1a. 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.

Program

```
#include <stdio.h>
#include <string.h>
int main()
{
   char s[20] = "helloWorld";
   char result[10];
   int i, len;
   len = strlen(s);
   for (i = 0; i < len; i++) {
      result[i] = s[i] ^ 0;
      printf("%c", result[i]);
   }
}
```

Output:

1b. Write a Java program to implement the DES algorithm logic

Program:

```
import java.util.*;
import javax.crypto.*;
import java.io.*;
import java.security.*;
import java.security.spec.*;
public class DES {
   public static void main(String[] args) throws Exception {
       String msg = "welcome";
       byte[] myMsg = msg.getBytes();
       KeyGenerator kg = KeyGenerator.getInstance("DES");
       SecretKey sk = kg.generateKey();
       Cipher cipher = Cipher.getInstance("DES");
       cipher.init(Cipher.ENCRYPT_MODE, sk);
       byte[] encryptedBytes = cipher.doFinal(myMsg);
       cipher.init(cipher.DECRYPT_MODE, sk);
       byte[] decryptedBytes = cipher.doFinal(encryptedBytes);
       String encryptedData = new String(encryptedBytes);
       String decryptedData = new String(decryptedBytes);
       System.out.println("Message : " + msg);
       System.out.println("Encrypted Data : " + encryptedData);
       System.out.println("Decrypted Data : " + decryptedData);
   }
}
```

Output:

2a. 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 <string.h>
int main()
{
   char s[20] = "helloWorld";
   char result[10];
   int i, len;
   len = strlen(s);
   for (i = 0; i < len; i++)
   {
      result[i] = s[i] & 127;
      printf("%c", result[i]);
   }
   printf("\n");
   for (i = 0; i < len; i++)
   {
      result[i] = s[i] | 127;
      printf("%c", result[i]);
   }
   printf("\n");
   for (i = 0; i < len; i++)
   {
      result[i] = s[i] ^ 127;
      printf("%c", result[i]);
   }
}
```

Output:

2b. Write a Java program to perform encryption and decryption using Substitution Cipher.

Program:

```
import java.io.*;
import java.util.*;

public class Substitution {
    static Scanner sc = new Scanner(System.in);

    public static void main(String[] args) {
        String a = "abcdefghijklmnopqrstuvwxyz";
        String b = "zyxwvutsrqponmlkjihgfedcba";

        System.out.print("Enter a String: ");
        String msg = sc.nextLine();
        String decrypt = "";
        char c;
        for (int i = 0; i < msg.length(); i++) {
            c = msg.charAt(i);
            int j = a.indexOf(c);
            decrypt += b.charAt(i);
        }
        System.out.println("The encrypted data is " + decrypt);
    }
}
```

Output:

3a. Write a Java program to perform encryption and decryption using Ceaser Cipher.

Program:

```
import java.io.*;
import java.util.*;

public class Ceaser {

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

    public static void main(String[] args) {
        System.out.print("Enter a String: ");
        String msg = sc.nextLine();
        System.out.print("Enter the key: ");
        int key = sc.nextInt();
        String encrypted = encrypt(msg, key);
        System.out.println("Encrypted String is : " + encrypted);
        String decrypted = decrypt(encrypted, key);
        System.out.println("Decrypted String is : " + decrypted);
    }

    public static String encrypt(String msg, int key) {
        String encrypted = "";
        for (int i = 0; i < msg.length(); i++) {
            int c = msg.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 msg, int key) {
        String decrypted = "";
        for (int i = 0; i < msg.length(); i++) {
            int c = msg.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:

3b. Write a C/JAVA program to implement the Rijndael algorithm logic.

Program:

```
import javax.crypto.*;
import javax.crypto.spec.*;
import java.io.*;

public class AES {
  public static String asHex(byte buf[]) {
    StringBuffer buff = new StringBuffer(buf.length * 2);
    for (int i = 0; i < buf.length; i++) {
      if (((int) buf[i] & 0xff) < 0x10)
        buff.append("0");
      buff.append(Long.toString((int) buf[i] & 0xff, 16));
    }
    return buff.toString();
  }

  public static void main(String[] args) throws Exception {
    String msg = "AES Algorithm";
    KeyGenerator kg = KeyGenerator.getInstance("AES");
    kg.init(128);
    SecretKey sk = kg.generateKey();
    byte[] raw = sk.getEncoded();
    SecretKeySpec skspec = new SecretKeySpec(raw, "AES");
    Cipher cipher = Cipher.getInstance("AES");
    cipher.init(Cipher.ENCRYPT_MODE, skspec);
    byte[] encrypted = cipher.doFinal((args.length == 0 ? msg : args[0]).getBytes());
    System.out.println("Encrypted string : " + asHex(encrypted));
    cipher.init(Cipher.DECRYPT_MODE, skspec);
    byte[] original = cipher.doFinal(encrypted);
    String originalString = new String(original);
    System.out.println("original String in Hexadecimal : " + asHex(original));
    System.out.println("Original String : " + originalString);

  }
}
```

Output:

4. Write a Java program to perform encryption and decryption using Hill Cipher

Program:

```
import java.io.*;
import java.util.*;

public class Hill {
   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 {
      getkeymer();
      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("Decrypted string is: ");
      for (int i = 0; i < 3; i++)
         System.out.print((char) (decrypt[i][0] % 26 + 97));
      System.out.println();
   }

   public static void getkeymer() throws IOException {
      System.out.println("Enter 3 X 3 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.nextInt();

      System.out.print("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++)
                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][k] * q - p * c[k][j];
                        b[i][j] = b[i][k] * q - p * b[k][j];
                    }
            }

        for (int i = 0; i < 3; i++)
            for (int j = 0; j < 3; j++)
                b[i][j] /= c[i][j];

        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.println();
        }
    }
}
```

Output:

5a. Write a C/JAVA program to implement the Blow Fish algorithm logic

Program:

```
import java.io.*;
import java.security.*;
import java.util.*;
import java.util.Base64.Encoder;

import javax.crypto.*;
public class BlowFish {
  public static void main(String[] args) throws Exception{
    KeyGenerator kg = KeyGenerator.getInstance("BlowFish");
    kg.init(128);
    SecretKey sk = kg.generateKey();
    Cipher cipher = Cipher.getInstance("BlowFish");
    cipher.init(Cipher.ENCRYPT_MODE, sk);
    Encoder encoder = Base64.getEncoder();
    byte iv[] = cipher.getIV();
    if(iv != null) {
      System.out.println("Initialisation Vector " + encoder.encodeToString(iv));
      FileInputStream fin = new FileInputStream("inputFile.txt");
      FileOutputStream fout = new FileOutputStream("OutputFile.txt");
      CipherOutputStream cout = new CipherOutputStream(fout, cipher);
      int input = 0;
      while((input = fin.read()) != -1) cout .write(input);
      fin.close();
      cout.close();
    }

  }
}
```

Output:

5b. Write a Java program to implement RSA Algorithm

Program:

```
import java.math.*;
import java.util.*;

public class RSA {
    public static void main(String[] args) {
        int p = 3, q = 11, n, z, d = 0, e, i;
        int msg = 12;
        double c;
        BigInteger msgBack;
        n = p * q;
        z = (p - 1) * (q - 1);
        System.out.println("Value of z : " + z);
        for (e = 2; e < z; e++)
            if (gcd(e, z) == 1)
                break;

        System.out.println("Value of e : " + e);
        for (i = 0; i <= q; i++) {
            int x = 1 + (i * z);
            if (x % e == 0) {
                d = x / e;
                break;
            }
        }
        System.out.println("The value of d : " + d);
        c = Math.pow(msg, e) % n;
        System.out.println("Encrypted msg is " + c);
        BigInteger N = BigInteger.valueOf(n);
        BigInteger C = BigDecimal.valueOf(c).toBigInteger();
        msgBack = (C.pow(d)).mod(N);
        System.out.println("Decrypted msg : " + msgBack);
    }

    static int gcd(int a, int b) {
        if (a == 0)
            return b;
        return gcd(a % b, a);
    }

}
```

Output:

6. Using Java Cryptography, encrypt the text “Hello world” using BlowFish. Create your own key using Java key tool.

Program:

```
import javax.crypto.*;
import javax.swing.*;

public class BlowFish1 {
   public static void main(String[] args) throws Exception {
      KeyGenerator kg = KeyGenerator.getInstance("BlowFish");
      SecretKey sk = kg.generateKey();
      Cipher cipher = Cipher.getInstance("BlowFish");
      String inputText = JOptionPane.showInputDialog("Input Your Message : ");
      byte[] encrypted = cipher.doFinal(inputText.getBytes());
      cipher.init(Cipher.DECRYPT_MODE, sk);
      byte[] decrypted = cipher.doFinal(encrypted);
      JOptionPane.showMessageDialog(JOptionPane.getRootFrame(),
            "Encrypted text : " + new String(encrypted) + "\n" + "Decrypted text : " + new
String(decrypted));
      System.exit(0);
   }
}
```

Output:

7. 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.*;
import java.security.*;
import javax.crypto.spec.*;

public class DiffeHellman {
   public static void main(String[] args) throws Exception {
      BigInteger p = new BigInteger(Integer.toString(47));
      BigInteger g = new BigInteger(Integer.toString(71));
      createKey();
      int bitLength = 512;
      SecureRandom rnd = new SecureRandom();
      p = BigInteger.probablePrime(bitLength, rnd);
      g = BigInteger.probablePrime(bitLength, rnd);
      createSpecificKey(p, g);
   }

   public static void createKey() throws Exception {
      KeyPairGenerator kpg = KeyPairGenerator.getInstance("DiffieHellman");
      kpg.initialize(512);
      KeyPair kp = kpg.generateKeyPair();
      KeyFactory kfactory = KeyFactory.getInstance("DiffieHellman");
      DHPublicKeySpec kspec = (DHPublicKeySpec) kfactory.getKeySpec(kp.getPublic(),
DHPublicKeySpec.class);
      System.out.println("Public key is: " + kspec);
   }

   public static void createSpecificKey(BigInteger p, BigInteger g) throws Exception {
      KeyPairGenerator kpg = KeyPairGenerator.getInstance("DiffieHellman");
      DHParameterSpec param = new DHParameterSpec(p, g);
      kpg.initialize(param);
      KeyPair kp = kpg.generateKeyPair();
      KeyFactory kfactory = KeyFactory.getInstance("DiffieHellman");
      DHPublicKeySpec kspec = (DHPublicKeySpec) kfactory.getKeySpec(kp.getPublic(),
            DHPublicKeySpec.class);
      System.out.println("Public key is : " + kspec);
   }
}
```

Output:

8. 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' };
    StringBuffer buf = 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: