**INDEX**

|  |  |  |
| --- | --- | --- |
| **S.NO.** | **TOPIC** | **PAGE NUMBER** |
| 1 | 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 displays the result. | 1 |
| 2 | 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 | 2 |
| 3 | Write a Java program to perform encryption and decryption using the following algorithms:   1. **Ceaser Cipher** 2. **Substitution Cipher** 3. **Hill Cipher** | 3-9 |
| 4 | Write a Java program to implement the DES algorithm logic | 10-12 |
| 5 | Write a C/JAVA program to implement the BlowFish algorithm logic | 13-14 |
| 6 | Write a C/JAVA program to implement the Rijndael algorithm logic. | 15 |
| 7 | Using Java Cryptography, encrypt the text “Hello world” using BlowFish. Create your own key using Java keytool. | 17-18 |
| 8 | Write a Java program to implement RSA Algoithm | 19 |
| 9 | Implement the 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). | 21-22 |
| 10 | Calculate the message digest of a text using the SHA-1 algorithm in JAVA. | 23-24 |
| 11 | Calculate the message digest of a text using the SHA-1 algorithm in JAVA. | 25-26 |

# **XOR a string with a Zero**

**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> main()

{

char str[]="Hello World"; char str1[11];

int i,len; len=strlen(str); for(i=0;i<len;i++)

{

str1[i]=str[i]^0; printf("%c",str1[i]);

}

printf("\n");

}

**Output:** Hello World Hello World

# **XOR a string with a 127**

**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> void main()

{

char str[]="Hello World"; char str1[11];

char str2[11]=str[]; int i,len;

len = strlen(str);

for(i=0;i<len;i++)

{

str1[i] = str[i]&127; printf("%c",str1[i]);

}

printf("\n");

for(i=0;i<len;i++)

{

str3[i] = str2[i]^127; printf("%c",str3[i]);

}

printf("\n");

}

**Output:**

Hello World Hello World Hello World

# **Encryption & Decryption using Cipher Algorithms**

**AIM:** Write a Java program to perform encryption and decryption using the following algorithms:

* 1. Ceaser Cipher
  2. Substitution Cipher
  3. Hill Cipher

**PROGRAM:**

* 1. **Ceaser Cipher**

import java.io.BufferedReader; import java.io.IOException;

import java.io.InputStreamReader; import java.util.Scanner;

public class CeaserCipher {

static Scanner sc=new Scanner(System.in);

static BufferedReader br = new BufferedReader(new InputStreamReader(System.in)); public static void main(String[] args) throws IOException {

// TODO code application logic here

System.out.print("Enter any String: "); String str = br.readLine();

System.out.print("\nEnter the Key: "); int key = sc.nextInt();

String encrypted = encrypt(str, key); System.out.println("\nEncrypted String is: " +encrypted);

String decrypted = decrypt(encrypted, key); System.out.println("\nDecrypted String is: "

+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)) {

c = c - (key % 26);

if (c < 'a')

}

c = c + 26;

decrypted += (char) c;

}

return decrypted;

}

}

**Output:**

Enter any String: Hello World Enter the Key: 5

Encrypted String is: MjqqtBtwqi Decrypted String is: Hello World

**b) Substitution Cipher**

**PROGRAM:**

import java.io.\*; import java.util.\*;

public class SubstitutionCipher {

static Scanner sc = new Scanner(System.in);

static BufferedReader br = new BufferedReader(new InputStreamReader(System.in)); public static void main(String[] args) throws IOException {

// TODO code application logic here String a = "abcdefghijklmnopqrstuvwxyz"; String b = "zyxwvutsrqponmlkjihgfedcba";

System.out.print("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(j);

}

System.out.println("The encrypted data is: " +decrypt);

}

}

**Output:**

Enter any string: aceho

The encrypted data is: zxvsl

**a) 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 BufferedReader br = new BufferedReader(new InputStreamReader(System.in)); static Scanner sc = new Scanner(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];

}

inverse();

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("\nDecrypted 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 IOException { System.out.println("Enter 3x3 matrix for key (It should be inversible): "); for(int i=0;i<3;i++)

for(int j=0;j<3;j++) a[i][j] = sc.nextFloat();

System.out.print("\nEnter 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() { floatp,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][i]; }

System.out.println(""); System.out.println("\nInverse Matrix is : "); 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 a 3 letter string: hai Encrypted string is :fdx Inverse Matrix is :

0.083333336 0.41666666 -0.33333334

-0.41666666 -0.083333336 0.6666667

0.5833333 -0.083333336 -0.33333334

Decrypted string is: hai

# **Java program for DES algorithm logic**

**AIM:** Write a Java program to implement the DES algorithm logic.

**PROGRAM:**

import java.util.\*;

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.DESedeKeySpec; import sun.misc.BASE64Decoder;

import sun.misc.BASE64Encoder; public class DES {

private static final String UNICODE\_FORMAT = "UTF8";

public static final String DESEDE\_ENCRYPTION\_SCHEME = "DESede"; privateKeySpecmyKeySpec; privateSecretKeyFactorymySecretKeyFactory;

private Cipher cipher; byte[] keyAsBytes;

private String myEncryptionKey; private String myEncryptionScheme; SecretKey key;

static BufferedReader br = new BufferedReader(new InputStreamReader(System.in)); public DES() throws Exception {

// TODO code application logic here myEncryptionKey

= "ThisIsSecretEncryptionKey"; myEncryptionScheme = DESEDE\_ENCRYPTION\_SCHEME; keyAsBytes =

myEncryptionKey.getBytes(UNICODE\_FORMAT); myKeySpec

= new DESedeKeySpec(keyAsBytes);

mySecretKeyFactory = SecretKeyFactory.getInstance(myEncryptionScheme); cipher = Cipher.getInstance(myEncryptionScheme);

key = mySecretKeyFactory.generateSecret(myKeySpec);

}

public String encrypt(String unencryptedString)

{ String encryptedString = null;

try {

cipher.init(Cipher.ENCRYPT\_MODE, key);

byte[] plainText = unencryptedString.getBytes(UNICODE\_FORMAT); byte[] encryptedText = cipher.doFinal(plainText);

BASE64Encoder base64encoder = new BASE64Encoder(); encryptedString = base64encoder.encode(encryptedText); }

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

public String decrypt(String encryptedString)

{ String decryptedText=null;

try {

cipher.init(Cipher.DECRYPT\_MODE, key);

BASE64Decoder base64decoder = new BASE64Decoder(); byte[] encryptedText = base64decoder.decodeBuffer(encryptedString); byte[] plainText = cipher.doFinal(encryptedText); decryptedText= bytes2String(plainText); }

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

private static String bytes2String(byte[] bytes)

{ StringBufferstringBuffer = new StringBuffer(); for (int i = 0; i <bytes.length;

i++) { stringBuffer.append((char) bytes[i]); } returnstringBuffer.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(stringToEncrypt); String decrypted = myEncryptor.decrypt(encrypted); System.out.println("\nString To Encrypt: " +stringToEncrypt); System.out.println("\nEncrypted Value : " +encrypted);

System.out.println("\nDecrypted Value : " +decrypted); System.out.println("");

}

}

**OUTPUT:**

Enter the string: Welcome String To Encrypt: Welcome

Encrypted Value : BPQMwc0wKvg= Decrypted Value : Welcome

# **Program to implement BlowFish algorithm logic**

**AIM:** Write a C/JAVA program to implement the BlowFish algorithm logic.

**PROGRAM:**

import java.io.\*;

import java.io.FileInputStream; import java.io.FileOutputStream; import java.security.Key;

import javax.crypto.Cipher;

import javax.crypto.CipherOutputStream; import javax.crypto.KeyGenerator;

import sun.misc.BASE64Encoder; public class BlowFish {

public static void main(String[] args) throws Exception {

// TODO code application logic here KeyGeneratorkeyGenerator = KeyGenerator.getInstance("Blowfish"); keyGenerator.init(128); Key secretKey = keyGenerator.generateKey();

Cipher cipherOut = Cipher.getInstance("Blowfish/CFB/NoPadding"); cipherOut.init(Cipher.ENCRYPT\_MODE, secretKey); BASE64Encoder encoder = new BASE64Encoder();

byte iv[] = cipherOut.getIV(); if (iv != null) {

System.out.println("Initialization Vector of the Cipher: " + encoder.encode(iv)); }

FileInputStream fin = new FileInputStream("inputFile.txt"); FileOutputStreamfout = new FileOutputStream("outputFile.txt"); CipherOutputStreamcout = new CipherOutputStream(fout, cipherOut); int input = 0;

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

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

**OUTPUT:**

Initialization Vector of the Cipher: dI1MXzW97oQ= Contents of inputFile.txt: Hello World

Contents of outputFile.txt: ùJÖ˜ NåI“

# **Program to implement Rijndael algorithm logic**

**AIM:** Write a C/JAVA program to implement the Rijndael algorithm logic.

**PROGRAM:**

import java.security.\*; import javax.crypto.\*; import javax.crypto.spec.\*; import java.io.\*;

public class AES {

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 message="AES still rocks!!";

// Get the KeyGenerator

KeyGenerator kgen = KeyGenerator.getInstance("AES"); kgen.init(128); // 192 and 256 bits may not be available

// Generate the secret key specs. SecretKey skey = kgen.generateKey(); byte[] raw = skey.getEncoded();

SecretKeySpec skeySpec = new SecretKeySpec(raw, "AES");

// Instantiate the cipher

Cipher cipher = Cipher.getInstance("AES"); cipher.init(Cipher.ENCRYPT\_MODE, skeySpec);

byte[] encrypted = cipher.doFinal((args.length == 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:**

Input your message: Hello KGRCET Encrypted text: 3ooo&&(\*&\*4r4 Decrypted text: Hello KGRCET

# **Encrypt a string using BlowFish algorithm**

**AIM:** Using Java Cryptography, encrypt the text “Hello world” using BlowFish. Create your own key using Java keytool.

**PROGRAM:**

import javax.crypto.Cipher; import javax.crypto.KeyGenerator; import javax.crypto.SecretKey; import javax.swing.JOptionPane; public class BlowFishCipher {

public static void main(String[] args) throws Exception {

// create a key generator based upon the Blowfish cipher KeyGeneratorkeygenerator = KeyGenerator.getInstance("Blowfish");

// create a key

// create a cipher based upon Blowfish Cipher cipher = Cipher.getInstance("Blowfish");

// initialise cipher to with secret key cipher.init(Cipher.ENCRYPT\_MODE, secretkey);

// get the text to encrypt

String inputText = JOptionPane.showInputDialog("Input your message: "); // encrypt message

byte[] encrypted = cipher.doFinal(inputText.getBytes());

// re-initialise the cipher to be in decrypt mode cipher.init(Cipher.DECRYPT\_MODE, secretkey);

// decrypt message

byte[] decrypted = cipher.doFinal(encrypted);

// and display the results

JOptionPane.showMessageDialog(JOptionPane.getRootFrame(), "\nEncrypted text: " + new String(encrypted) + "\n" + "\nDecrypted text: " + new String(decrypted));

System.exit(0);

} }

**OUTPUT:**

Input your message: Hello world Encrypted text: 3ooo&&(\*&\*4r4 Decrypted text: Hello world

# **RSA Algorithm**

**AIM:** Write a Java program to implement RSA Algoithm.

**PROGRAM:**

import java.io.BufferedReader; import java.io.InputStreamReader; import java.math.\*;

import java.util.Random; import java.util.Scanner; public class RSA {

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

// TODO code application logic here System.out.print("Enter a Prime number: ");

BigInteger p = sc.nextBigInteger(); // Here's one prime number.. System.out.print("Enter another prime number: "); BigInteger q = sc.nextBigInteger(); // ..and another.

BigInteger n = p.multiply(q);

BigInteger n2 = p.subtract(BigInteger.ONE).multiply(q.subtract(BigInteger.ONE)); BigInteger e = generateE(n2);

BigInteger d = e.modInverse(n2); // Here's the multiplicative inverse

System.out.println("Encryption keys are: " + e + ", " + n); System.out.println("Decryption keys are: " + d + ", " + n);

}

public static BigIntegergenerateE(BigIntegerfiofn) { 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: 5

Enter another prime number: 11 Encryption keys are: 33, 55

Decryption keys are: 17, 55

# **Diffie-Hellman**

**AIM:** Implement the 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:**

import java.math.BigInteger; import java.security.KeyFactory; import java.security.KeyPair;

import java.security.KeyPairGenerator; import java.security.SecureRandom;

import javax.crypto.spec.DHParameterSpec; import javax.crypto.spec.DHPublicKeySpec; public class DiffeHellman {

public final static int pValue = 47; public final static int gValue = 71; public final static int XaValue = 9; public final static int XbValue = 14;

public static void main(String[] args) throws Exception

{ // TODO code application logic here

BigInteger p = new BigInteger(Integer.toString(pValue)); BigInteger g = new BigInteger(Integer.toString(gValue)); BigIntegerXa = new BigInteger(Integer.toString(XaValue)); BigIntegerXb = new BigInteger(Integer.toString(XbValue)); createKey(); intbitLength = 512; // 512 bits

SecureRandomrnd = new SecureRandom();

p = BigInteger.probablePrime(bitLength, rnd); g = BigInteger.probablePrime(bitLength, rnd);

createSpecificKey(p, g);

}

public static void createKey() throws Exception {

KeyPairGeneratorkpg = KeyPairGenerator.getInstance("DiffieHellman"); kpg.initialize(512);

KeyPairkp = kpg.generateKeyPair();

KeyFactorykfactory = KeyFactory.getInstance("DiffieHellman"); DHPublicKeySpeckspec = (DHPublicKeySpec) kfactory.getKeySpec(kp.getPublic(), DHPublicKeySpec.class);

System.out.println("Public key is: " +kspec);

}

public static void createSpecificKey(BigInteger p, BigInteger g) throws Exception { KeyPairGeneratorkpg = KeyPairGenerator.getInstance("DiffieHellman"); DHParameterSpecparam = new DHParameterSpec(p, g); kpg.initialize(param);

KeyPairkp = kpg.generateKeyPair();

KeyFactorykfactory = KeyFactory.getInstance("DiffieHellman"); DHPublicKeySpeckspec = (DHPublicKeySpec) kfactory.getKeySpec(kp.getPublic(), DHPublicKeySpec.class);

System.out.println("\nPublic key is : " +kspec);

}

}

**OUTPUT:**

Public key is: javax.crypto.spec.DHPublicKeySpec@5afd29 Public key is: javax.crypto.spec.DHPublicKeySpec@9971ad

# **SHA-1**

**AIM:** Calculate the message digest of a text using the SHA-1 algorithm in JAVA.

**PROGRAM:**

import java.security.\*; public class SHA1 {

public static void main(String[] a) { try {

MessageDigest md = MessageDigest.getInstance("SHA1"); System.out.println("Message digest object info: "); System.out.println(" Algorithm = " +md.getAlgorithm()); System.out.println(" Provider = " +md.getProvider()); System.out.println(" ToString = " +md.toString());

String input = ""; md.update(input.getBytes()); byte[] output = md.digest(); System.out.println();

System.out.println("SHA1(\""+input+"\") = " +bytesToHex(output));

input = "abc"; md.update(input.getBytes()); output = md.digest(); System.out.println();

System.out.println("SHA1(\""+input+"\") = " +bytesToHex(output));

input = "abcdefghijklmnopqrstuvwxyz"; md.update(input.getBytes());

output = md.digest(); System.out.println();

System.out.println("SHA1(\"" +input+"\") = " +bytesToHex(output)); System.out.println(""); }

catch (Exception e) {

System.out.println("Exception: " +e);

}

}

public static String bytesToHex(byte[] b) {

char hexDigit[] = {'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'D', 'E', 'F'};

StringBufferbuf = new StringBuffer(); for (int j=0; j<b.length; j++) { buf.append(hexDigit[(b[j] >> 4) & 0x0f]); buf.append(hexDigit[b[j] & 0x0f]); } returnbuf.toString(); }

}

**OUTPUT:**

Message digest object info: Algorithm = SHA1

Provider = SUN version 1.6

ToString = SHA1 Message Digest from SUN, <initialized> SHA1("") = DA39A3EE5E6B4B0D3255BFEF95601890AFD80709 SHA1("abc") = A9993E364706816ABA3E25717850C26C9CD0D89D

SHA1("abcdefghijklmnopqrstuvwxyz")=32D10C7B8CF96570CA04CE37F2A19D8424 0D3A89

# **Message Digest Algorithm5 (MD5)**

**AIM:** Calculate the message digest of a text using the SHA-1 algorithm in JAVA.

**PROGRAM:**

import java.security.\*; public class MD5 {

public static void main(String[] a) {

// TODO code application logic here

try {

MessageDigest md = MessageDigest.getInstance("MD5"); System.out.println("Message digest object info: "); System.out.println(" Algorithm = " +md.getAlgorithm()); System.out.println(" Provider = " +md.getProvider()); System.out.println(" ToString = " +md.toString());

String input = ""; md.update(input.getBytes());

byte[] output = md.digest(); System.out.println();

System.out.println("MD5(\""+input+"\") = " +bytesToHex(output));

input = "abc"; md.update(input.getBytes()); output = md.digest(); System.out.println();

System.out.println("MD5(\""+input+"\") = " +bytesToHex(output));

input = "abcdefghijklmnopqrstuvwxyz"; md.update(input.getBytes());

output = md.digest(); System.out.println();

System.out.println("MD5(\"" +input+"\") = "

+bytesToHex(output)); System.out.println("");

}

catch (Exception e) { System.out.println("Exception: " +e); }

}

public static String bytesToHex(byte[] b) {

char hexDigit[] = {'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'D', 'E', 'F'};

StringBufferbuf = new StringBuffer(); for (int j=0; j<b.length; j++) { buf.append(hexDigit[(b[j] >> 4) & 0x0f]); buf.append(hexDigit[b[j] & 0x0f]); }

return buf.toString(); } }

**OUTPUT:**

Message digest object info: Algorithm = MD5

Provider = SUN version 1.6

ToString = MD5 Message Digest from SUN, <initialized> MD5("") = D41D8CD98F00B204E9800998ECF8427E MD5("abc") =

900150983CD24FB0D6963F7D28E17F72 MD5("abcdefghijklmnopqrstuvwxyz")

= C3FCD3D76192E4007DFB496CCA67E13B