);
byte[] encryptedBytes = cipher.doFinal(plaintext.getBytes());
        return Base64.getEncoder().encodeToString(encryptedBytes);
    }

    // Method to decrypt data using the AES algorithm
    public static String decrypt(String ciphertext, SecretKey key) throws Exception {
        Cipher cipher = Cipher.getInstance("AES");
cipher.init(Cipher.DECRYPT_MODE,  key);
byte[] decryptedBytes = cipher.doFinal(Base64.getDecoder().decode(ciphertext));
        return new String(decryptedBytes);
    }
    public static void main(String[] args) {
        try {
            // Generate a secret key for AES
SecretKeysecretKey = generateKey(128);

            // Plain text to be encrypted
            String plaintext = "Hello, World!";
System.out.println("Original Text: " + plaintext);

            // Encrypt the plain text
            String encryptedText = encrypt(plaintext, secretKey);
System.out.println("Encrypted Text: " + encryptedText);

            // Decrypt the encrypted text
            String decryptedText = decrypt(encryptedText, secretKey);
System.out.println("Decrypted Text: " + decryptedText);

        } catch (Exception e) {
e.printStackTrace();
        }
    }
}
```

OutPut:

```
Original Text: Hello, World!
Encrypted Text: AYss0loz6Ml+kWPZ8lj6bA==
Decrypted Text: Hello, World!
```

Week7

Write a java program the RC4 logic using cryptography; encrypt the text "Hello World" using Blowfish. Create your own key using java key tool?

Program:

```
import java.util.Scanner;

public class RC4 {
    private byte[] S = new byte[256];
    private int x = 0;
    private int y = 0;

    // Constructor to initialize the key
    public RC4(byte[] key) {
init(key);
    }

    // Initialize the permutation in the array S
    private void init(byte[] key) {
        int keyLength = key.length;
        for (int i = 0; i< 256; i++) {
            S[i] = (byte) i;
        }
        int j = 0;
        for (int i = 0; i< 256; i++) {
            j = (j + S[i] + key[i % keyLength]) & 0xFF;
swap(i, j);
        }
    }

    // Swap elements in the array S
    private void swap(int i, int j) {
        byte temp = S[i];
        S[i] = S[j];
        S[j] = temp;
    }

    // Generate the key stream and perform encryption/decryption
    public byte[] encrypt(byte[] plaintext) {
byte[] ciphertext = new byte[plaintext.length];
        for (int i = 0; i<plaintext.length; i++) {
            ciphertext[i] = (byte) (plaintext[i] ^ keyItem());
        }
        return ciphertext;
    }

    // Generate the next byte of the key stream
    private byte keyItem() {
        x = (x + 1) & 0xFF;
        y = (y + S[x]) & 0xFF;
swap(x, y);
        return S[(S[x] + S[y]) & 0xFF];
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
System.out.println("Enter a key for RC4 encryption (e.g., mysecretkey):");
        String keyString = scanner.nextLine();
byte[] key = keyString.getBytes();

        RC4 rc4 = new RC4(key);

        String plaintext = "Hello World";
System.out.println("Original Text: " + plaintext);

byte[] ciphertext = rc4.encrypt(plaintext.getBytes());
System.out.println("Encrypted Text: " + new String(ciphertext));
```

```
        // Decrypting the ciphertext
byte[] decryptedText = rc4.encrypt(ciphertext); // RC4 is symmetric, so encryption and decryption are the
same
System.out.println("Decrypted Text: " + new String(decryptedText));

scanner.close();
    }
}
```

Output:

```
Enter a key for RC4 encryption (e.g., mysecretkey):
1
Original Text: Hello World
Encrypted Text: (•??g?_x001D_P
Decrypted Text:?5??.1LW.?V
```

Week8

Write a java program to implement RSA algorithm?

Program:

```
import java.math.BigInteger;
import java.security.KeyFactory;
import java.security.KeyPair;
import java.security.KeyPairGenerator;
import java.security.PrivateKey;
import java.security.PublicKey;
import  java.security.spec.RSAPrivateKeySpec;
import  java.security.spec.RSAPublicKeySpec;
import javax.crypto.Cipher;

public class RSAExample {

    public static void main(String[] args) {
        try {
            // Generate RSA key pair
KeyPairGeneratorkeyPairGenerator = KeyPairGenerator.getInstance("RSA");
keyPairGenerator.initialize(2048); // Key size (2048 bits for strong security)
KeyPairkeyPair = keyPairGenerator.generateKeyPair();
PublicKeypublicKey = keyPair.getPublic();
PrivateKeyprivateKey = keyPair.getPrivate();

            // Print the key details
printKeyDetails(publicKey, privateKey);

            // Text to be encrypted
            String plaintext = "Hello, RSA!";
System.out.println("Original Text: " + plaintext);

            // Encrypt the text using the public key
byte[] encryptedText = encrypt(plaintext, publicKey);
System.out.println("Encrypted Text: " + new String(encryptedText));

            // Decrypt the text using the private key
            String decryptedText = decrypt(encryptedText, privateKey);
System.out.println("Decrypted Text: " + decryptedText);

        } catch (Exception e) {
e.printStackTrace();
        }
    }

    // Method to encrypt data using RSA
    public static byte[] encrypt(String plaintext, PublicKeypublicKey) throws Exception {
        Cipher cipher = Cipher.getInstance("RSA");
cipher.init(Cipher.ENCRYPT_MODE, publicKey);
        return cipher.doFinal(plaintext.getBytes());
    }

    // Method to decrypt data using RSA
    public static String decrypt(byte[] ciphertext, PrivateKeyprivateKey) throws Exception {
        Cipher cipher = Cipher.getInstance("RSA");
cipher.init(Cipher.DECRYPT_MODE, privateKey);
byte[] decryptedBytes = cipher.doFinal(ciphertext);
        return new String(decryptedBytes);
    }

    // Method to print the details of the RSA keys
    public static void printKeyDetails(PublicKeypublicKey, PrivateKeyprivateKey) throws Exception {
KeyFactorykeyFactory = KeyFactory.getInstance("RSA");
RSAPublicKeySpecpublicKeySpec = keyFactory.getKeySpec(publicKey, RSAPublicKeySpec.class);
RSAPrivateKeySpecprivateKeySpec = keyFactory.getKeySpec(privateKey, RSAPrivateKeySpec.class);

System.out.println("Public Key Modulus: " + publicKeySpec.getModulus());
System.out.println("Public Key Exponent: " + publicKeySpec.getPublicExponent());
System.out.println("Private Key Modulus: " + privateKeySpec.getModulus());
System.out.println("Private Key Exponent: " + privateKeySpec.getPrivateExponent());
    }
}
```

OutPut:

```
Public Key Modulus:
3103851071182999980178437253194698335813785499008809099163739014841928787450915373262
2733244983277737425077411478209840595153629807540368384755355846411296224073599759101
4283839215888404330145110999789082920861995618207537028017885177596642270140469172656
9499028635733826672972928975918170582429858963639265389821540710762951916896996052000
4554964847765472617712501675590258013735930876058192220554294491498130406733122672893
2862356499895135946154217300881512447104893817049022278622079639340768788655414924344
1783437961075350888073421578047323323608141167938368803938146508864759956444041221451
6283780812790414217103
Public Key Exponent: 65537
Private Key Modulus:
3103851071182999980178437253194698335813785499008809099163739014841928787450915373262
2733244983277737425077411478209840595153629807540368384755355846411296224073599759101
4283839215888404330145110999789082920861995618207537028017885177596642270140469172656
9499028635733826672972928975918170582429858963639265389821540710762951916896996052000
4554964847765472617712501675590258013735930876058192220554294491498130406733122672893
2862356499895135946154217300881512447104893817049022278622079639340768788655414924344
1783437961075350888073421578047323323608141167938368803938146508864759956444041221451
6283780812790414217103
Private Key Exponent:
8087242457164793576380822212727568973611273201561625368466668815698121057496045121660
5244806954003180534754700154403045303087322061363707606097693887928848485936309182052
2744531859943298036461527905213601470412047725179959761727483298387210797704907394645
7842960920364194923125217082377692122857663863634907791753677319972893013752261852766
4763121751321039733087486021668971272031812561097418111308594124185992951219182978783
4840154912876213641294508691780263690089327177664990807100164009896785992599333694408
4150579568040937047234823727319352079512156028482589697469338705761831391191660030457
8570293849554660716l7
Original Text: Hello, RSA!
Encrypted Text: ??_x0019_????~?6?91DQv<?_x0012_)xx
```

```
(C"???XF?K?J?%?A?o???_x001D_-B61??1N
?O
?2?
?????K???0
Decrypted Text: Hello, RSA!
```

Week9

Write a java program to calculate the message digest of text using the SHA-1 algorithm?

Program:

```
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

public class SHA1DigestExample {

    public static void main(String[] args) {
        String input = "Hello, World!";  // The input text for which SHA-1 hash is to be calculated

        try {
            // Create a MessageDigest instance for SHA-1
MessageDigest md = MessageDigest.getInstance("SHA-1");

            // Update the MessageDigest with the bytes of the input string
md.update(input.getBytes());

            // Perform the hash computation and get the resulting byte array
byte[] digest = md.digest();

            // Convert the byte array into a hexadecimal string
            StringBuilder sb = new StringBuilder();
            for (byte b : digest) {
sb.append(String.format("%02x", b));
            }
            // Print the resulting SHA-1 hash
System.out.println("SHA-1 Digest: " + sb.toString());

        } catch (NoSuchAlgorithmException e) {
System.out.println("SHA-1 algorithm not found: " + e.getMessage());
        }
    }
}                                                                    xx
```

OutPut:

```
SHA-1 Digest: 0a0a9f2a6772942557ab5355d76af442f8f65e01
```

Week10

Write a java program to calculate the message digest of text using the MD5 algorithm?

Program:

```
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

public class MD5DigestExample {

    public static void main(String[] args) {
        String input = "Hello, World!";  // The input text for which MD5 hash is to be calculated

        try {
            // Create a MessageDigest instance for MD5
MessageDigest md = MessageDigest.getInstance("MD5");

            // Update the MessageDigest with the bytes of the input string
md.update(input.getBytes());
            // Perform the hash computation and get the resulting byte array
byte[] digest = md.digest();

            // Convert the byte array into a hexadecimal string
            StringBuilder sb = new StringBuilder();
            for (byte b : digest) {
sb.append(String.format("%02x", b));
            }
            // Print the resulting MD5 hash
System.out.println("MD5 Digest: " + sb.toString());

        } catch (NoSuchAlgorithmException e) {
System.out.println("MD5 algorithm not found: " + e.getMessage());
        }
    }
}
```

xx

OutPut:

```
MD5 Digest: 65a8e27d8879283831b664bd8b7f0ad4
```

1. Define Cryptography and its benefits?
2. What are the few major applications of cryptography in the modern world?
3. What is decryption? What is its need?
4. What type of information can be secured with Cryptography?
5. What exactly do you know about RSA?
6. What is the Digital Signature Algorithm?
7. Differentiate symmetric and asymmetric encryption?
8. What is the Caesar cipher?
9. What is plain text?
10. What is cipher text?
11. What are the mathematical algorithms used in symmetric cryptography?
12. What are the mathematical algorithms used in asymmetric cryptography?
13. What is the difference between a private key and a public key?
14. What is a block cipher?
15. What is Transposition Ciphers?
16. What is the International Data Encryption Algorithm (IDEA)?
17. How is a Key Distribution Center (KDC) used?
18. What are the specific components of the Public Key Infrastructure (PKI)?