Java Program: Compact AES Encryption and Decryption (AES-GCM)

```java
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import javax.crypto.spec.GCMParameterSpec;
import java.nio.ByteBuffer;
import java.security.SecureRandom;
import java.util.Base64;

public class CompactAES {

    private static final String ALGORITHM = "AES/GCM/NoPadding";
    private static final int KEY_SIZE = 256;
    private static final int IV_LENGTH = 12; // 96 bits is recommended
    private static final int TAG_LENGTH = 128; // in bits

    public static String encrypt(String plainText, SecretKey key) throws Exception {
        byte[] iv = new byte[IV_LENGTH];
        new SecureRandom().nextBytes(iv); // Create a random IV

        Cipher cipher = Cipher.getInstance(ALGORITHM);
        GCMParameterSpec gcmSpec = new GCMParameterSpec(TAG_LENGTH, iv);
        cipher.init(Cipher.ENCRYPT_MODE, key, gcmSpec);

        byte[] cipherText = cipher.doFinal(plainText.getBytes());

        // Prepend IV to ciphertext for decryption
        byte[] encrypted = ByteBuffer.allocate(iv.length + cipherText.length)
                .put(iv).put(cipherText).array();

        return Base64.getEncoder().encodeToString(encrypted);
    }

    public static String decrypt(String encryptedText, SecretKey key) throws Exception {
        byte[] decoded = Base64.getDecoder().decode(encryptedText);
        ByteBuffer bb = ByteBuffer.wrap(decoded);

        // Extract the IV from the beginning
        byte[] iv = new byte[IV_LENGTH];
        bb.get(iv);

        // Extract the actual ciphertext
        byte[] cipherText = new byte[bb.remaining()];
        bb.get(cipherText);

        Cipher cipher = Cipher.getInstance(ALGORITHM);
        GCMParameterSpec gcmSpec = new GCMParameterSpec(TAG_LENGTH, iv);
        cipher.init(Cipher.DECRYPT_MODE, key, gcmSpec);

        return new String(cipher.doFinal(cipherText));
    }

    public static void main(String[] args) throws Exception {
        String plainText = "This is a modern secret message.";
        System.out.println("Original: " + plainText);

        KeyGenerator keyGen = KeyGenerator.getInstance("AES");
        keyGen.init(KEY_SIZE);
        SecretKey secretKey = keyGen.generateKey();

        String encryptedText = encrypt(plainText, secretKey);
        System.out.println("Encrypted: " + encryptedText);

        String decryptedText = decrypt(encryptedText, secretKey);
        System.out.println("Decrypted: " + decryptedText);
    }
}
```