Aim: Write a C program that contains a string (char pointer) with a value \Hello world". The program should XOR each character in this string with 0 and display the result

Program:

```
#include<stdlib.h>
int main()
{
  Char str[] = "Hello world";
  Char str1[11];
  int i, len;
   len = strlen(str);
  for(i=0; i<len; i++)
  {
     str1[i] = str[i] ^1;
     printf("%c", str1[i]);
  }
  printf("\n");
  return 0;
}
```

output:

Hello world

Aim: Write a C 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 display the result.

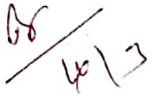
program:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
void main()
{
    char str[] = "Hello world";
    char str1[11];
    char str2[11];
    char str3[11];
    int i,len;
    len = strlen(str);
    strcpy(str2,str);
    for(i=0;i<len;i++)
    { str1[i] = str[i] & 127;
      printf("%c",str1[i]);
    }printf("\n");
    for(i=0;i<len;i++)
    { str1[i] = str2[i]^127;
      printf("%c",str3[i]);
    }
    printf("\n");

}
```

output:

Hello world
!!<-!!!!->(



Aim: write a Java program to perform encryption and decryption using the following

Algorithms:

a) Ceasar Cipher:

Program:

```
import java.io.BufferReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.Scanner;
Public class CeaserCipher
{
  Static Scanner sc=new Scanner(System.in);
  Static BufferReader br=new BufferReader
   (new InputStreamReader (System.in));
   Public static void main(String[] args)
   throws IoException
   {
     //To pocode application logichere.
       System.out.println("Enter any string:");
       String str = br.readline();
       System.out.print("\nEnter the key:");
       int key = sc.next Int();
       String encrypted = encrypt (str, key);
       System.out.println ("\nEncrypted strings:" +
        encrypted);
       String decrypted = decrypt(encrypted, key);
       System.out.println("\n Decrypted strings:
       +decrypted);
       System.out.println("\n");
   }
```

```
Public Static String encrypt (String str, int key)
{
    String encrypted = " ";
    for(int i=0; i<str.length(); i++)
    {
        int c=str.charAt(i);
        if(Character.isUpperCase(c))
        {
            c=c+(key %26);
            if (c>'z')
            c=c-26;
        }
        else if(Character.isLowerCase(c))
        {
            c=c+(key%26);
            if(c>'z')
                c=c-26;
        }
        encrypted+=(char)c;
    }
    return encrypted;
}
public Static String decrypt(String str, int key)
{ String decrypted = " ";
  for(int i=0; i<str.length(); i++){
    int c= str.charAt(i);
    if(Character.isUpperCase(c))
    {
        c=c-(key%26);
        if (c<'A')
            c=c+26;
    }
    else if(Character.isLowerCase(c))
    {
```

Output:

```
Enter any string: Hello World
Enter the key: 2
Encrypted string is: Jgnnq Yqtnf
Decrypted string is: Hello World.
```

```
        c = c - (key % 26);
        if (c < 'a')
            c = c + 26;
    }
    decrypted += (char)c;
}
return decrypted;
}
}
```

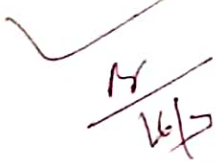
b. Substitution Cipher:

Program:

```
import java.io.*;
import java.util.*;
Public class SubstitutionCipher
{ Static Scanner sc=new Scanner(System.in);
  Static BufferedReader br=new InputStrea-
  -m Reader(System.in));
  Public static void main(String[] args)
  throws IoException
  { //TO Do Code application logic Here
  String a="abcdefghijklmnopqrstuvwxyz";
  String b="zyxwvuutsrqponml[kjihgfedcb
  System.out.println("Enter any string:");
  String str=br.readline();
  String decrypt="";
  char c;
  for(int i=0;i<str.length();i++)
  {
      c=str.charAt[i];
      int j=a.indexof(c);
      decrypt=decrypt+b.charAt(i);
  }
  System.out.println("The encrypted data is:
  +decrypt);
  }
}
```

Output:

Enter any string: Substitution
The encrypted data is: hfyhtgxgfgxlm



5. Hill Cipher:

Program:

```
import java.io.*;
import java.util.*;
import java.io.*;
public class HillCipher
{
  static float[][]decrypt=new float[3][1];
  static float[][] a = new float[3][3];
  static float[][] b = new float[3][3];
  static float[][] mes = new float[3][1];
  static float[][] res = new float[3][1];
  static BufferReader br = new BufferedReader(new input stream Reader(System.in));
  public static void main(String [] args )
  throws IoException
  {
   // ToDo code application logic here
    getkeymes();
    for(int i=0; i<3; i++)
    for (int j=0; j<1;j++)
    for(int k=0; k<3; k++)
    {
      res[i][j]=res[i][j]+a[i][k]*mes[k][j];
    }
    System.Out.print("\nEncrypted string is: ");
    for(int i=0; i<3;i++)
    {
      System.out.print((char)(res[i][0]%.26 +97));
      res[i][0]= res[i][0];
    }
    invcose();
    for(int i=0; i<3 ; i++)
    for(int j=0; j<1; j++)
    for(int k=0; k<3; k++)
```

```
{
decrypt[i][j] = decrypt[i][j] * b[i][k] * res[k
[j];
}
System.out.print("\n Decrypted string is:");
for(int i=0; i<3; i++)
{
System.out.print(char)(decrypt[i][0] % 26+97)
}
System.out.print("\n");
}
public static void getkeymes() throws IOExcept
-ion
{ System.out.println("Enter 3x3 matrix for key(It sh
-ould be inversible):");
for(int i=0; i<3; i++)
for(int j=0; j<3; j++)
a[i][j] = sc.nextFloat();
System.out.println("\n Enter a 3 letter string:");
String msg = br.readLine();
for(int i=0; i<3; i++)
mes[i][0] = msg.charAt(i) - 97;
}
public static void inverse()
{
float p, q;
float[] c = a;
for(int i=0; i<3; i++)
for(int j=0; j<3; j++)
{
//a[i][j] = sc.nextFloat();
if(i==j)
b[i][j] = 1
else
b[i][j] = 0;
}
for(int k=0; k<3; k++)
```

```
{
  for(int i=0;i<3;i++)
  {
     p=c[i][k];
     q=c[k][k];
     for(int j=0;j<3;j++)
     {
        if(i!=k)
        {
           c[i][j]=c[i][j]*q-p*c[k][j];
           b[i][j]=b[i][j]*q-p*b[k][j];
        }
     }
  }
for(int i=0;i<3;i++)
for(int j=0;j<3;j++)
{   b[i][j]=b[i][j]/c[i][j];
}
system.out.println(" ");
system.out.println("\n Inverse matrixes:");
for(int i=0;i<3;i++)
{ for(int j=0;j<3;j++)
   { system.out.print(b[i][j]+" ");
     system.out.print("\n");
   }
}
}
```

Output:

```
Enter 3x3 matrix:
6   24   1
13  16   10
20  11   15
Enter a 3 letter string: hel
Encrypted string is: Hj
Inverse matrix is: 0.158  -0.727  0.5079
                   0.0113  0.158  -0.1065
                   -0.224  0.857  -0.4887
Decrypted string is: hel
```

16/3

Aim: write a Java program to implement DES algorithm

Program:

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.security.spec.KeySpec;
import javax.crypto.Cipher;
import javax.crypto.SecretKey;
import javax.crypto.SecretKeyFactory;
import javax.crypto.spec.DESKeySpec;
import java.util.Base64;

Public class DES
{
  Private static final String UNICODE_FORMAT
        ="UTF-8";
  Public static final String DES_ENCRYPTION_
  SCHEME  ="DES";
  Private KeySpec myKeySpec;
  Private SecretKeyFactory mySecretKeyFactory;
  Private Cipher cipher;
  byte[] keyAsBytes;
  Private String myEncryptionKey;
  Private String myEncryptionScheme;
  Secret Key  key;
  Static BufferedReader br= new BufferedReader
  (new InputStreamReader(System.in))
  Public DES() throws Exception
  {
    myEncryptionKey = "this is Secret Encryption
                       key";
    myEncryptionScheme = DES_ENCRYPTION_
                         SCHEME;
```

```
keyAsBytes = myEncryptionKey.getBytes
                (UNICODE_FORMAT);
my keySpec = new DESKeySpace(keyAsBytes);
my SecretkeyFactory = Secretkey Factory.
    getInstance(myEncryption Scheme);
}
public String encrypt(String unencryptedString)
{
  String encryptedString = null;
  try
  {
    Cipher.init(Cipher.ENCRYPT_MODE, key);
    Byte[] plainText = unencrypted String.
                getBytes(UNICODE-FORMAT);
    Byte[] encryptedText = Cipher.doFinal(plain
                Text);
    Base64.Encoder Base64encoder = Base64.
                getEncoder();
    Byte[] encryptedString1 = Base64 encoder.
                encode(encrypted Text);
    encrypted String = Bytes2String(encry.
                -pted string1);
  }
  catch(Exception e)
  {
    e.printStackTrace();
  }
  return encryptedString;
}
Public String decrypt(String encrypted String)
{
  String decryptedText = null;
  try
  {
    Cipher.init(Cipher.DECRYPT_MODE, key);
    Base64.Decoder Base64decoder = Base64
                getDecoder();
```

```
Byte[] encrypted Text = base64decoder.decode(e-
-ncrypted String);
Byte[] plainText = Cipher.doFinal(encrypted Text);
decrypted Text = Byte2String(plain Text);
}
catch (Exception e)
{
    e.printStackTrace();
}
return decrypted Text;
}
private static String Byte2string(Byte[] Bytes)
{
    StringBuilder stringBuffer = new StringBuilder();
    for(int i=0; i< Bytes.length; i++)
    {
        stringBuffer.append((char) Bytes[i]);
    }
    return stringBuffer.toString();
}
public static void main(String args[]) throws
                Exception
{
    System.out.print("Enter the string:");
    DES myEncryptor = new DES()
    String stringToEncrypt = br.readLine();
    String encrypted = myEncryptor.encrypt
                (String To Encrypt);
    String decrypted = myEncryptor.decrypt(encr
                ypted);
    system.out.println("\nString to Encrypt
    System.out.println("Encrypted value
                +encrypted);
    system.out.println("Decrypted value:
                +decrypted);
}
}
```

output:

Enter the string: Gayathri
String to Encrypt: GayathRi
Encrypted value: I+Qr=CPd2
Decrypted value: Gayathri

16/1/23

Experiment-5

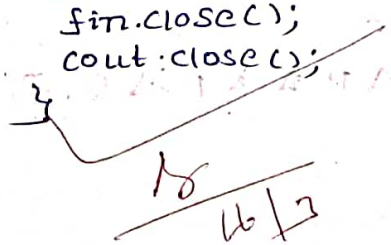
Aim: Write a program to Implement BlowFish Algorithm.

program:

```
import java.io.*;
import java.io.FileInputStream;
import java.io.FileOutputstream;
import java.security.Key;
import java.crypto.Cipher;
import java.crypto.CipherOutputstream;
import java.Caypto.KeyGenerator;
import java.util.*;
public class BlowFish
{
  public static void main (String[] args) throws
    Exception
    {
      Key Generator KeyGenerator = KeyGenerator.
                    getInstance("BlowFish");
      KeyGenerator.init(128);
      Key secretkey = KeyGenerator.generatekey();
      Cipher CipherOut = Cipher.getInstance("BF
      -WFish/CFB/NoPadding");
      CipherOut.init(Cipher.ENCRYPT-MODE, secretkey
      Base64.Encoder encoder = Base64.getEncoder();
      byte iv[] = CipherOut.getIV();
      if (iv!=null)
      { System.out.println("Initialization vect-
        -or of cipher:"+encoder.encode(iv));
      }
      FileInputstream fin = new FileInputstream
       ("Inputfile.txt");
      FileOutputstream fout = new file output Stream
      ("output file.txt");
      CipherOutstream cout = new CipherOutput str-
      -eam(fout, cipher out);
      int Input = 0;
```

```
while(input=fin.read())!=-1)
{
        cout.write(input);

}
fin.close();
cout:close();
}
```



Output:

Initialization vector of the Cipher:

[B@224edc67]

input.txt: Hello, Everyone

output-file.txt: He ĿH - Δ P Δ Δ P Δ ⊡ Δ

16/7

Experiment-6

Aim: write a program to Implement Rijndael algorithm

Program:

```
import java.security.*;
import javax.crypto.*;
import javax.crypto.spec.*;
import java.io.*;
public class Rijndael
{ public static String asHex(byte buf[])
  { StringBuffer strbuf = new StringBuffer
    (buf.length *2);
    int i;
    for(i=0; i<buf.length; i++)
    { if((int)buf[i] & 0xff) < 0x10)
          strbuf.append("0");
      strbuf.append(Long.toString((int)buf[i] &
      0xff, 16));
    }
    return strbuf.toString();
  }
  public static void main(String[] args) throws
  Exception
  { String messages = "AES still rocks!!";
    KeyGenerator kgen = KeyGenerator.getInstance
    ("AES");
    kgen.init(128);
    SecretKey skey = kgen.generateKey()
    byte[] raw = skey.getEncoded();
    SecretKeySpec skeySpec = new SecretKeySpec(r
    -aw, "AES");
    Cipher cipher = Cipher.getInstance("AES");
    cipher.init(Cipher.ENCRYPT_MODE, skeySpec);
```

```
byte[] encrypted = cipher.doFinal (args.leng-
-th == 0 ? message:args[0]).getBytes());
System.out.println ("encrypted string:"
+asHex(encrypted));
Cipher.init(Cipher.DECRYPT_MODE,skeySpec);
byte[] original = cipher.dofinal (encrypted);
String originalString = new String(original);
System.out.println ("original string:"+
originalString + " " +asHex(original));
}
}
```

output:

encrypted string: b2c9c04988 5701dd1026 3c
7963d1a662f9bb4b2b0cb59 9c5db7a

original string: AES still rocks!!
41455320737469
20726fc3b73212.

16/12/2

```
public static BigInteger generateE(BigInteger
fiofn)
{
    int y, intGCD;
    BigInteger e;
    BigInteger gcd;
    Random x = new Random();
    do
    {
      y = x.nextInt(fiofn.intValue() - 1);
      String z = Integer.toString(y);
      e = new BigInteger(z);
      gcd = fiofn.gcd(e);
      intGCD = gcd.intValue();
    } while(y <= 2 || intGCD != 1);
    return e;
    }
}
```

Output:

```
Enter a prime number:
7
Enter another prime number:
5
Encryption keys are: 14, 35
Decryption keys are: 19, 35
```

16
16/7

Experiment-9

Aim: Write a program to Implement Diffie-Hellman key exchange mechanism using HTML and JavaScript, Consider the end-user as one of the parties (Alice) and the JavaScript application as other party (bob

Program:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.8">
<title> Diffie-Hellman key Exchange</title>
<style>
  body
  {
    font-family: Arial, sans-serif;
    margin:0
    padding: 20px;
    background-color: #f2f2f2;
  }
  h2
  {
    color: #333;
  }
  .container
  {
      max-width: 400px;
      margin: 0 auto;
      background-color: #fff;
      border-radius: 8px;
      padding: 20px;
      box-shadow: 0 0 10px rgba(0,0,0,0.1);
  }
```

```
input[type="text"]
{
  width: calc(100% - 40px);
  Padding: 10px;
  margin-bottom: 10px;
  border: 1px solid #ccc;
  border-radius: 4px;
  font-size: 16px;
}
button
{
  width: 100%
  padding: 10px
  background-color: #007bff;
  border: none;
  border-radius: 4px;
  color: #fff;
  font-size: 16px;
  cursor: pointer;
  transition: background-color 0.3s;
}
button:hover
{
  background-color: #0056b3;
}
#sharedSecret
{
  margin-top: 20px;
  font-size: 18px;
}
</style>
</head>
<body>
<div class="container">
  <h2>Diffie-Hellman Key Exchange</h2>
  <p>Enter Alice's private key:</p>
```

```
<input type = "text" id = "alice private ke
placeholder = "Alice's private key">
   <p>Enter Bob's private key : </p>
<input type = "text" id = "bob private key"
placeholder = "Bob's private key ">
<button on click = "perform key excha-
-nge()" > perform key Exchange </butt-
-on>
<p id = "Shared secret"></p>
<div>
<script>
function perform key Exchange()
{
  Const alice_private key = document.get Elem-
  -ent By Id ('alice_private key')
  Const Bob private key = document.get Element
  By Id ('Bob_private key')
  //Generate Bob's private key (random
  number)
  // Const bob private key = math.floor(mat-
  -h.random()*100)+1);
  Const p = 23; // prime numbers
  Const g = 5; // primitive root
  //calculate public Keys
  Const alice public key = mod pow(g, alice private
  key, p);
  Const bob public key = mod pow(g, bob private
  key, p)
  //calculate shared Secret
```

```
const sharedSecret = modPow(bob public key, alice private key, p);

// Display shared secret
document.getElementById('sharedSecret').innerText = "Shared Secret

}

function modPow(base, exponent, modulus)
{
    if(modulus == 1) return 0;
    let result = 1;
    base = base % modulus;
    while(exponent > 0)
    {
        if(exponent % 2 == 1){
            result = (result + base) % modulus;
        }
        exponent = exponent >> 1;
        base = (base * base) % modulus;
}
    return result;
}
</script>
</body>
</html>
```