LANZA PROCESOS

```java
import java.io.IOException;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 *
 * @author David Jiménez Riscardo
 */
public class Lanzador {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {

        Process proceso1, proceso2;

        proceso1 = lanzaProceso("javac Palindromo.java");

        try {
            proceso1.waitFor();
        } catch (InterruptedException ex) {
            Logger.getLogger(Lanzador.class.getName()).log(Level.SEVERE, null, ex);
        }

        proceso2 = lanzaProceso("cmd /c start cmd /k java Palindromo");
    }

    /** *
     * Función para lanzar un proceso
     * @param cadena Comando a ejecurar
     * @return Proceso
     */
    public static Process lanzaProceso(String cadena){

        Process proceso = null;

        //Lanzamos el proceso
        try {
            proceso = Runtime.getRuntime().exec(cadena);
        } catch (IOException ex) {
            System.out.println("Error de entrada/salida. " + ex.getMessage());
        } catch (SecurityException ex){
            System.out.println("Error de seguridad. " + ex.getMessage());
        } catch (Exception ex){
            System.out.println("Error. " + ex.getMessage());
        }
        return proceso;
    }

}
```

PALINDROMO

```java
package ej5_mandarcompilaryejecutarclase;

import java.io.*;
import java.util.Scanner;

public class Palindromo {

   public static void main(String[] args) {
      // ****************************************
      // ****************************************
      //
      // Leemos una cadena desde el teclado o
      // desde las tuberias
      //
      // ****************************************
      // ****************************************
      InputStreamReader streamTuberia = null;
      BufferedReader tuberia = null;
      InputStream streamTeclado = null;
      Scanner teclado = null;
      String palindromo = "";
      try {
         // Prepara el buffer para leer de tubería
         streamTuberia = new InputStreamReader(System.in);
         tuberia = new BufferedReader(streamTuberia);
         // Prepara el buffer para leer de teclado
         streamTeclado = System.in;
         teclado = new Scanner(streamTeclado);

         boolean preparado = tuberia.ready();
         if (preparado) {
            // Hay datos en la tubería y se leen
            palindromo = tuberia.readLine();
         } else {
            // No hay datos en la tubería y se solicitan por teclado
            palindromo = teclado.nextLine();
         }
         if (palindromo == null || palindromo.isEmpty()) {
            System.out.println("\nLa cadena que se ha recibido esta nula o vacia\n.");
         } else if (esPalindromo(palindromo)) {
            System.out.println("\nLa palabra " + palindromo + " si es un palindromo.\n");
         } else {
            System.out.println("\nLa palabra " + palindromo + " no es un palindromo.\n");
         }
      } catch (IOException ex) {
         System.err.println("Error debido a la E/S general.");
      } catch (Exception e) {
         System.err.println("Ha ocurrido un error: " + e.getMessage());
      } finally {
         // Se cierran las conexiones
         try {
            if (streamTuberia != null) {
               streamTuberia.close();
```

```
            }
            if (tuberia != null) {
               tuberia.close();
            }
            if (streamTeclado != null) {
               streamTeclado.close();
            }
            if (teclado != null) {
               teclado.close();
            }
         } catch (IOException ex) {
            System.err.println("Error debido a la E/S en el cierre de las conexiones.");
         }
      }

   }

   private static boolean esPalindromo(String palindromo) {
      // La cadena la convertimos a minúsculas y le quitamos
      // los espacios, puntos y comas. También quitamos las
      // tildes.
      palindromo = palindromo.toLowerCase().replace("á", "a").replace("é", "e").replace("í", "i").replace("ó", "o")
            .replace("ú", "u").replace(" ", "").replace(".", "").replace(",", "");

      // Invertimos la cadena y si es igual que la original entonces
      // es un palíndromo
      String invertida = new StringBuilder(palindromo).reverse().toString();

      return invertida.equals(palindromo);
   }

}
```

MAIN DEL PALINDROMO

```
package ej5_mandarcompilaryejecutarclase;

import java.io.*;

public class Main {

   public static void main(String[] args) {

      // ************************************
      // ************************************
      //
      // Compilamos el programa Palindromo
      //
      // ************************************
      // ************************************
      compilarClase_ProcessBuilder();

      // ************************************
      // ************************************
      //
      // Ejecutamos el programa Palindromo
      //
      // ************************************
      // ************************************
      ejecutarClase_ProcessBuilder();
   }

   public static void compilarClase_ProcessBuilder() {
      try {
         Process proceso = new ProcessBuilder("CMD", "/C",
               "JAVAC src\\ej5_mandarcompilaryejecutarclase\\Palindromo.java").start();
         int estado = proceso.waitFor();
         if (estado == 0) {
            // OK
            System.out.println("Clase compilada.");
         } else {
            // KO
            System.out.println("Error al compilar la clase");
         }
      } catch (SecurityException eSec) {
         System.out.println("Error de seguridad. Sin permiso para ejecutar el proceso");
      } catch (Exception e) {
         System.err.println("Ha ocurrido un error: " + e.getMessage());
      }
   }

   private static void ejecutarClase_ProcessBuilder() {
      InputStream is = null;
      try {
         // WAAKKAAW es un palindromo
         Process proceso = new ProcessBuilder("CMD", "/C",
               "ECHO WAAKKAAW | java src\\ej5_mandarcompilaryejecutarclase\\Palindromo.java").start();
```

```
            int estado = proceso.waitFor();
            if (estado == 0) {
                // OK
                System.out.println("\nClase ejecutada.");
                is = proceso.getInputStream();
                int c;
                while ((c = is.read()) != -1) {
                    System.out.print((char) c);
                }
            } else {
                // KO
                System.out.println("Error al ejecutar la clase\n");
            }
        } catch (SecurityException eSec) {
            System.out.println("Error de seguridad. Sin permiso para ejecutar el proceso");
        } catch (Exception e) {
            System.err.println("Ha ocurrido un error: " + e.getMessage());
        } finally {
            // Se cierran las conexiones
            try {
                if (is != null) {
                    is.close();
                }
            } catch (IOException ex) {
                System.err.println(
                        "Error debido a la E/S en los cierre de los flujos.");
            }
        }
    }

}
```