Department of Computer Science and Engineering

KPR Institute of Engineering and Technology

B.E. – COMPUTER SCIENCE AND ENGINEERING

LABORATORY RECORD

U19CS703 – SECURITY LABORATORY

(Regulation 2019)

NOV-DEC 2022

KPR INSTITUTE OF ENGINEERING AND TECHNOLOGY

(Autonomous)

COIMBATORE – 641 407

LABORATORY RECORD

Name : …….

Roll Number : …….

Subject Code & Title: …….

Department : …….

Year & Semester : …….

This is the certified record of work done by…….

Register Number……

Faculty In- Charge **Head of the Department**

Place:

Date:

He/ She has submitted the record for the End Semester Practical Examination held on ……

Internal Examiner External Examiner

INDEX

Ex. No.	Date	Experiment Name	Page No	Marks	Sign
1		Perform encryption, decryption using the following substitution techniques (i)Ceaser cipher,(ii)Playfair cipher,(iii)Hill Cipher (iv)Vigenere cipher			
2		Perform encryption, decryption using following transposition techniques (i) Rail fence (ii) Row & column transformation			
3		Apply DES algorithm for practical application			
4		Apply AES algorithm for practical application			
5		Suppose Alice wants her Friends to encrypt email messages before sending them to her. Computers represent text as long numbers(01 for "A", 02 for "B" and so on), so an email message is just a very big number. Implement RSA encryption Scheme to encrypt and then decrypt electronic communications			
6		The first requirement is for institutions, governments or enterprises that need to assure their constituents that forms or documents are authentic and have maintained their integrity. The second requirement is for employees, customers, or citizens that need to approval or acknowledgement that a document or form has been read and approval or agreed to in principal. Implement the solution for above said requirements.			
7		Demonstrate intrusion detection system(ids) using any tool e.g., Snort or any other s/w			
8		Calculate the message digest of a text using the SHA-1 Algorithm in Java			
9		Case Study - AES			
Total Marks					

Vision of the Institution

To become a premier institute of academic excellence by imparting technical, intellectual and professional skills to students for meeting the diverse needs ofthe industry, society, the nation and the world at large.

Mission of the Institution

1. Commitment to offer value-based education and enhancement of practicalskills
2. Continuous assessment of teaching and learning process through scholarlyactivities
3. Enriching research and innovative activities in collaboration with industry and institute of repute
4. Ensuring the academic process to uphold culture, ethics and social responsibility

Vision of the Department

To foster the students by providing learner centric teaching environment, continuous learning, research and development to become thriving professionals and entrepreneurs to excel in the field of computer science and contribute to the society.

Mission of the Department

- Providing value-based education and contented learning experience to the students
- Educating the students with the state of art technologies and cultivating their proficiency in analytical and designing skills
- Enabling the students to achieve a successful career in Computer Science and Engineering or related fields to meet the changing needs of variousstakeholders
- Guiding the students in research by nurturing their interest in continuouslearning towards serving the society and the country

Ex.no:1(a) Date:	**Encryption and Decryption Using Ceaser Cipher**

AIM:

To encrypt and decrypt the given message by using Ceaser Cipher encryption algorithm.

ALGORITHMS:

1. In Ceaser Cipher each letter in the plaintext is replaced by a letter some fixed number of positions down the alphabet.
2. Enter the input.
3. Encryption of a letter x by a shift n can be described mathematically as,
   $$En(x) = (x + n) \bmod 26$$
4. Decryption is performed similarly,
   $$Dn(x) = (x - n) \bmod 26$$

PROGRAM:

CaesarCipher.java

```
class caesarCipher {
public static String encode(String enc, int offset) {
offset = offset % 26 + 26;
StringBuilder encoded = new StringBuilder();
 for(char i : enc.toCharArray()) {
if (Character.isLetter(i)) {
if (Character.isUpperCase(i)) {
encoded.append((char) ('A' + (i - 'A' + offset) % 26));
} else {
encoded.append((char) ('a' + (i - 'a' + offset) % 26));
}
} else {
encoded.append(i);
}
```

```
return encoded.toString();
}
public static String decode(String enc, int offset) {return encode(enc, 26 - offset);
}
public static void main(String[] args) throws java.lang.Exception {
String msg= "GANESHA MOORTHI";
System.out.println("Simulating Caesar Cipher\n      ");
System.out.println("Input : " + msg);
System.out.printf("EncryptedMessage : ");
System.out.println(caesarCipher.encode(msg, 3));
System.out.printf("Decrypted Message : ");
System.out.println(caesarCipher.decode(caesarCipher.encode(msg, 3), 3));
}
}
```

OUTPUT:

SimulatingCaesarCipher

Input : Ganesha Moorthi
Encrypted Message : Jdqhvkd Prruwkl
Decrypted Message : Ganesha Moorthi

RESULT:
Thus the program for ceaser cipher encryption and decryption algorithm has been implemented and the output verified successfully.

Ex. No : 1(b)Date:	Playfair Cipher

AIM:

To implement a program to encrypt a plain text and decrypt a cipher text using play fair Cipher substitution technique.

ALGORITHM:

1. To encrypt a message, one would break the message into digrams (groups of 2letters)
2. For example, "HelloWorld" becomes "HE LL OW OR LD".
3. These digrams will be substituted using the key table.
4. Since encryption requires pairs of letters, messages with an odd number of characters usually append an uncommon letter, such as "X", to completethefinal digram.
5. The two letters of the digram are considered opposite corners of a rectangle in the key table. To perform the substitution, apply the following 4 rules, in order, toeach pair of letters in the plaintext:

PROGRAM:

playfairCipher.java

```
import java.awt.Point;
class playfairCipher {
    private static char[][] charTable;
    privatestatic Point[] positions;
    private static String prepareText(String s, boolean chgJtoI)
      {          s =s.toUpperCase().replaceAll("[^A-Z]", "");
      return chgJtoI ? s.replace("J", "I") : s.replace("Q", "");
    }
    private static void createTbl(String key, boolean
      chgJtoI) {charTable = new char[5][5];
      positions = new Point[26];
      String s = prepareText(key + "ABCDEFGHIJKLMNOPQRSTUVWXYZ",
      chgJtoI);
      int len = s.length();
      for (int i = 0, k = 0; i < len; i++) { char
        c = s.charAt(i);
```

```
        if (positions[c - 'A'] ==
            null) { charTable[k / 5][k
            % 5] = c;
            positions[c - 'A'] = new Point(k % 5, k
            / 5);k++;
        }
    }
}

private static String codec(StringBuilder txt, int dir) {
int len= txt.length();
    for (int i = 0; i < len; i += 2) {
    char a= txt.charAt(i);
        char b = txt.charAt(i + 1);
        int row1 = positions[a - 'A'].y;
        int row2 = positions[b -'A'].y;
        int col1 = positions[a - 'A'].x;
        int col2 = positions[b - 'A'].x;
        if (row1 == row2) {
            col1 = (col1 + dir) %
            5;col2 = (col2 + dir)
            % 5;
        } else if (col1 == col2) {
            row1 = (row1 + dir) % 5;
            row2= (row2 + dir) % 5;
        } else {
            int tmp = col1;
            col1 = col2;
            col2= tmp;
        }
        txt.setCharAt(i,
        charTable[row1][col1]); txt.setCharAt(i
        + 1, charTable[row2][col2]);
    }
    return txt.toString();
}
private static String encode(String s) {
    StringBuilder sb = new StringBuilder(s);
```

```
        for(int i = 0; i < sb.length(); i += 2) {
            if (i == sb.length() - 1) {
                sb.append(sb.length() % 2 == 1 ? 'X' : "");
            } else if (sb.charAt(i) == sb.charAt(i +
                1)) {sb.insert(i + 1, 'X');
            }
        }
        return codec(sb, 1);
    }
    private static String decode(String s) {
        returncodec(new StringBuilder(s), 4);
    }
    public static void main(String[] args) throws java.lang.Exception {
         String key= "CSE";
        String txt = " GANESHA MOORTHI"; /* make sure string length is even */ /*
        change J to I */boolean chgJtoI = true;
        createTbl(key, chgJtoI);
        String enc = encode(prepareText(txt, chgJtoI));
        System.out.println("Simulating Playfair Cipher\n ---------------------- ");
        System.out.println("Input Message : " + txt);
        System.out.println("Encrypted Message : " + enc);
        System.out.println("Decrypted Message : " +
        decode(enc));
    }
}
```

OUTPUT:

```
SimulatingPlayfairCipher
-----------------------------
Input Message : GaneshaMoorthi
Encrypted Message : HEMAAFENMZMUYNGZ
Decrypted Message : GANESHAMOORTHI
```

RESULT:

Thus the program for playfair cipher encryption and decryption algorithm hasbeen implemented and the output verified successfully

Ex. No : 1(c)Date:	Hill Cipher

AIM:

To implement a program to encrypt and decrypt using the Hill cipher substitution technique

ALGORITHM:

1. In the Hill cipher Each letter is represented by a number modulo 26.
2. To encrypt a message, each block of n letters is multiplied by an invertible ***n x n*** matrix, again ***modulus 26***.
3. To decrypt the message, each block is multiplied by the inverse of the matrixused for encryption.
4. The matrix used for encryption is the cipher key, and it should be chosenrandomly from the ***set of invertible n × n matrices (modulo 26).***
5. The cipher can, be adapted to an alphabet with any number of letters.
6. All arithmetic just needs to be done modulo the number of letters instead ofmodulo 26.

PROGRAM:

HillCipher.java

```
class hillCipher
{
   /* 3x3 key matrix for 3 characters at once */
   public static int[][] keymat = new int[][] { { 1, 2, 1 }, { 2, 3, 2 },{ 2, 2, 1 } }; /* key
   inverse matrix */
   public static int[][] invkeymat = new int[][] { { -1, 0, 1 }, { 2, -1, 0 }, { -2, 2, -1} };
   public static String key = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
   private static String encode(char a, char b, char c) {
       String ret = "";
      int x, y, z;
      int posa = (int) a - 65;
      intposb = (int) b - 65;
      int posc = (int) c - 65;
      x = posa * keymat[0][0] + posb * keymat[1][0] + posc *keymat[2][0]; y = posa*
      keymat[0][1] + posb * keymat[1][1] + posc * keymat[2][1]; z = posa *
      keymat[0][2] + posb * keymat[1][2] + posc * keymat[2][2]; a = key.charAt(x
      % 26);
```

```
    b = key.charAt(y % 26);
    c = key.charAt(z % 26);
    return ret = "" + a + b
    + c;
     }
private static String decode(char a, char b, char c) {
    String ret = "";
    int x, y, z;
    int posa = (int) a - 65;
    int posb = (int) b -65;
    int posc = (int) c - 65;
    x = posa*invkeymat[0][0] + posb*invkeymat[1][0] + posc *invkeymat[2][0];
    y = posa * invkeymat[0][1] + posb * invkeymat[1][1] + posc *invkeymat[2][1];
   z = posa * invkeymat[0][2] + posb * invkeymat[1][2] + posc*
    invkeymat[2][2];
    a = key.charAt((x % 26 < 0) ? (26 + x % 26) : (x % 26));
   b = key.charAt((y % 26 < 0) ? (26 + y % 26) : (y % 26));
   c =key.charAt((z % 26 < 0) ? (26 + z % 26) : (z % 26));
       ret = ""+ a + b + c;
   return ret;
}
public static void main(String[] args) throws java.lang.Exception {
     Stringmsg;
    String enc = "";
    String dec = "";
    int n;
    msg = ("GANESHA MOORTHI ");
    System.out.println("simulation of Hill Cipher\n------------------------ ");
    System.out.println("Input message : " + msg);
     msg= msg.toUpperCase();
    msg = msg.replaceAll("\\s", "");
    n = msg.length() % 3;
    if (n != 0) { for(int i = 1; i <= (3 - n);
        i++) {
           msg += 'X';
        }
```

```
        }
        System.out.println("padded message : " +
        msg);char[] pdchars = msg.toCharArray();
        for (int i = 0; i < msg.length(); i += 3) {
            enc += encode(pdchars[i], pdchars[i + 1], pdchars[i + 2]);
        }
        System.out.println("encoded message : " + enc);
        char[]dechars = enc.toCharArray();
        for (int i = 0; i < enc.length(); i += 3) {
            dec += decode(dechars[i], dechars[i + 1], dechars[i + 2]);
        }
        System.out.println("decoded message : " + dec);
    }
}
```

OUTPUT:

```
simulation ofHillCipher
---------------------------------
Input message : GaneshaMoorthi
padded message :GANESHAMOORTHIX
encoded message : GMTCYVAMMINPRGU
decoded message : GANESHAMOORTHIX
```

RESULT:

Thus the program for hill cipher encryption and decryption algorithm has been implemented and the output verified successfully

Ex. No :1(d) Date:	Vigenere Cipher

AIM:

To implement a program for encryption and decryption using vigenere cipher substitution technique

ALGORITHM:

1. The Vigenere cipher is a method of encrypting alphabetic text by usingaseries of different Caesar ciphers based on the letters of a keyword.
2. It is a simple form of *polyalphabetic* substitution.
3. To encrypt, a table of alphabets can be used, termed a Vigenere square, orVigenere table.
4. It consists of the alphabet written out 26 times in different rows, each alphabet shifted cyclically to the left compared to the previous alphabet,corresponding to the 26 possible Caesar ciphers.
5. At different points in the encryption process, the cipher uses a differentalphabet from one of the rows used.
6. The alphabet at each point depends on a repeating keyword.

PROGRAM:

vigenereCipher.java

```
public class vigenereCipher {
   static String encode(String text, final String
      key) {String res = "";
      text = text.toUpperCase();
      for (int i = 0, j = 0; i < text.length(); i++) {
         charc = text.charAt(i);
         if (c < 'A' || c > 'Z') { continue;
         }
         res += (char) ((c + key.charAt(j) - 2 * 'A') % 26 + 'A');
          j =++j % key.length();
      }
      return res;
   }
```

```
    static String decode(String text, final String key) {
     String res = "";
     text = text.toUpperCase();
     for (int i = 0, j = 0; i < text.length(); i++) {
     charc = text.charAt(i);
        if (c < 'A' || c > 'Z') {
         continue;
        }
        res += (char) ((c - key.charAt(j) + 26) % 26 + 'A');
         j =++j % key.length();
    }
    return res;
  }
  public static void main(String[] args) throws java.lang.Exception {
     String key= "VIGENERECIPHER";
    String msg = " GANESHA MOORTHI ";
    System.out.println("Simulating Vigenere Cipher\n---------- ");
    System.out.println("Input Message : " + msg);
    Stringenc = encode(msg, key);
    System.out.println("Encrypted Message : " +
    enc);
    System.out.println("Decrypted Message : " + decode(enc, key));
  }
}
```

OUTPUT:

```
Simulating Vigenere Cipher
---------- ----------
Input Message : GaneshaMoorthi
Encrypted Message : BITIFLRQQWGALZ
Decrypted Message : GANESHAMOORTHI
```

Aim & Algorithm	
Program & Execution	
Experiment & Results	
Viva	
Total	

RESULT:

Thus the program for vigenere cipher encryption and decryption algorithm hasbeen implemented and the output verified successfully.

Ex. No : 2(a) Date:	**Rail Fence Cipher Transposition Technique**

AIM:

To implement a program for encryption and decryption using rail fence transposition technique.

ALGORITHM:

1. Get the plain text from the user.
2. In the rail fence cipher, the plaintext is written downwards and diagonally on successive "rails" of an imaginary fence, then moving up when we reach the bottom rail.
3. When we reach the top rail, the message is written downwards again until thewhole plaintext is written out.
4. The message is then read off in rows.This is Encryption.

PROGRAM:

railFenceCipher.java

```
class railfenceCipherHelper {
        int depth;
        String encode(String msg, int depth) throws Exception
      {
       int r =depth;
       int l =
       msg.length();int c
       = l / depth;
       int k = 0;
       char mat[][] = new char[r][c];
       String enc = "";
       for (int i = 0; i < c; i++) {
       for(int j = 0; j < r; j++) {
             if (k != l) {
                mat[j][i] = msg.charAt(k++);
             } else {
                mat[j][i] = 'X';
             }
          }
       }
```

```
        for (int i = 0; i < r; i++) {
        for(int j = 0; j < c; j++) {
              enc += mat[i][j];
            }
        }
        return enc;
    }
    String decode(String encmsg, int depth) throws Exception {
         int r= depth;
        int l = encmsg.length();
        int c = l / depth;
        int k = 0;
        char mat[][] = new char[r][c];
        String dec = "";
        for (int i = 0; i < r; i++) {
        for(int j = 0; j < c; j++) {
              mat[i][j] = encmsg.charAt(k++);
            }
        }
        for (int i = 0; i < c; i++)
            {
         for(int j = 0; j < r; j++) {
              dec += mat[j][i];
            }
        }
        return dec;
    }
}

class railFenceCipher {
    public static void main(String[] args) throws java.lang.Exception {
     railfenceCipherHelperrf = new railfenceCipherHelper();
        String msg, enc, dec;
        msg = " GANESHA
        MOORTHI ";
        int depth = 2;
        enc = rf.encode(msg,
        depth);dec = rf.decode(enc,
        depth);
```

```
        System.out.println("Simulating Railfence Cipher\n ------------------------");
        System.out.println("Input Message : " + msg);
        System.out.println("Encrypted Message : " +
        enc);System.out.printf("Decrypted Message : "
        + dec);
    }
}
```

OUTPUT:

```
SimulatingRailfenceCipher
---------------------------------
Input Message : Ganesha Moorthi, Chennai
Encrypted Message : GnsaMoti hnaaeh orh,Ceni
Decrypted Message : Ganesha Moorthi, Chennai
```

Aim & Algorithm	
Program & Execution	
Experiment & Results	
Viva	
Total	

RESULT:

Thus the java program for Rail Fence Transposition Technique has beenimplemented andthe output verified successful.

Ex. No : 2(b) Date:	Row and Column Transformation Technique

AIM:

To implement a program for encryption and decryption by using row and column transformation technique.

ALGORITHM:

1. Get the input from the user.
2. User the row column transformation technique to get the cipher text
3. Reverse the process to get the decrypt the cipher text to plain text.

PROGRAM:

TransCipher.java

```
import java.util.*;
class TransCipher
{
   public static void main(String args[]) {
    Scanner sc = new Scanner(System.in);
      System.out.println("Enter the plain
      text"); String pl = sc.nextLine();
      sc.close();
      String s = "";
      intstart = 0;
      for (int i = 0; i < pl.length();
         i++) { if(pl.charAt(i) == ' ') {
            s = s + pl.substring(start, i);
            start= i + 1;
         }
      }
      s = s + pl.substring(start);
```

```
    System.out.print(s);
    System.out.println();

    // end of space deletion

    int k = s.length();
    int  l= 0;
    int col = 4;
    int row = s.length() / col;
    char ch[][] = new
    char[row][col];
    for(int i = 0; i < row; i++) {
        for (int j = 0; j < col; j++) { if (l
            < k) {
                ch[i][j] = s.charAt(l); l++;
            } else {
                ch[i][j] = '#';
            }
        }
    }

    // arranged in matrix

    char trans[][] = new
    char[col][row];
     for(int i = 0; i < row; i++) {
        for (int j = 0; j < col; j++) {
        trans[j][i]= ch[i][j];
        }
    }

    for (int i = 0; i < col; i++) {
    for (intj = 0; j < row; j++) {
            System.out.print(trans[i][j]);
        }
    }
    // display
    System.out.println();
}}
```

OUTPUT:

Enter the plain text
GaneshaMoorthi
GaneshaMoorthi
GaneshaMoorthi
GsoahonareMt

Aim & Algorithm	
Program & Execution	
Experiment & Results	
Viva	
Total	

RESULT:

Thus the java program for Row and Column Transposition Technique has been implemented and the output verified successfully.

Ex. No : 3 Date:	Data Encryption Standard (DES) Algorithm (User Message Encryption)

AIM:

To use Data Encryption Standard (DES) Algorithm for a practical application like User Message Encryption.

ALGORITHM:

1. Create a DES Key.
2. Create a Cipher instance from Cipher class, specify the followinginformation and separated by a slash (/).
   a. Algorithm name
   b. Mode (optional)
   c. Padding scheme (optional)
3. Convert String into ***Byte[]*** array format.
4. Make Cipher in encrypt mode, and encrypt it with ***Cipher.doFinal()*** method.
5. Make Cipher in decrypt mode, and decrypt it with ***Cipher.doFinal()*** method.

PROGRAM:

DES.java

```
import java.security.InvalidKeyException;
import
java.security.NoSuchAlgorithmException;
import
javax.crypto.BadPaddingException;
import javax.crypto.Cipher;
import javax.crypto.IllegalBlockSizeException;
importjavax.crypto.KeyGenerator;
import javax.crypto.NoSuchPaddingException; import
javax.crypto.SecretKey;

public class DES
{
        public static void main(String[] argv) {

                try{
```

```
System.out.println("Message Encryption Using DES Algorithm\n ------ ");
KeyGenerator keygenerator =KeyGenerator.getInstance("DES");
        desCipher = Cipher.getInstance("DES/ECB/PKCS5Padding");
        desCipher.init(Cipher.ENCRYPT_MODE, myDesKey);
        byte[] text= " GANESHA MOORTHI ".getBytes();
        System.out.println("Message[Byte Format] : " + text);
        System.out.println("Message : " + new String(text));
        byte[] textEncrypted = desCipher.doFinal(text);
        System.out.println("Encrypted Message: " +
        textEncrypted);
        desCipher.init(Cipher.DECRYPT_MODE, myDesKey);
        byte[]textDecrypted = desCipher.doFinal(textEncrypted);
        System.out.println("Decrypted Message: " + new
        String(textDecrypted));

    }catch(NoSuchAlgorithmException e){
          e.printStackTrace();
    }catch(NoSuchPaddingException e){
          e.printStackTrace();
    }catch(InvalidKeyException e){
          e.printStackTrace();
    }catch(IllegalBlockSizeException e){
          e.printStackTrace();
    }catch(BadPaddingException e){
          e.printStackTrace();
      }}}
```

OUTPUT:

```
Message Encryption UsingDESAlgorithm
-------------
Message [Byte Format] : [B@204f30ec
Message : Ganesha Moorthi
Encrypted Message: [B@146ba0ac
Decrypted Message: Ganesha Moorthi
```

Aim & Algorithm	
Program & Execution	
Experiment & Results	
Viva	
Total	

RESULT:

Thus the java program for DES Algorithm has been implemented and the outputverified successfully.

Ex. No : 4 Date :	Advanced Encryption Standard (AES) Algorithm (URL Encryption)

AIM:

To use Advanced Encryption Standard (AES) Algorithm for a practical application like URL Encryption.

ALGORITHM:

1. AES is based on a design principle known as a substitution–permutation.
2. AES does not use a Feistel network like DES, it uses variant of Rijndael.
3. It has a fixed block size of 128 bits, and a key size of 128, 192, or 256 bits.
4. AES operates on a 4 × 4 column-major order array of bytes, termed the state

PROGRAM:

AES.java

```
import java.io.UnsupportedEncodingException;
importjava.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
importjava.util.Arrays;
import java.util.Base64;
import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;

publicclass AES {

   private static SecretKeySpec
   secretKey;private static byte[] key;
   public static void setKey(String
      myKey) {
    MessageDigest sha = null;
    try {
        key = myKey.getBytes("UTF-8");
        sha = MessageDigest.getInstance("SHA-
        1");
        key = sha.digest(key);
        key = Arrays.copyOf(key, 16);
        secretKey = new SecretKeySpec(key, "AES");
```

```
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        } catch (UnsupportedEncodingException
            e) {e.printStackTrace();
        }
    }

    public static String encrypt(String strToEncrypt, String secret) {
     try {setKey(secret);
            Cipher cipher = Cipher.getInstance("AES/ECB/PKCS5Padding");
            cipher.init(Cipher.ENCRYPT_MODE,
            secretKey);return
            Base64.getEncoder().encodeToString(cipher.d
            oFinal(strToEncrypt.getBytes("UTF
-8")));
        } catch (Exception e) {
            System.out.println("Error while encrypting: " + e.toString());
        }
        return null;
    }
    public static String decrypt(String strToDecrypt, String secret) {
    try {setKey(secret);
            Cipher cipher = Cipher.getInstance("AES/ECB/PKCS5PADDING");
            cipher.init(Cipher.DECRYPT_MODE,
            secretKey);return new
String(cipher.doFinal(Base64.getDecoder().decode(strToDecrypt)));
        } catch (Exception e) {
            System.out.println("Error while decrypting: " + e.toString());
        }
        return null;
    }
    public static void main(String[] args) {
        final String secretKey = "
        GANESHA MOORTHI ";
        String originalString = "www. Ganesha Moorthi.edu";
        String encryptedString = AES.encrypt(originalString, secretKey);
         StringdecryptedString = AES.decrypt(encryptedString, secretKey);
```

```
        System.out.println("URL Encryption Using AES Algorithm\n---------- ");
        System.out.println("Original URL : " +originalString);
        System.out.println("Encrypted URL : " +
        encryptedString);
        System.out.println("Decrypted URL : " + decrypted
        String);
    }
}
```

OUTPUT:

```
URL Encryption UsingAESAlgorithm
---------------------
Encrypted URL : oNDsybbt+gie3H/c1TDHqvGqJifuthLGtunvkhikB5I=
Original URL : www.GaneshaMoorthi.edu
Decrypted URL : www.GaneshaMoorthi.edu
```

Aim & Algorithm	
Program & Execution	
Experiment & Results	
Viva	
Total	

RESULT:

Thus the java program for AES Algorithm has been implemented for URL Encryption andthe output verified successfully.

Ex. No : 5 Date :	RSA Algorithm

AIM:

To implement RSA (Rivest–Shamir–Adleman) algorithm by using HTML and Javascript.

ALGORITHM:

1. Choose two prime number p and q
2. Compute the value of n and **p**
3. Find the value of ***e*** (public key)
4. Compute the value of ***d*** (private key) using gcd()
5. Do the encryption and decryption
   a. Encryption is given as,
   $$c = t^e \bmod n$$
   b.Decryption is given as,
   $$t = c^d \bmod n$$

PROGRAM:

```
import
java.io.DataInputStream;
import java.io.IOException;
import java.math.BigInteger;
import java.util.Random;

public class Main
 {
    private BigInteger P;
    private BigInteger Q;
    private BigInteger N;
    private BigInteger
    PHI;private
    BigInteger e; private
    BigInteger d;
    private int maxLength =
```

```
1024;

private Random R;

 public Main()
{
    R = new Random();
    P = BigInteger.probablePrime(maxLength, R);
    Q = BigInteger.probablePrime(maxLength, R);
    N = P.multiply(Q);
    PHI = P.subtract(BigInteger.ONE).multiply(
    Q.subtract(BigInteger.ONE));
     e = BigInteger.probablePrime(maxLength / 2, R);
    while (PHI.gcd(e).compareTo(BigInteger.ONE) > 0 && e.compareTo(PHI) < 0)
    {
       e.add(BigInteger.ONE);
    }
    d = e.modInverse(PHI);
}
public Main(BigInteger e, BigInteger d, BigInteger N)
{
    this.e = e;
    this.d = d;
    this.N    =
    N;
}
public static void main (String [] arguments) throws IOException
{
    Main rsa = new Main();
    DataInputStream input = new
    DataInputStream(System.in);String inputString;
    System.out.println("Enter message you wish to send.");
    inputString =  input.readLine();
    System.out.println("Encrypting the message: " + inputString);
    System.out.println("The message in bytes is:: " + bToS(inputString.getBytes()));

    // encryption
    byte[] cipher = rsa.encryptMessage(inputString.getBytes());
    // decryption
```

```
        byte[] plain = rsa.decryptMessage(cipher);
        System.out.println("Decrypting Bytes: " + bToS(plain));
        System.out.println("Plain message is: " + new
        String(plain));
    }
    private static String bToS(byte[] cipher)
    {
        String temp = "";
        for (byte b :
        cipher)
        {
            temp += Byte.toString(b);
        }
        return temp;
}
    // Encrypting the message
    public byte[] encryptMessage(byte[] message)
    {
        return (new BigInteger(message)).modPow(e, N).toByteArray();
    }
    // Decrypting the message
    public byte[] decryptMessage(byte[] message)
    {
        return (new BigInteger(message)).modPow(d, N).toByteArray();
    }

}
```

OUTPUT:

```
Enter message you wish to send.
HI I AM GANESH
Encrypting the message: HI I AM GANESH
The message in bytes is:: 72733273326577327165786983 72
Decrypting Bytes: 72733273326577327165786983 72
Plain message is: HI I AM GANESH
```

Aim & Algorithm	
Program & Execution	
Experiment & Results	
Viva	
Total	

RESULT:

Thus the RSA algorithm has been implemented using HTML & CSS and theoutputhas been verified successful.

Ex. No : 6 Date :	Digital Signature Standard

AIM:

To implement the SIGNATURE SCHEME - Digital Signature Standard.

ALGORITHM:

a. Create a KeyPairGenerator object.
b. Initialize the KeyPairGenerator object.
c. Generate the KeyPairGenerator. ...
d. Get the private key from the pair.
e. Create a signature object.
f. Initialize the Signature object.
g. Add data to the Signature object
h. Calculate the Signature

PROGRAM:

```
import java.security.KeyPair;
import
java.security.KeyPairGenerator;
import      java.security.PrivateKey;
import java.security.Signature;
import java.util.Scanner;

public class CreatingDigitalSignature {
   public static void main(String args[]) throws Exception {
      Scanner sc = new Scanner(System.in);
      System.out.println("Enter some text");
      String msg = sc.nextLine();
      KeyPairGenerator keyPairGen = KeyPairGenerator.getInstance("DSA");
      keyPairGen.initialize(2048);
      KeyPair pair =
      keyPairGen.generateKeyPair();PrivateKey
      privKey = pair.getPrivate();
      Signature sign = Signature.getInstance("SHA256withDSA");
      sign.initSign(privKey);
      byte[] bytes =
      "msg".getBytes();
```

```
        sign.update(bytes);
        byte[] signature = sign.sign();
        System.out.println("Digital signature for given text: "+new String(signature,"UTF8"));
    }
}
```

OUTPUT:

```
Enter some text
Ganesha Moorthi
Digital signature for given
text:0<hK:}ZZu$QI
JayR-:9
```

Aim & Algorithm	
Program & Execution	
Experiment & Results	
Viva	
Total	

RESULT:

Thus the Digital Signature Standard Signature Scheme has been implemented and the output has been verified successfully.

Ex. No : 7 Date :	Demonstration of Intrusion Detection System(IDS)

AIM:

To demonstrate Intrusion Detection System (IDS) using Snort software tool.

STEPS ON CONFIGURING AND INTRUSION DETECTION:

1. Download Snort from the Snort.org website.(http://www.snort.org/snort-downloads)
2. Download Rules(https://www.snort.org/snort-rules). You must register toget therules. (You should download these often)
3. Double click on the .exe to install snort. This will install snort in the "C:\Snort"folder.It is important to have WinPcap (https://www.winpcap.org/install/) installed
4. Extract the Rules file. You will need WinRAR for the .gz file.
5. Copy all files from the "rules" folder of the extracted folder. Nowpaste therules into *"C:\Snort\rules"* folder.
6. Copy "snort.conf" file from the "etc" folder of the extracted folder. You mustpaste it into "C:\Snort\etc" folder. Overwrite any existing file. Remember if you modify your snort.conf file and download a new file, you must modify it forSnort to work.
7. Open a command prompt (cmd.exe) and navigate to folder "C:\Snort\bin"folder. (at the Prompt, type cd\snort\bin)
8. To start (execute) snort in sniffer mode use following command:snort -dev -i 3
    a. -i indicates the interface number. You must pick the correct interface number. Inmycase, it is 3.
    b. -dev is used to run snort to capture packets on your network.
    c. To check the interface list, use following command:snort -W

   a. Finding an interface
   b. You can tell which interface to use by looking at the Index number and findingMicrosoft.As you can see in the above example, the other interfaces are for VMWare. My interface is 3.

2. To run snort in IDS mode, you will need to configure the file "snort.conf"according to your network environment.
3. To specify the network address that you want to protect in snort.conffile, lookfor the following line.
   a. var HOME_NET 192.168.1.0/24 (You will normally see any here)
4. You may also want to set the addresses of DNS_SERVERS, if youhave someon your network.
   a. Example:
   b. example snort
5. Change the RULE_PATH variable to the path of rules folder.var RULE_PATH c:\snort\rules
   a. path to rules
6. Change the path of all library files with the name and path on yoursystem. andyou must change the pathof snort_dynamicpreprocessorvariable.

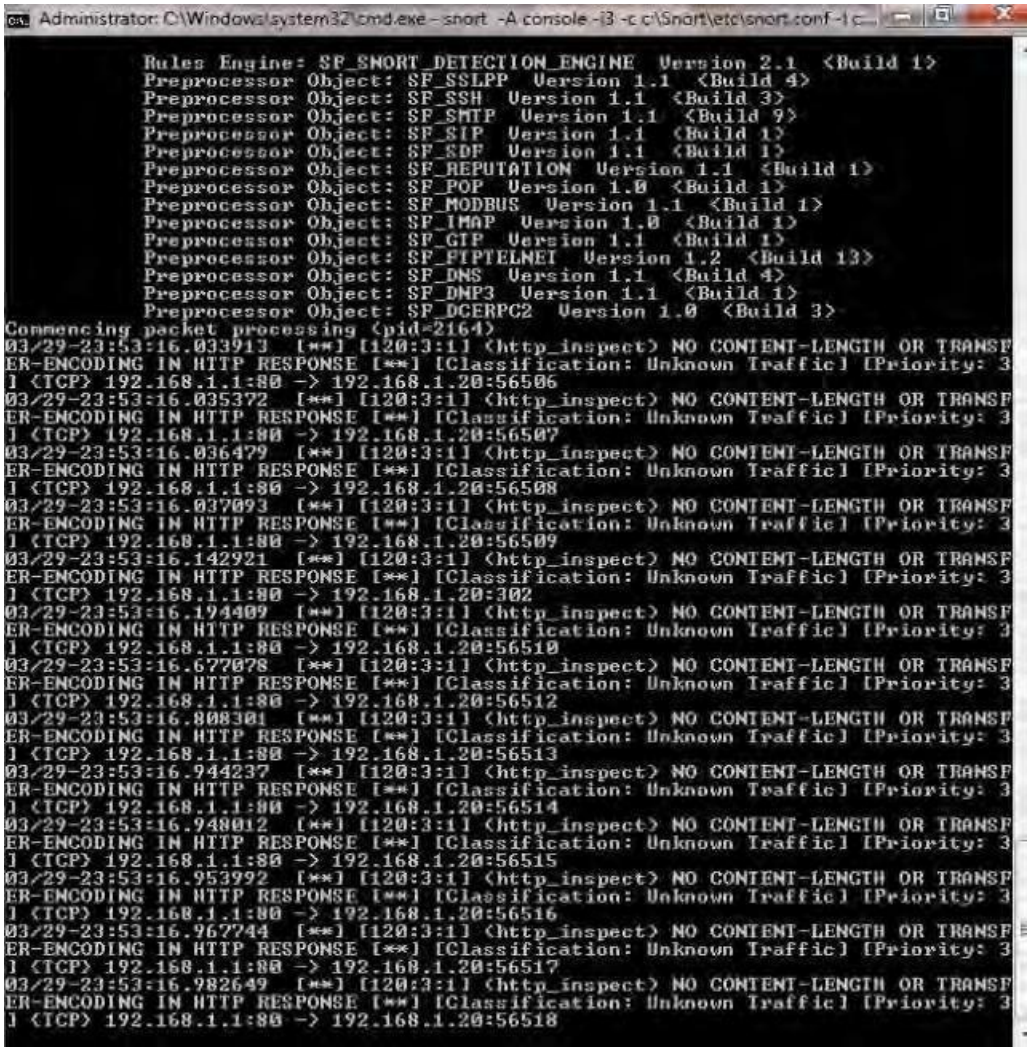
C:\Snort\lib\snort_dynamiccpreprocessor

a. You need to do this to all library files in the "C:\Snort\lib" folder. The old path might be:"/usr/local/lib/…". you will need to replace that path with your system path. Using C:\Snort\lib

7. Change the path of the "dynamicengine" variable value in the "snort.conf"file..
   a. Example:
   b. dynamicengine C:\Snort\lib\snort_dynamicengine\sf_engine.dll
   c. 15 Add the paths for "include classification.config" and "include reference.config"files.include c:\snort\etc\classification.configinclude
   d. c:\snort\etc\reference.config
8. Remove the comment (#) on the line to allow ICMP rules, if it is commentedwith a #.
   a. include $RULE_PATH/icmp.rules
9. You can also remove the comment of ICMP-info rules comment,if it iscommented.
   a. include $RULE_PATH/icmp-info.rules
10. To add log files to store alerts generated by snort, search for the"output log"test in snort.conf and add the following line:
    a. output alert_fast: snort-alerts.ids
11. Comment (add a #) the whitelist
    a. $WHITE_LIST_PATH/white_list.rules andthe blacklist
    b. Change the nested_ip inner , \ to nested_ip inner #, \
12. Comment out (#) following lines:#preprocessor normalize_ip4
    a. #preprocessor normalize_tcp: ips ecn stream#preprocessor normalize_icmp4 #preprocessor normalize_ip6
    b. #preprocessor normalize_icmp6
13. Save the "snort.conf" file.
14. To start snort in IDS mode, run the following command:
    a. snort -c c:\snort\etc\snort.conf -l c:\snort\log -i 3(Note: 3is used for my interface card)

b. If a log is created, select the appropriate program to open it. You can useWordPard or NotePad++ to read the file.
c. To generate Log files in ASCII mode, you can use following command whilerunningsnort in IDS mode:
d. snort -A console -i3 -c c:\Snort\etc\snort.conf -l c:\Snort\log -K ascii

15. Scan the computer that is running snort from another computer byusing PINGor NMap (ZenMap).
    a. After scanning or during the scan you can check the snort-alerts.ids file in the logfolder to insure it is logging properly. You will see IP address folders appear
    b. Snort monitoring traffic –



Aim & Algorithm	
Program & Execution	
Experiment & Results	
Viva	
Total	

RESULT:

Thus the Intrusion Detection System(IDS) has been demonstrated by usingtheOpen Source Snort Intrusion Detection Tool.

Ex.no:8 Date:	**Calculate the Message Digest of a text using SHA-1 algorithm**

AIM:

To calculate the Message Digest of a text using SHA-1 algorithm

ALGORITHMS:

SHA-1 works by feeding a message as a bit string of length less than 2^{64}264 bits, and producing a 160-bit hash value known as a *message digest*.

Note that the message below is represented in hexadecimal notation for compactness.

PROGRAM:

```
// Java program to calculate SHA-1 hash value

import java.math.BigInteger;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

public class GFG {
	public static String encryptThisString(String input)
	{
		try {
			// getInstance() method is called with algorithm SHA-1
			MessageDigest md = MessageDigest.getInstance("SHA-1");

			// digest() method is called
			// to calculate message digest of the input string
			// returned as array of byte
			byte[] messageDigest = md.digest(input.getBytes());

			// Convert byte array into signum representation
			BigInteger no = new BigInteger(1, messageDigest);

			// Convert message digest into hex value
			String hashtext = no.toString(16);
```

```
                // Add preceding 0s to make it 32 bit
                while (hashtext.length() < 32) {
                        hashtext = "0" + hashtext;
                }

                // return the HashText
                return hashtext;
        }

        // For specifying wrong message digest algorithmscatch
(NoSuchAlgorithmException e) {
                throw new RuntimeException(e);
        }}
    // Driver code
    public static void main(String args[]) throws NoSuchAlgorithmException
    {
        System.out.println("HashCode Generated by SHA-1 for: ");
        String s1 = "GeeksForGeeks";
        System.out.println("\n" + s1 + " : " + encryptThisString(s1));
        String s2 = "hello world";
        System.out.println("\n" + s2 + " : " + encryptThisString(s2));
    }}
```

OUTPUT:

HashCode Generated by SHA-1 for:

Ganesha : df73451d0ba10641b8646bd3726597b90c84d1a8

Moorthi : c25c211382339e3ca8b91bbaccb7cd4da89f4066

Aim & Algorithm	
Program & Execution	
Experiment & Results	
Viva	
Total	

RESULT:

Thus, to Calculate the Message Digest of a text using SHA-1 algorithm has been executed Succesfully

Ex.no:9 Date:	**Case Study - DES**

AIM:

Real time case study on Data Encryption Standard (DES) Algorithm.

CASE STUDY:

The Business Challenge

- A proposal from IBM, a modifi cation of a project called Lucifer, was accepted as DES
- . DES was published in the Federal Register in March 1975 as a draft of the Federal Information Processing Standard (FIPS).
- After the publication, the draft was criticized severely for two reasons.
- First, critics questioned the small key length (only 56 bits), which could make the cipher vulnerable to brute-force attack. Second, critics were concerned about some hidden design behind the internal structure of DES.

SOLUTION

- Bit 15 in the input becomes bit 63 in the output. Bit 64 in the input becomes bit 25 in the output. So the output has only two 1s, bit 25 and bit 63. The result in hexadecimal is 0x0000 0080 0000 0002 For synthesis.
- Only bit 25 and bit 64 are 1s; the other bits are 0s. In the fi nal permutation, bit 25 becomes bit 64 and bit 63 becomes bit 15. The result 0x0002 0000 0000 0001 ‘
- which go to the next round (or fi nal permutation box). As we discussed in Chapter 5, we can assume that each round has two cipher elements (mixer and swapper). Each of these elements is invertible. The swapper is obviously invertible

RESULTS

- Also, we have employed pipelining S-box implementation using composite field that decreases the path delay. Therefore, the throughput is increased.
- Furthermore, efficient key expansion architecture suitable with the full-pipelined round units is introduced.
- Our implementation takes 100 clock cycles to load the first cipher block. Then, it will appear on consecutive clock cycles.
- The implementation is done by virtex-5 and virtex-6 FPGAs.
- It can encrypt data blocks at a rate of 81.68 Gbps and 108.69 Gbps respectively.
- Our proposed design improves in throughput all the reported architectures of AES algorithm, with consuming low area.

Aim & Algorithm	
Program & Execution	
Experiment & Results	
Viva	
Total	