PSP\Tarea4PSP\src\tarea4pspsolucion\GestionCifrados.java

```java
package tarea4pspsolucion;

import java.io.File;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.nio.file.StandardOpenOption;
import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;

public class GestionCifrados {

    private static final File BD_USUARIOS = new File("usuarios.txt");
    private static final String CLAVE = "1234567890123456";
    private static final String ALGORITMO = "AES";

    private static byte[] cifrar(String datos) throws Exception {
        Cipher cipher = Cipher.getInstance(ALGORITMO);
        SecretKeySpec keySpec = new SecretKeySpec(CLAVE.getBytes(), ALGORITMO);
        cipher.init(Cipher.ENCRYPT_MODE, keySpec);
        return cipher.doFinal(datos.getBytes());
    }

    private static String descifrar(byte[] datosCifrados) throws Exception {
        Cipher cipher = Cipher.getInstance(ALGORITMO);
        SecretKeySpec keySpec = new SecretKeySpec(CLAVE.getBytes(), ALGORITMO);
        cipher.init(Cipher.DECRYPT_MODE, keySpec);
        return new String(cipher.doFinal(datosCifrados));
    }

    public static String obtenerUsuarios() throws IOException, Exception {
        try {
            Path path = Paths.get(BD_USUARIOS.getPath());
```

```
            String usuarios = "";
            // Verificamos si el archivo existe y tiene contenido
            if (Files.exists(path) && Files.size(path) > 0) {
                byte[] contenidoCifrado = Files.readAllBytes(path);
                usuarios = descifrar(contenidoCifrado);
            }

            return usuarios;
        } catch (IOException e) {
            System.err.println("No se pudo leer el archivo: " + e.getMessage());
            // devuelve un array vacio si no es posible leer el archivo
            return new String();
        }

    }

    public static boolean escribirUsuarios(String listado) throws Exception {
        try {
            Path path = Paths.get(BD_USUARIOS.getPath());
            // Escribe el contenido cifrado en el archivo.
            // Si el archivo no existe, lo crea (StandardOpenOption.CREATE).
            // Si el archivo ya existe, agrega el nuevo contenido al final sin borrar lo
            // anterior (StandardOpenOption.APPEND).
            Files.write(path, cifrar(listado), StandardOpenOption.CREATE);
            return true;
        } catch (IOException e) {
            System.err.println("No se pudo leer el archivo: " + e.getMessage());
            // devuelve un array vacio si no es posible leer el archivo
            return false;
        }

    }

}
```