AES ALGORITHM

PROGRAM:

```
import java.nio.charset.StandardCharsets;
import java.security.spec.KeySpec;
import java.util.Base64;
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;

class AES {
    // Class private variables
    private static final String SECRET_KEY
        = "my_super_secret_key_ho_ho_ho";

    private static final String SALT = "ssshhhhhhhhhhh!!!!";

    // This method use to encrypt to string
    public static String encrypt(String strToEncrypt)
    {
        try {

            // Create default byte array
            byte[] iv = { 0, 0, 0, 0, 0, 0, 0, 0,
                          0, 0, 0, 0, 0, 0, 0, 0 };
            IvParameterSpec ivspec
                = new IvParameterSpec(iv);

            // Create SecretKeyFactory object
            SecretKeyFactory factory
                = SecretKeyFactory.getInstance(
                    "PBKDF2WithHmacSHA256");

            // Create KeySpec object and assign with
            // constructor
            KeySpec spec = new PBEKeySpec(
                SECRET_KEY.toCharArray(), SALT.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 encrypted string
        return Base64.getEncoder().encodeToString(
            cipher.doFinal(strToEncrypt.getBytes(
                StandardCharsets.UTF_8)));
    }
    catch (Exception e) {
        System.out.println("Error while encrypting: "
                           + e.toString());
    }
    return null;
}

// This method use to decrypt to string
public static String decrypt(String strToDecrypt)
{
    try {

        // Default byte array
        byte[] iv = { 0, 0, 0, 0, 0, 0, 0, 0,
                      0, 0, 0, 0, 0, 0, 0, 0 };
        // Create IvParameterSpec object and assign with
        // constructor
        IvParameterSpec ivspec
            = new IvParameterSpec(iv);

        // Create SecretKeyFactory Object
        SecretKeyFactory factory
            = SecretKeyFactory.getInstance(
                "PBKDF2WithHmacSHA256");

        // Create KeySpec object and assign with
        // constructor
        KeySpec spec = new PBEKeySpec(
            SECRET_KEY.toCharArray(), SALT.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 decrypted string
        return new String(cipher.doFinal(
            Base64.getDecoder().decode(strToDecrypt)));
    }
    catch (Exception e) {
        System.out.println("Error while decrypting: "
                           + e.toString());
    }
    return null;
```

```
    }
}

// driver code
public class Main {
    public static void main(String[] args)
    {
        // Create String variables
        String originalString = "ALFAAZ2321";

        // Call encryption method
        String encryptedString
            = AES.encrypt(originalString);

        // Call decryption method
        String decryptedString
            = AES.decrypt(encryptedString);

        // Print all strings
        System.out.println(originalString);
        System.out.println(encryptedString);
        System.out.println(decryptedString);
    }
}
```

OUTPUT: