Esteban Moya Vargas

Código: A00068020

Tarea #2

Vectores DES

1. 0101010101010101 4BD388FF6CD81D4F 1000000000000000

```
<terminated> DESTest [Java Application] (
Key      : 0101010101010101
Message  : 1000000000000000
Cipher   : 4BD388FF6CD81D4F
Expected: 4BD388FF6CD81D4F
```

2. 0101010101010101 D9031B0271BD5A0A 0080000000000000

```
<terminated> DESTest [Java Application] C:
Key      : 0101010101010101
Message  : 0080000000000000
Cipher   : D9031B0271BD5A0A
Expected: D9031B0271BD5A0A
```

3. 0101010101010101 424250B37C3DD951 0040000000000000

```
Key      : 0101010101010101
Message  : 0040000000000000
Cipher   : 424250B37C3DD951
Expected: 424250B37C3DD951
```

4. 0101010101010101 B8061B7ECD9A21E5 0020000000000000

```
<terminated> DESTest [Java Application] (
Key      : 0101010101010101
Message  : 0020000000000000
Cipher   : B8061B7ECD9A21E5
Expected: B8061B7ECD9A21E5
```

Para la segunda parte de la tarea fue necesario cambiar la siguiente linea de codigo:

```
public static void main(String[] args) {
      String test = "1";
      try {
         byte[] theKey = null;
         byte[] theMsg = null;
         byte[] theExp = null;
         if (test.equals("1")) {
            theKey = hexToBytes("0101010101010101");
            theMsg = hexToBytes("0020000000000000");
            theExp = hexToBytes("B8061B7ECD9A21E5");
         } else if (test.equals("2")) {
            theKey = hexToBytes("38627974656B6579"); // "8bytekey"
            theMsg = hexToBytes("6D6573736167652E"); // "message."
            theExp = hexToBytes("7CF45E129445D451");
         } else {
            System.out.println("Usage:");
            System.out.println("java JceSunDesTest 1/2");
            return;
         }
         KeySpec ks = new DESKeySpec(theKey);
         SecretKeyFactory kf
            = SecretKeyFactory.getInstance("DES");
         SecretKey ky = kf.generateSecret(ks);
         Cipher cf = Cipher.getInstance("DES/ECB/NoPadding");
         cf.init(Cipher.DECRYPT_MODE, ky);
         byte[] theCph = cf.doFinal(theMsg);
         System.out.println("Key     : "+bytesToHex(theKey));
         System.out.println("Message : "+bytesToHex(theMsg));
         System.out.println("Cipher  : "+bytesToHex(theCph));
         System.out.println("Expected: "+bytesToHex(theExp));
      } catch (Exception e) {
         e.printStackTrace();
         return;
```

Vectores DES-DECRYPT

1. 0101010101010101 79E90DBC98F92CCA 0000000000200000

```
Key      : 0101010101010101
Message  : 79E90DBC98F92CCA
Cipher   : 0000000000200000
Expected: 0000000000200000
```

2. 0101010101010101 866ECEDD8072BB0E 0000000000100000

```
<terminated> DESTest [Java Application] C:\Pr
Key      : 0101010101010101
Message  : 866ECEDD8072BB0E
Cipher   : 0000000000100000
Expected: 0000000000100000
```

3. 0101010101010101 8B54536F2F3E64A8 0000000000080000

```
<terminated> DESTest [Java Application] C:\
Key      : 0101010101010101
Message  : 8B54536F2F3E64A8
Cipher   : 0000000000080000
Expected: 0000000000080000
```

4. 0101010101010101 EA51D3975595B86B 0000000000040000

```
<terminated> DESTest [Java Application]
Key      : 0101010101010101
Message  : EA51D3975595B86B
Cipher   : 0000000000040000
Expected: 0000000000040000
```

Para la tercera parte de la tarea fue necesario dejar el código como se presenta a continuación:

```
public static void main(String[] args) {
      String test = "1";
      try {
         byte[] theKey = null;
         byte[] theMsg = null;
         byte[] theExp = null;
         byte[] theIv=null;
         if (test.equals("1")) {
            theKey = hexToBytes("00000000000000000000000000000000");
            theIv=hexToBytes("00000000000000000000000000000000");
            theMsg = hexToBytes("6a84867cd77e12ad07ea1be895c53fa3");
            theExp = hexToBytes("732281c0a0aab8f7a54a0c67a0c45ecf");
         } else if (test.equals("2")) {
            theKey = hexToBytes("38627974656B6579"); // "8bytekey"
            theMsg = hexToBytes("6D6573736167652E"); // "message."
            theExp = hexToBytes("7CF45E129445D451");
         } else {
            System.out.println("Usage:");
            System.out.println("java JceSunDesTest 1/2");
            return;
         }
         SecretKeySpec skeySpec = new SecretKeySpec(theKey, "AES");
         IvParameterSpec ivParameterSpec = new IvParameterSpec(theIv);
         Cipher cf = Cipher.getInstance("AES/CBC/PKCS5Padding");
         cf.init(Cipher.ENCRYPT_MODE,skeySpec,ivParameterSpec);
         byte[] theCph = cf.doFinal(theMsg);
         System.out.println("Key     : "+bytesToHex(theKey));
         System.out.println("Message : "+bytesToHex(theMsg));
         System.out.println("Cipher  : "+bytesToHex(theCph));
         System.out.println("Expected: "+bytesToHex(theExp));
      } catch (Exception e) {
```

VECTORES AES

1)

Key=00000000000000000000000000000000

Iv:00000000000000000000000000000000

Message:6a84867cd77e12ad07ea1be895c53fa3

Expected:732281c0a0aab8f7a54a0c67a0c45ecf

```
<terminated> DESTest [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (4/02/2017 4:23:26 p. m.)
Key      : 00000000000000000000000000000000
Message  : 6A84867CD77E12AD07EA1BE895C53FA3
Cipher   : 732281C0A0AAB8F7A54A0C67A0C45ECFCF52019292387D1B2C9D44C45D418A48
Expected: 732281C0A0AAB8F7A54A0C67A0C45ECF
```

2)

Key= 4278b840fb44aaa757c1bf04acbe1a3e

Iv: 57f02a5c5339daeb0a2908a06ac6393f

Message:
3c888bbbb1a8eb9f3e9b87acaad986c466e2f7071c83083b8a557971918850e5

Expected:
479c89ec14bc98994e62b2c705b5014e175bd7832e7e60a1e92aac568a861eb7

```
<terminated> DESTest [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (4/02/2017 4:41:04 p. m.)
Key      : 4278B840FB44AAA757C1BF04ACBE1A3E
Message  : 3C888BBBB1A8EB9F3E9B87ACAAD986C466E2F7071C83083B8A557971918850E5
Cipher   : 479C89EC14BC98994E62B2C705B5014E175BD7832E7E60A1E92AAC568A861EB7FC2DC2F4A527CE39F79C56B31432C779
Expected: 479C89EC14BC98994E62B2C705B5014E175BD7832E7E60A1E92AAC568A861EB7
```

Nota: Profe estos fueron los 2 únicos vectores que encontré porque en la página que nos diste ya no aparecen.

Fuentes:

https://bit502.wordpress.com/2014/06/27/codigo-java-encriptar-y-desencriptar-texto-usando-el-algoritmo-aes-con-cifrado-por-bloques-cbc-de-128-bits/