Teoría de la Información y Codificación

Práctica 0: Toma de Contacto con la Asignatura

José A. Montenegro Montes

26 de septiembre de 2014

1. Enunciado

En esta tarea vamos a codificar cinco tipos de algoritmos de detección de errores: NIF, ISBN, ISBN13, Código de producto (UPC) y Código Control número de cuenta bancaria (opcional).

Los algoritmos para crear los códigos de control mencionados son proporcionados en las transparencias de la asignatura y adjuntos en la última sección de las prácticas, menos el Código Control número de cuenta bancaria.

La práctica consta de la creación de una clase abstracta de codificación, y la implementación de los cuatros algoritmos solicitados (y uno opcional).

Los primeros tres métodos están implementados en el código aportado. El alumno debe revisar que la implementación cumple con la definición de los algoritmos e implementar el último método($corregirDatos$).

- Creación de un código control para un mensaje dado ($generarCodigoControl$).
- Verificación mensaje es correcto según el código control ($verificar$).
- Corrección del código control erróneo de un mensaje dado ($corregirControl$).
- Corrección del bit erróneo de un mensaje (un único error posible), dado un código control correcto ($corregirDatos$).

En el caso concreto del ISBN-10 sabemos que el código de control es mod 11, que ocurre si el código de control que obtenemos es 10. ¿Cómo lo incluimos en el código?.

2. Conclusiones

El objetivo de esta práctica es tomar contacto con este tipo de algoritmos y observar las ventajas y desventajas que nos presentan estos algoritmos.

Se abre el debate si son códigos correctores o simplemente códigos detectores de errores.

A lo largo del curso veremos una variedad de algoritmos tanto para la detección como para la corrección de errores, cuya calidad puede medirse en los número de bits que detecta / corrige.

3. Código

Clase Codificación

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

package practica0;

public abstract class Codificacion {

    /**
     * Comprueba si el codigo es valido.
     *
     * @param codigo    Codigo a comprobar.
     * @return          Verdadero si es Codigo es valido, falso en otro caso.
     */
    public abstract boolean verificar(String codigo);

    /**
     * Devuelve el Codigo valido a partir del Codigo incorrecto que recibe como parametro.
     *
     * @param codigo    Codigo incorrecto.
     * @return          Codigo corregido.
     */
    public String corregirControl(String codigo){
         if (verificar(codigo)) return codigo;
         else return generarCodigoControl(codigo);

    };

    /****************
     * Genera el Codigo control a partir de los datos
     * @param codigo
     * @return
     */
```

```java
public abstract String generarCodigoControl(String codigo);

    /*************
     * Supuestamente el Codigo de control esta bien buscar donde es el error.
     * @param codigo
     * @return
     */
    public abstract String[] corregirDatos(String codigo);
}
```

Clase ISBN10 Java.

```java
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

package practica0;

/**
 *
 * @author
 */
public class ISBN extends Codificacion {

    private static final int MODULO = 11;

/*************
 *
 * @param codigo
 * @return
 */
    public boolean verificar(String codigo) {
        codigo = codigo.replaceAll("-", "");
        int resultado = 0;
        try {
            for (int i = 0; i < codigo.length(); i++) {
                resultado += Integer.parseInt(codigo.substring(i, i+1))*(i+1);
            }
        } catch (NumberFormatException e) {
            return false;
        }
        return resultado % MODULO == 0;
    }
```

```
33
34 /***************
35  *
36  * @param codigo
37  * @return
38  */
39     @Override
40     public String generarCodigoControl(String codigo) {
41
42      String retorno;
43      int resultado = 0;
44      codigo = codigo.replaceAll("-", "");
45      retorno=codigo;
46
47      if (codigo.length()==9){
48              try {
49              for (int i = 0; i < codigo.length(); i++) {
50                  resultado += (Integer.parseInt(codigo.substring(i, i+1))*(i+1));
51              }
52              retorno = codigo+(resultado% MODULO);
53          } catch (NumberFormatException e) {
54              return null;
55          }
56      }
57
58         return retorno;
59     }
60
61
62
63     @Override
64     public String[] corregirDatos(String codigo) {
65         throw new UnsupportedOperationException("Not supported yet.");
66     }
67
68
69 }
```

Clase ISBN13 Java.

```
1 /*
2  * To change this template, choose Tools | Templates
3  * and open the template in the editor.
4  */
```

```java

package practica0;

/**
 *
 * @author monte
 */
public class ISBN13 extends Codificacion{

    private static final int MODULO = 10;
/***************
 *
 */
     public boolean verificar(String codigo) {
       codigo = codigo.replaceAll("-", "");
       int resultado = 0;
       try {
           for (int i = 0; i < codigo.length(); i++) {
              if (i%2==0)
               resultado += (Integer.parseInt(codigo.substring(i, i+1))*1);
              else
               resultado += (Integer.parseInt(codigo.substring(i, i+1))*3);
           }
       } catch (NumberFormatException e) {
           return false;
       }
       return resultado % MODULO == 0;
    }

/************
 *
 */
    public String generarCodigoControl(String codigo) {

    String retorno;
      codigo = codigo.replaceAll("-", "");
    retorno=codigo;
     int resultado = 0;
     if (codigo.length()==12){
       try {
           for (int i = 0; i < codigo.length(); i++) {
              if (i%2==0)
               resultado += (Integer.parseInt(codigo.substring(i, i+1))*1);
              else
               resultado += (Integer.parseInt(codigo.substring(i, i+1))*3);
           }
```

```
51          } catch (NumberFormatException e) {
52              return null;
53          }
54
55          resultado = resultado % MODULO;
56          resultado = (MODULO -resultado)%MODULO;
57          retorno= codigo+resultado;
58       }
59          return retorno;
60      }
61
62  /******************
63   *
64   */
65      public String[] corregirDatos(String codigo) {
66          throw new UnsupportedOperationException("Not supported yet.");
67      }
68
69  }
```

Clase NIF Java.

```
1  /*
2  * To change this template, choose Tools | Templates
3  * and open the template in the editor.
4  */
5
6  package practica0;
7
8  /**
9  *
10 * @author
11 */
12 public class NIF extends Codificacion {
13
14
15     private static final String NIF_TABLA = "TRWAGMYFPDXBNJZSQVHLCKE";
16
17
18
19
20     @Override
21     public boolean verificar(String nif) {
22        int dni = Integer.parseInt(nif.substring(0, nif.length()-1));
```

```
        try {
            return NIF_TABLA.charAt(dni % NIF_TABLA.length()) == nif.charAt(nif.length()-1);
        } catch (NumberFormatException e) {
            return false;
        }
    }


    /*************
     *
     * @param codigo
     * @return
     */
    public String generarCodigoControl(String codigo) {

        if (codigo!=null){
         int dni = Integer.parseInt(codigo, 10);
         return codigo+ NIF_TABLA.charAt(dni % NIF_TABLA.length());
        }
        else return null;
    }


    public String [] corregirDatos(String codigo) {
        throw new UnsupportedOperationException("Not supported yet.");
    }
}
```

Clase UPC Java.

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

package practica0;

/**
 *
 * @author
 */
public class UPC extends Codificacion {

    private static final int MULTIPLO = 10;
```

```
15
16 /**********************
17  *
18  * @param codigo
19  * @return
20  */
21     public boolean verificar(String codigo) {
22         int resultado = 0;
23         int v;
24         try {
25             for (int i = 0; i < codigo.length(); i++) {
26                 v = Integer.parseInt(codigo.substring(i, i+1));
27                 resultado += i%2 == 0 ? v : v*3;
28             }
29         } catch (NumberFormatException e) {
30             return false;
31         }
32         return resultado % MULTIPLO == 0;
33     }
34
35
36     /***************
37      *
38      * @param codigo
39      * @return
40      */
41     public String generarCodigoControl(String codigo) {
42         throw new UnsupportedOperationException("Not supported yet.");
43     }
44
45
46     public String[] corregirDatos(String codigo) {
47         throw new UnsupportedOperationException("Not supported yet.");
48     }
49
50 }
```

Clase Main Java.

```
1 /*
2  * To change this template, choose Tools | Templates
3  * and open the template in the editor.
4  */
5
```

```java
package practica0;

import practica0.Codificacion;
import practica0.ISBN;
import practica0.NIF;
import practica0.UPC;

/**
 *
 * @author Monte
 */
public class Main {

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


        String codigoISBN = "157955008"; //  1579550088 ;
        String codigoISBN13 = "978-1-84800-272-"; //978-1-84800-272-2;
        String codigoNIF = "55555555"; // 55555555K
        String codigoUPC = "9780444485199";


        /**********************
         *
         */
        Codificacion   cod = new ISBN();

        codigoISBN=cod.generarCodigoControl(codigoISBN);
        System.out.print("ISBN: "+codigoISBN+ ": ");
        System.out.println(cod.verificar(codigoISBN) ? "valido" : "No valido");


        /*********************************
         *
         */
        cod = new ISBN13();
        codigoISBN13=cod.generarCodigoControl(codigoISBN13);
        System.out.print("ISBN13: "+codigoISBN13+ ": ");
        System.out.println(cod.verificar(codigoISBN13)  ? "valido" : "No valido");

        /*********************************
         *
         */
```

```
52          cod = new NIF();
53          codigoNIF = cod.generarCodigoControl(codigoNIF);
54          System.out.print("NIF: "+codigoNIF+ ": ");
55          System.out.println(cod.verificar(codigoNIF)  ? "valido" : "No valido");
56
57          /*****************
58           *
59           */
60          cod = new UPC();
61          System.out.print("UPC: "+codigoUPC+ ": ");
62          System.out.println(cod.verificar(codigoUPC) ? "valido" : "No valido");
63
64      }
65
66  }
```