Produce

```
package aleatorios;
import java.util.Random; //Paquete para la generación de números aleatorios

/** Aplicación que genera 40 números aleatorios entre los valores 0 y 100
 * @author Marga Nieto
 */
public class Main {

    /** Método main de la aplicación, punto de entrada al programa.
     * @param args Los comandos recibidos por la línea de ejecución
     * no son tenidos en cuenta.
     */
    public static void main(String[] args) {
         // Lo único que hacemos es escribir los 40 números aleatorios en
        // la salida estándar del proceso
        Random v = new Random(); //Instanciamos el objeto para generar aleatorios
        for (int i=0; i<40; i++) //Bucle para generar 40 valores
            /*//Podemos generar números aleatorios utilizando Math.Random()
             *  System.out.println(Math.floor(Math.random()*100));
             */
            System.out.println(v.nextInt(101)); //valores enteros entre 0 y (101-1)
            //Escribimos un número por línea, para que se incluya intro
    }

}
```

Ordena

```
package ordenarnumeros;

import java.io.InputStreamReader; //Importamos los paquetes necesarios
import java.io.BufferedReader;    //para manejar la entrada/salida del proceso
import java.io.IOException;

import java.util.LinkedList; //Para trabajar con listas
import java.util.Collections; //Para ordenar listas

/** Aplicación que devolverá ordenados los números que recibe en su entrada estándar.
 * El número de elemntos en el conjunto de números puede ser cualquiera (no fijo).
 * @author Marga Nieto
 */
public class Main {

    /** Método main de la aplicación. Punto de entrada de ejecución.
     * Devolverá ordenados los números que recibe en su entrada estándar.
     * El número de elemntos en el conjunto de números puede ser cualquiera (no fijo).
     * @param args Los argumentos de la línea de comandos no se tendrán en cuenta
     */
    public static void main(String[] args) {
        // Vamos a leer de la entrada estándar del proceso y escribir
```

```
    // los valores recibidos en la salida estándar del proceso, ordenados.

    InputStreamReader isr = new InputStreamReader(System.in);
    BufferedReader bf = new BufferedReader (isr);
    // Obtenemos el stream de lectura de la entrada estándar
    // utilizamos un lector con Buffered, para no perder ningún dato
    String lineaTeclado = null; //Variable para ir leyendo lo leído de teclado
    LinkedList lista = new LinkedList();

    try{
        System.out.println("Proceso ordenaNúmeros");
        //Mostramos que el proceso que está escribiendo es el que está
        //leyendo los datos.
        while ((lineaTeclado = bf.readLine())!= null){ //Mientras haya datos disponibles
            if (isNumeric(lineaTeclado)) { //Comprobamos que el valor leído es numérico
                int v = Integer.parseInt(lineaTeclado);
                //Estamos seguros de que no
                lista.add(v);
            }
        }
        //Ordenamos la lista de valores
        ordena(lista);
        //Mostramos los datos ordenados
        System.out.println("Los enteros ordenados son: ");
        for(Object elemento: lista) //Recorremos todos los elementos de la lista
            System.out.println(elemento.toString()); //escribimos cada uno de ellos
        System.out.println("Número de elementos leídos: " + lista.size());
    }catch(IOException ex){
        System.err.println("Se ha producido un error de E/S. Su descripción es: ");
        System.err.println(ex.toString());
    }catch(Exception ex){
        System.err.println("Se ha producido un error. Su descripción es: ");
        System.err.println(ex.toString());
    }
}
/**
 * Método que comprueba si una cadena representa un dato numérico entero
 * @param dato String con el valor que se quiere comprobar.
 * @return true en el caso de que el contenido sea numérico entero y false en caso contrario
 */
private static boolean isNumeric(String dato){
    try{
        Integer.parseInt(dato); //Intentamos hacer la conversión
        return true; //Si no se ha producido excepción, el valor es numérico
    }catch(Exception e){
        return false; //El dato no se puede convertir en número
    }
}

/**
 * Método que ordena una lista
 * @param lista Lista de tipo LinkedList que va a ser ordenada
```

```
     */
    private static void ordena(LinkedList lista) {
        Collections.sort(lista);   //Ordenamos los elementos, en orden ascenente
                                   //según su orden natural
        //Podemos utilizar directamente Collections.sort(lista);
        //Porque los elementos que contiene son enteros y son comparables
    }
}
```