Problem Statement: 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.

Code:

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
   string str = "Hello World";
   cout<<"\t Original String --> "<<str<<endl;
   cout<<"\t String after AND operation --> ";
   for(int i=0;i<str.size();i++)
   {
      printf("%c",str[i]&127);
   }
   cout<<endl;
   cout<<"\t String after OR operation --> ";
   for(int i=0;i<str.size();i++)
   {
      printf("%c",str[i]|127);
   }
   cout<<endl;
   cout<<"\t String after XOR operation --> ";
   for(int i=0;i<str.size();i++)
   {
      printf("%c",str[i]^127);
   }
   cout<<endl;
   return 0;
}
```

Sample Output:

```
Original String --> Hello World
String after AND operation --> Hello World
String after OR operation --> ⌂⌂⌂⌂⌂⌂⌂⌂⌂⌂⌂
String after XOR operation --> 7→!!!►_(►
```

Problem Statement: Write a Java program to implement DES algorithm.

Code:

```
import java.util.*;
import javax.crypto.BadPaddingException;
import javax.crypto.Cipher;
import javax.crypto.IllegalBlockSizeException;
import javax.crypto.KeyGenerator;
import javax.crypto.NoSuchPaddingException;
import javax.crypto.SecretKey;
import javax.crypto.SecretKeyFactory;
import javax.crypto.spec.DESKeySpec;
import java.io.*;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.security.spec.InvalidKeySpecException;

class DES{
   public static void main(String[] args) throws IOException, NoSuchAlgorithmException,
InvalidKeyException, InvalidKeySpecException, NoSuchPaddingException, IllegalBlockSizeException,
BadPaddingException {

      String  message="This is a confidential message.";
      byte[] myMessage =message.getBytes();
      KeyGenerator Mygenerator = KeyGenerator.getInstance("DES");
      SecretKey myDesKey = Mygenerator.generateKey();
      Cipher myCipher = Cipher.getInstance("DES");
      myCipher.init(Cipher.ENCRYPT_MODE, myDesKey);
      byte[] myEncryptedBytes=myCipher.doFinal(myMessage);
      myCipher.init(Cipher.DECRYPT_MODE, myDesKey);
      byte[] myDecryptedBytes=myCipher.doFinal(myEncryptedBytes);

      String encrypteddata=new String(myEncryptedBytes);
      String decrypteddata=new String(myDecryptedBytes);

      System.out.println("Message : "+ message+"\n");
      System.out.println("Encrypted - "+ encrypteddata+"\n");
      System.out.println("Decrypted Message - "+ decrypteddata);
   }
}
```

Sample Output:

```
Message : This is a confidential message.
Encrypted - ·~M???5??|3?5xv~?   ???·????[????U
Decrypted Message - This is a confidential message.
```

Problem Statement: Write a Java program to implement AES algorithm.

Code:

```
import javax.crypto.Cipher;
import javax.crypto.SecretKey;
import javax.crypto.SecretKeyFactory;
import javax.crypto.spec.IvParameterSpec;
import javax.crypto.spec.PBEKeySpec;
import javax.crypto.spec.SecretKeySpec;
import java.nio.charset.StandardCharsets;
import java.security.InvalidAlgorithmParameterException;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.security.spec.InvalidKeySpecException;
import java.security.spec.KeySpec;
import java.util.Base64;
import javax.crypto.BadPaddingException;
import javax.crypto.IllegalBlockSizeException;
import javax.crypto.NoSuchPaddingException;
public class AESExample
{
   private static final String SECRET_KEY = "123456789";
   private static final String SALTVALUE = "abcdefg";
   public static String encrypt(String strToEncrypt)
   {
   try
   {
     byte[] iv = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
     IvParameterSpec ivspec = new IvParameterSpec(iv);
     SecretKeyFactory factory = SecretKeyFactory.getInstance("PBKDF2WithHmacSHA256");
     KeySpec spec = new PBEKeySpec(SECRET_KEY.toCharArray(), SALTVALUE.getBytes(), 65536, 256);
     SecretKey tmp = factory.generateSecret(spec);
     SecretKeySpec secretKey = new SecretKeySpec(tmp.getEncoded(), "AES");
     Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
     cipher.init(Cipher.ENCRYPT_MODE, secretKey, ivspec);
     return Base64.getEncoder()
.encodeToString(cipher.doFinal(strToEncrypt.getBytes(StandardCharsets.UTF_8)));
   }
   catch (InvalidAlgorithmParameterException | InvalidKeyException | NoSuchAlgorithmException |
InvalidKeySpecException | BadPaddingException | IllegalBlockSizeException |
NoSuchPaddingException e)
   {
     System.out.println("Error occured during encryption: " + e.toString());
   }
   return null;
   }
```

```
   public static String decrypt(String strToDecrypt)
   {
   try
   {
     byte[] iv = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
     IvParameterSpec ivspec = new IvParameterSpec(iv);
     SecretKeyFactory factory = SecretKeyFactory.getInstance("PBKDF2WithHmacSHA256");
     KeySpec spec = new PBEKeySpec(SECRET_KEY.toCharArray(), SALTVALUE.getBytes(), 65536, 256);
     SecretKey tmp = factory.generateSecret(spec);
     SecretKeySpec secretKey = new SecretKeySpec(tmp.getEncoded(), "AES");
     Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5PADDING");
     cipher.init(Cipher.DECRYPT_MODE, secretKey, ivspec);
     return new String(cipher.doFinal(Base64.getDecoder().decode(strToDecrypt)));
   }
   catch (InvalidAlgorithmParameterException | InvalidKeyException | NoSuchAlgorithmException |
InvalidKeySpecException | BadPaddingException | IllegalBlockSizeException |
NoSuchPaddingException e)
   {
     System.out.println("Error occured during decryption: " + e.toString());
   }
   return null;
   }
   public static void main(String[] args)
   {
      String originalval = "AES Encryption";
      String encryptedval = encrypt(originalval);
      String decryptedval = decrypt(encryptedval);
      System.out.println("Original value: " + originalval+"\n");
      System.out.print("Encrypted value: " + encryptedval+"\n");
      System.out.println("Decrypted value: " + decryptedval);
   }
}
```

Sample Output:

```
Original value: AES Encryption
Encrypted value: V5E9I52IxhMaW4+hJhl56g==
Decrypted value: AES Encryption
```

Problem Statement: Write a C++ program to implement RSA algorithm.

Code:

```
#include<bits/stdc++.h>
using namespace std;
int gcd(int a, int b) {
  int t;
  while(1) {
    t= a%b;
    if(t==0)
    return b;
    a = b;
    b= t;
  }
}
int main() {
  double p = 13, q=11;
  double n=p*q;
  double track, phi= (p-1)*(q-1),e=7;
  while(e<phi) {
    track = gcd(e,phi);
    if(track==1) break;
    else e++;
  }
  double d1=1/e; double d=fmod(d1,phi); double message = 9; double c = pow(message,e);
  double m = pow(c,d); c=fmod(c,n);   m=fmod(m,n);
  cout<<"Original Message = "<<message<<"\n"<<"p = "<<p<<"\n"<<"q = "<<q;
  cout<<"\n"<<"n = pq = "<<n<<"\n"<<"phi = "<<phi<<"\n"<<"e = "<<e<<"\n"<<"d = "<<d;
  cout<<"\n"<<"Encrypted message = "<<c;
  cout<<"\n"<<"Decrypted message = "<<m;
  return 0;
}
```

Sample Output:

```
Original Message = 80
p = 13
q = 11
n = pq = 143
phi = 120
e = 7
d = 0.142857
Encrypted message = 141
Decrypted message = 80
```

Problem Statement: Calculate the message digest of a text using the MD5 algorithm in JAVA.

Code:

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

public class MD5 {
   public static String getMd5(String input)
   {
      try {
         MessageDigest md = MessageDigest.getInstance("MD5");
         byte[] messageDigest = md.digest(input.getBytes());

         BigInteger no = new BigInteger(1, messageDigest);

         String hashtext = no.toString(16);
         while (hashtext.length() < 32) {
            hashtext = "0" + hashtext;
         }
         return hashtext;
      }
      catch (NoSuchAlgorithmException e) {
         throw new RuntimeException(e);
      }
   }
   public static void main(String args[]) throws NoSuchAlgorithmException
   {
      String s = "Nikhil";
      System.out.println("Your HashCode Generated by MD5 is: " + getMd5(s));
   }
}
```

Sample Output:

```
java -cp /tmp/5PDbVbIf93 MD5
Your HashCode Generated by MD5 is: 8d2158205ca96a8aa2cca62a48b224c1
```