

# CÓDIGOS DE BARRAS

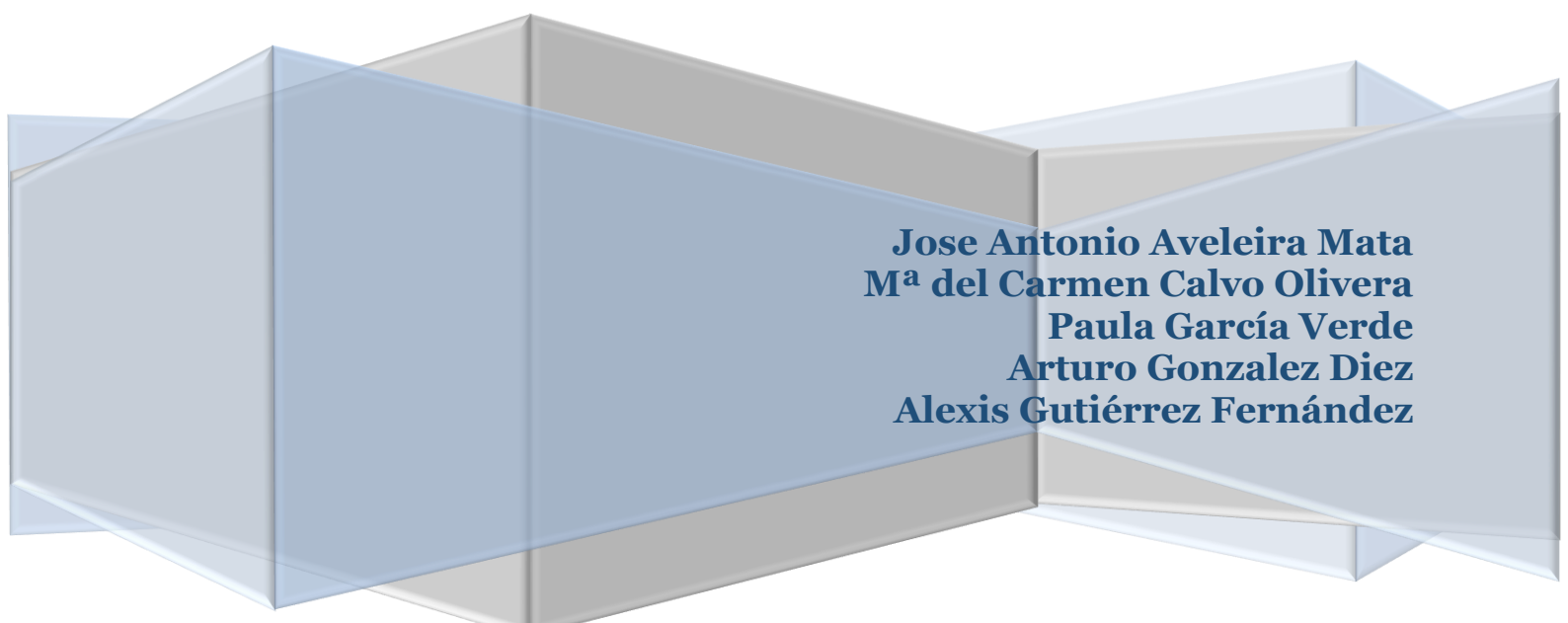
**Seguridad Informática**

**Profesores:**

**Andrés Sáez Schwedt**

**Adriana Suárez Corona**

**16/05/15**



**Jose Antonio Aveleira Mata  
M<sup>a</sup> del Carmen Calvo Olivera  
Paula García Verde  
Arturo Gonzalez Diez  
Alexis Gutiérrez Fernández**

## ÍNDICE

<b>1. Historia</b>	<b>2</b>
<b>2. ¿Qué es un código de barras?</b>	<b>2</b>
<b>3. ¿Cómo se codifica?</b>	<b>2</b>
<b>4. Tipos de códigos de barras</b>	<b>3</b>
4.1. Code128	3
4.2. Code39	3
4.3. Code93	3
4.4. EAN	4
4.5. Codabar	4
4.6. UCC	4
<b>5. EAN13</b>	<b>4</b>
<b>6. ¿Cómo se lee por el lector?</b>	<b>5</b>
<b>7. Probabilidad de borrón aleatorio</b>	<b>6</b>
<b>8. Casos curiosos</b>	<b>7</b>
8.1. Detenido un hombre por falsificar códigos de barras	
8.2. Dos informáticos alteraban los códigos de barras para estafar a los hipermercados	
8.3. Fabricante de leche tacha el número 666 en el código de barra de sus envases	
<b>9. Código explicado</b>	<b>11</b>
9.1. Validar código	
9.2. Calcular dígito de control	
9.3. Para recuperar dígito	
9.4. Para Borrar Random	
<b>10. Interfaz</b>	<b>17</b>
<b>11. Bibliografía</b>	<b>21</b>

# CÓDIGOS DE BARRAS

## 1. Historia

La primera patente de código de barras fue registrada en octubre de 1952 (US Patent #2,612,994) por los inventores Joseph Woodland, Jordin Johanson y Bernard Silver en Estados Unidos. La implementación fue posible gracias al trabajo de los ingenieros Raymond Alexander y Frank Stietz. El resultado de su trabajo fue un método para identificar los vagones del ferrocarril utilizando un sistema automático. Sin embargo, no fue hasta 1966 que el código de barras comenzó a utilizarse comercialmente y no tuvo un éxito comercial hasta 1980.

## 2. ¿Qué es un código de barras?

*El Código de Barras es una disposición en paralelo de barras y espacios que contienen información codificada en las barras y espacios del símbolo.*

El código de barras almacena información, almacena datos que pueden ser reunidos en él de manera rápida y con una gran precisión. Los códigos de barras representan un método simple y fácil para codificación de información de texto que puede ser leída por dispositivos ópticos, los cuales envían dicha información a una computadora como si la información hubiese sido tecleada.

Los códigos de barras se pueden imaginar como si fueran la versión impresa del código Morse, con barras angostas (y espacios) representando puntos, y barras anchas que representan rayas.

## 3. ¿Cómo se codifica?

Para codificar datos dentro de un símbolo\* impreso, se usa una barra predefinida y patrones de espacios o simbología\*\*

*\*Un símbolo de código de barras es la **visualización física**, es la impresión de un código de barras.*

*\*\*Una simbología es la forma en que se codifica la información en las barras y espacios del símbolo de código de barras.*

El código de barras representa la clave para acceder a un registro de alguna base de datos en donde realmente reside la información, o sea, los símbolos no contienen información del producto o artículo, no contienen el precio del producto, sino contiene una clave que identifica al producto.

## **4. Tipos de códigos de barras**

Los códigos de barra más utilizados a nivel mundial para identificación de los productos son Code128 , Code39 , Code93, EAN, UPC y Codabar, y cada uno de ellos se diferencian por ciertas características que responden a las necesidades precisas de los productos que serán etiquetados.

### **4.1. Code128**

Suele ser el más utilizado en el ambiente de la logística para etiquetar los productos, como así también en billetes y postales.

Este código trabaja con una codificación de caracteres alfanuméricos, hasta 106 caracteres diferentes, y posee una longitud variable.

### **4.2. Code39**

Ha sido uno de los primeros códigos que incluyeron una codificación alfanumérica. El mismo permite codificar hasta un número total de 43 caracteres, y posee una longitud variable.

### **4.3. Code93**

Fue desarrollado para expandir las posibilidades del Code39, permitiendo codificar hasta un total de 47 caracteres alfanuméricos, con una longitud variable.

#### **4.4 EAN**

Es el utilizado en todos los productos que se comercializan en el mercado europeo, su nombre surge de las siglas de European Article Numbering.

Se trata de un código que sólo permite una codificación numérica, y su longitud es limitada, ya que sólo ofrece la posibilidad de codificar entre 8 a 13 dígitos en sus diversas variantes.

#### **4.5 Codabar**

Se trata de un código utilizado comúnmente en bibliotecas, bancos de sangre, envíos, encomiendas y demás, y consiste en un código numérico que sólo permite codificar hasta 16 caracteres con una longitud variable.

#### **4.6 UCC**

Siglas de su nombre en inglés Uniform Code Council, es utilizado en todos los productos provenientes de los Estados Unidos, y se caracteriza por ser un código del tipo numérico, que posee una longitud limitada de entre 7 a 12 dígitos, dependiendo de su versión.

### **5. EAN13**

EAN (European Article Number o International Article Number) es un sistema de códigos de barras adoptado por la gran parte de países y empresas del mundo. En el año 2005, la asociación EAN se ha fusionado con la UCC (Uniform Code Council) para formar una nueva y única organización mundial identificada como GS1, con sede en Bélgica. El código EAN más usual es EAN13, constituido por 13 dígitos y con una estructura dividida en cuatro partes:

- Los primeros dígitos identifican a través de qué **Organización Nacional** se ha adscrito una empresa al Sistema EAN. Por ejemplo, en España se encarga de ello Aecoc y su código es el 84.

- Referencia del ítem, compuesto de:

- **Código de empresa:** es un número compuesto por entre 5 y 8 dígitos, dependiendo de las necesidades de la empresa, que identifica al propietario de la marca.
- **Código de producto:** completa los 12 primeros dígitos.
- **Dígito de control:** consta de un solo dígito y sirve para verificar que el código leído es correcto. Para calcularlo se suman los dígitos de las posiciones impares, se multiplica por 3, se le suman los dígitos de las posiciones pares y a este resultado se le resta el siguiente múltiplo de 10. El resultado final ha de coincidir con el dígito de control.

Por lo tanto el código de barras EAN 13 puede ser muy variable dependiendo de la empresa que lo utilice. Un ejemplo podría ser:



- [84] Los dos primeros dígitos identifican que la empresa se ha adscrito a Aecoc como Organización Nacional registradora de códigos de barras EAN.
- [25623] Los 5 dígitos siguientes indican el código de empresa asignado por la Organización Nacional Aecoc.
- [50012] Los 5 dígitos siguientes identifican el producto dentro de la empresa, este código será asignado por la empresa a su libre albedrío.
- [6] El último dígito es calculado, es el dígito de control.

## 6. ¿Cómo se lee por el lector?

El lector de código de barras decodifica la información a través de la digitalización proveniente de una fuente de luz reflejada en el código y luego se envía la información a una computadora como si la información hubiese sido ingresada por teclado. Este tipo de lector lo utilizaremos nosotros.

El procedimiento: el símbolo de código de barras es iluminado por una fuente de luz visible o infrarrojo, las barras oscuras absorben la luz y los espacios las reflejan nuevamente hacia un escáner.

El escáner transforma las fluctuaciones de luz en impulsos eléctricos los cuales copian las barras y el modelo de espacio en el código de barras. Un decodificador usa algoritmos matemáticos para traducir los impulsos eléctricos en un código binario y transmite el mensaje decodificado a un terminal manual, PC, o sistema centralizado de computación.

El decodificador puede estar integrado al escáner o ser externo al mismo. Los escáneres usan diodos emisores de luz visible o infrarroja (LED), láser de Helio-Neón o diodos láser de estado sólido (visibles o infrarrojos) con el fin de leer el símbolo.

Algunos de ellos necesitan estar en contacto con el símbolo, otros leen desde distancias de hasta varios pies. Algunos son estacionarios, otros portátiles como los escáneres manuales.

**Nosotros utilizaremos la entrada de datos por teclado,** (portátiles o montados) se conectan a una computadora y transmiten los datos al mismo tiempo que el código es leído.

## 7. PROBABILIDAD DE BORRÓN ALEATORIO

Se quiere programar un simulador aleatorio de errores de tipo borrón, en el cual haya una pequeña probabilidad de que el número de borrones generado sea mayor que la capacidad detectora del código.

Sea  $p$  = probabilidad de que se produzca un borrón en 1 dígito.

Sea  $X$  = número de borrones en el código de barras.

$X$  es una variable binomial con  $n = 13$  y  $p = ?$  ajustable por el usuario.

$p=0.05$  puede ser aceptable ya que, haciendo pruebas con R:

```
> dbinom(0,13,0.05)
```

```
[1] 0.5133421    <<<<< probabilidad de 0 errores
```

```
> dbinom(1,13,0.05)
```

```
[1] 0.3512341    <<<<< probabilidad de 1 error
```

Un 51% de las veces hay 0 errores, 35% de las veces hay un error, y el restante 14% de las veces hay más de 1 error. Variando el valor de  $p$  cambian las probabilidades anteriores, pero como punto de partida  $p = 0.05$  está bien. Se somete cada una de las 13 posiciones del código de barras a un test aleatorio, que le produzca un error con probabilidad  $p$ . Si el número sale menor que  $p$ , se genera un borrón, en caso contrario no. Y eso se hace para cada posición.

## 8. Casos curiosos

### 8.1. Detenido un hombre por falsificar códigos de barras

Realizaba pequeños fraudes con los que sacaba un beneficio de 200 a 300 euros, sobre todo con bebidas de distinto tipo.

Su forma de actuar era aprovechar momentos de una gran actividad comercial de los establecimientos para colocar en el código de barra de los paquetes otro que correspondía a una sola unidad del producto. De esa forma, abonaba un porcentaje mucho menor. Por ejemplo, en vez de pagar 10 o 12 euros por un paquete de cuatro bebidas energéticas, pagaba únicamente 2 o 3 euros. Para que el estafador consiguiese su propósito, el empleado debía estar despistado o no recordar el precio total del paquete. En los casos en los que el engaño era advertido por el empleado, el detenido simulaba tener prisa y desaparecía del establecimiento con rapidez.

Los códigos de barras proporcionan información sobre el producto y su precio, y su falsificación consiste en modificar su apariencia para que el lector electrónico



interprete otros datos. De esta forma, el empleado cobraría una cantidad mucho menor sin darse cuenta, lo que supondría una pérdida para la empresa.

## **8.2. Dos informáticos alteraban los códigos de barras para estafar a los hipermercados**

Dos jóvenes informáticos acusados de estafar en menos de 24 horas más de un millón y medio de pesetas en grandes almacenes mediante la manipulación de los códigos de barras.

Compraron en el centro comercial Hipercor de San José de Valderas siete juegos de altavoces de la marca Pioneer, por los que pagaron 2.020 pesetas cuando eran modelos valorados en 21.000, 15.100 y 30.580 pesetas.

Entraban en el centro comercial y buscaban el modelo más barato de la marca que les interesaba, anotaban la numeración del código de barras y salían a su furgoneta -dotada de ordenador, impresora y un programa informático especialmente diseñado para ello-. Allí elaboraban etiquetas adhesivas con el código de barras del producto más barato. Una vez de vuelta al establecimiento, pegaban las etiquetas falsas a los objetos deseados (siempre modelos de esa misma marca, pero mucho más valiosos). En la caja, el lector electrónico marcaba el precio esperado.

### 8.3. Fabricante de leche tacha el número 666 en el código de barra de sus envases



El código de barras queda tachado por una cruz roja y su código de identificación Bidi o QR a la derecha. (Russkoe)

El fabricante de leche Russkoe Moloko habló muy en serio cuando dijo que tachó “el nombre de la bestia”, representado por el número “666”, en los códigos de barras que se ven obligados a poner en los envases, y lo hizo con lo que llamó ser la “cruz del apóstol San Andrés”.

"Durante los últimos cinco años en todos los paquetes de productos bajo la marca "Russkoe Moloko" el código de barras está tachado con una cruz roja. Muchos de nuestros clientes se preguntan: ¿por qué? Recibimos suficientes preguntas, así que nos decidimos a explicar la razón", escribió la compañía de Vasili Boiko, quien también es el presidente del comité organizador del Movimiento Popular Santa Rusia.

Él agregó que es sabido que en el código de barras de los productos está invisible al simple ojo humano el número 666, impreso como tres barras que exceden el largo de las restantes: una central, y las otras dos a los extremos de cada lado. “Es el nombre del Anticristo”, dice Russkoe Moloko. "Bien pudieron haber usado otros números, sin embargo los números “se eligieron conscientemente”.

“Los ancianos ortodoxos han advertido desde hace tiempo que justamente en la forma de un código de barras estaría presente “la marca” que sería impresa en la frente y las manos” de las personas, por lo que creen que no hay duda que se trataría de un “código maligno”.

“El contenido del número de la bestia en los códigos de barras no se puede dudar. Se realizaron tres estudios de códigos de barras antiguos por expertos de alto nivel en el campo de la programación”, escribió.

Russkoe Moloko citó al documento aportado por expertos monjes del Monasterio Gregoriou en Athos, presentado en el Centro de Informática Sr. George Sahinian de la Universidad de Bonn: “Codificación de European Article Numbering (EAN- 13 )” del 10 de octubre de 1997.

Además nombró al estudio de once científicos y de otros expertos en el campo de las tecnologías electrónicas realizado para la Comisión especial del Santo Sínodo, ingresado por el Sr. Stamatis Kulutru al Santo Sínodo de 1993- 43. “Hay muchos otros. También hay estudios nacionales”, agregó la compañía.

Solo las barras entre los números 666 corresponden a cifras del 0 al 9.

Russkoe Moloko argumentó que incluso las antiguas escrituras previnieron este hecho. Juan el Evangelista en el Apocalipsis dijo: "Si alguno adora a la bestia y a su imagen y recibe la marca en su frente o en su mano, él beberá del vino de la ira de Dios, vaciado puro en el cáliz de su ira, y será atormentado con fuego y azufre delante los santos ángeles y del Cordero; y sufrirá el humo de su tormento por los siglos de los siglos; y no tienen reposo ni de día ni de noche los que adoran a la bestia y a su imagen, y reciben la marca de su nombre (Rev.14 : 9- 11 )”.

En un comentario del Apocalipsis, además estaría escrito que “el nombre de la bestia se extendería por todas partes, en materia de compra y venta, a las personas que no lo llevasen por cuenta propia”, según Aref Cesarea (9-10 cc). A esto se agrega que “Y el mártir Ireneo escribe que el número del Anticristo 666 es un símbolo del "mal, apostasía, injusticia, maldad, falsas profecías y el engaño”.

Muchos sacerdotes, religiosos, teólogos se oponen a los códigos de barras. La Compañía JSC "Russkoe Moloko" dice estar obligada a ponerlos en el embalaje, porque “sin ello sólo podríamos vender nuestros productos en los mercados

agrícolas, pero no en las tiendas. Sin embargo, no descartamos la posibilidad en el futuro de otros procedimientos alternativos de venta, descartando el código de barras”.

“Queremos mostrar nuestra posición: estamos con nuestro Señor Jesucristo, no el Anticristo y sus sirvientes. Y los cristianos no experimentan miedo cuando están haciendo acciones con pensamientos del Señor”, escribió Boskoi en su Web.

Muchos cristianos ortodoxos que creen en esta profecía hacen lo que relata Russkoe Moloko: “tiran inmediatamente el código de barras del embalaje, o arrancan la parte del paquete con el código de barras”.

“En cualquier caso”, advierte la compañía, “no ponga uno de estos paquetes en su escritorio. Y si no se puede romper o tirar a la basura”, Russkoe Moloko recomienda “tacharlo con una cruz”.

“La mayoría de los productores y los consumidores no saben y no piensan en el código de barras. Pero es el pecado cometido por la ignorancia”, concluyó.

En 2007, durante un período de su agitada vida pública, el dueño de Russkoe Moloko Vasili Boiko fue detenido por una disputa de tierras arrendadas, pero fue liberado.

En cambio en los últimos años, Boiko es recordado por una serie de iniciativas en su empresa. Por ejemplo todos los empleados que tienen uniones informales deben someterse a una ceremonia de boda para ser admitidos. Además prohíbe a las mujeres tener abortos, según el medio Forbes de Rusia.

## 9. Código Explicado

### 9.1. Para validar el código:

Dos atributos, uno que tendrá el código para acceder a él desde toda la clase, y otro constante que será el módulo.

```
private String codigo;  
private int modulo = 10;
```

Un constructor de la clase, a la que es necesario pasar el código como parámetro, y que comprueba que sintácticamente sea correcto, es decir, que no tenga caracteres y que tenga longitud de 13.

Si todas las comprobaciones son correctas, comprueba que sea válido.

```

public ValidarCodigo(String codigo) {
    boolean valido= true;
    if(codigo.length() != 13){
        JOptionPane.showMessageDialog(null, "No es un codigo
valido por longitud");
        valido = false;

    }else{
        for(int i=0; i<codigo.length(); i++) {
            if (!Character.isDigit(codigo.charAt(i))){
                JOptionPane.showMessageDialog(null, "No
es un codigo valido por caracteres");
                valido=false;
                break;
            }
        }
        if(valido){
            this.codigo = codigo;
            comprobacion();
        }
    }
}

```

La función de comprobación suma los impares (que son los de la posición par ya que en java se empieza en 0) y los pares por otro lado.

Al multiplicar los pares por 3 y sumar a los impares, se realiza el modulo y se guarda en suma.

Si suma es == 0 es válido, y si no, el dígito de control no está bien calculado.

```

private void comprobacion() {
    int impares = 0, pares = 0;

    for(int i=0; i<this.codigo.length(); i++) {
        if(i%2==0)
            impares = impares +
Character.getNumericValue(codigo.charAt(i));
        else
            pares = pares +
Character.getNumericValue(codigo.charAt(i));
    }

    int suma = (impares + (3*pares))%modulo;

    if(suma == 0){

```

```

        JOptionPane.showMessageDialog(null, "El codigo es
válido");
    }else{
        JOptionPane.showMessageDialog(null, "El codigo es
invalido. El digito de control no es correcto");
    }
}
}

```

## 9.2. Calcular dígito de control:

```

public class CalculoDigitoDeControl {

    private String codigo;
    private String codigoConControl;
    private int digitoDeControl;
    private int mod = 10;

    public int getControl() {
        return digitoDeControl;
    }

    public CalculoDigitoDeControl(String c) {
        this.codigo = c;
    }

    public String mostrar() {
        char[] caracteres = codigo.toCharArray();
        int suma = 0, impares = 0, pares = 0;

        boolean valido= true;
        if(codigo.length() != 12){
            JOptionPane.showMessageDialog(null, "No es un código
válido por longitud");
            valido = false;
        }else{
            for(int i=0; i<codigo.length(); i++) {
                if (!Character.isDigit(codigo.charAt(i))){
                    JOptionPane.showMessageDialog(null, "No
es un código válido por caracteres");
                    valido=false;
                    break;
                }
            }
        }
    }
}

```

Lo más importante para el cálculo del dígito de control reside en esta última parte de la clase. Si el código resulta válido, es decir, tiene una longitud de 12 dígitos y ninguno de ellos es una letra, realiza las operaciones necesarias para el cálculo. Suma por separado los dígitos en posición impar y lo que se encuentran en una posición par.

Posteriormente en la variable *suma* se guarda la suma de pares e impares multiplicando por 3 el resultado de los pares, y todo ello módulo 10.

Despejamos para conseguir el dígito de control y se muestra un mensaje indicando que para el código introducido corresponde el dígito de control X.

```
        if(valido){
            for(int i=1; i<=caracteres.length; i++) {
                if(i%2==1)
                    impares +=
Character.getNumericValue(caracteres[i-1]);
                else if(i%2==0)
                    pares +=
Character.getNumericValue(caracteres[i-1]);
            }
            suma = (impares + (3*pares))%mod;
            digitoDeControl = (10-suma)%mod;
            codigoConControl = codigo + digitoDeControl;

            JOptionPane.showMessageDialog(null, "El dígito de
control para el código " + codigo + " es: " + digitoDeControl,
"DÍGITO DE CONTROL", JOptionPane.INFORMATION_MESSAGE);
            return String.valueOf(codigoConControl);
        }

        return codigo;
    }
}
```

### 9.3. Para recuperar digito:

Comenzamos llamando a la función recuperarDigito y le pasamos el código introducido con una incógnita, la cual transformamos a minúscula por comodidad.

```
public static String recuperarDigito(String cb){
    //Se introducirÃ¡ en el TextBox correspondiente una
cadena de 13 caracteres con una X.

    //Para evitar errores, hacemos que la X sea minÃºscula y
la busquemos en el String
    cb = cb.toUpperCase();
}
```

```

int pos = cb.indexOf("X");

//Este será el String de salida del método.
String cbString = cb;
int cuentaX = 0;

```

Transformamos el String obtenido anteriormente a un array en el que cada número ocupara una posición para acceder a ellos más fácilmente.

```

int[] arrayCb = new int[13];
for(int i=0;i< cbString.length();i++)
{
    if(cbString.charAt(i)=='X'){
        cuentaX ++;
        arrayCb[i]=0;
    }else{
        arrayCb[i]=Integer.parseInt(cbString.charAt(i)+"");
    }
}

```

Si después de colocar cada uno, detectamos que ha existido más de un carácter, anulamos, ya que solo tenemos capacidad correctora de 1 borrón.

Si por el contrario, no hay borrón, no hay nada que corregir.

```

if (cuentaX>=2){
    JOptionPane.showMessageDialog(null, "Hay dos o más
borrones. La capacidad correctora del programa es 1.");
    return cbString;
}else if (cuentaX==0){
    JOptionPane.showMessageDialog(null, "El código no
tiene borrones ");
    return cbString;
}

```

Cuando todo está bien, cambiamos la X por un 0 y lo transformamos en long para trabajar con el como Número.

```

else{//Si hay 1 X
    //Sustituimos la X por un 0
    cb = cb.replace("X","0");
}

```



```

//Parseamos el string para convertirlo a long.
String result = "";
long codigo= Long.parseLong(cb);

```

Comenzamos el cálculo, del estilo a los anteriores

```

//ALGORITMO del código de barras.
int suma = 0;
for (int i=0; i< arrayCb.length;i++){
    //a1+3a2 en este caso sera array[0]+3*array[1]
    etc.
    if (i==0 || i%2==0){
        suma += arrayCb[i];
    }else {
        suma += 3*arrayCb[i];
    }
}

//Creamos un long con el código de barras.
if (suma%10 == 0) {
    JOptionPane.showMessageDialog(null, "El dígito
    borrado es un 0");
    cbString = cbString.replace("X",0+"");
}else{
    if (pos%2==0){
        JOptionPane.showMessageDialog(null, "El
        dígito borrado es un "+(10 - (suma%10)));
        cbString = cbString.replace("X",(10 -
        (suma%10)+""));
    }else{
        //Los pares por lo de que pos empieza en
        0..
        cbString =
        cbString.replace("X",(3*(suma%10))%10+"");
        JOptionPane.showMessageDialog(null, "El
        dígito borrado es un "+(3*(suma%10))%10);
    }
}

return cbString;
}
}
}

```

#### 9.4. Para Borrar Random

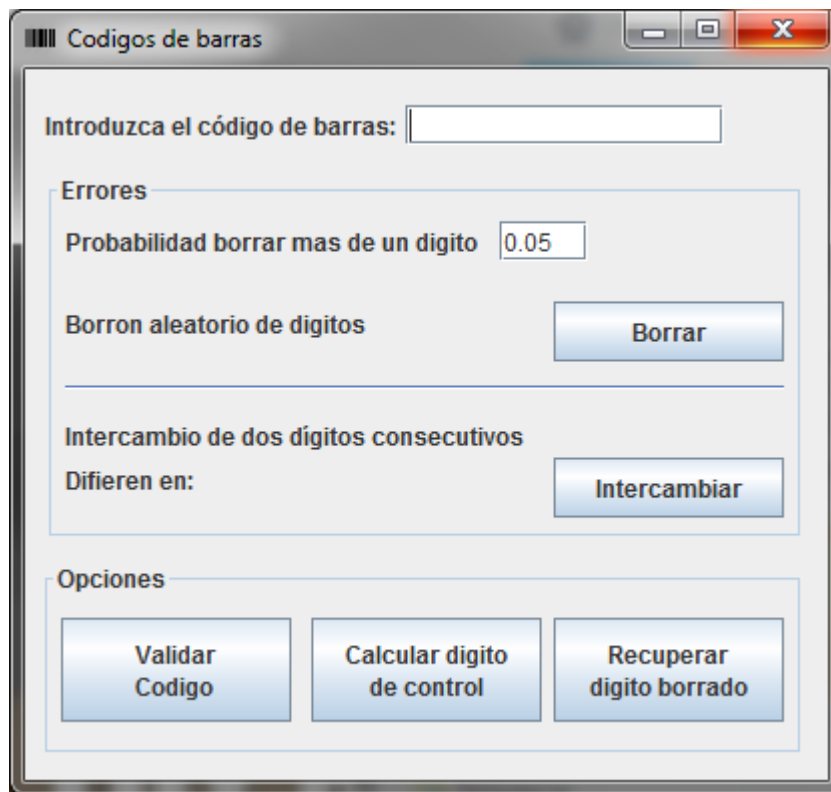
Al constructor le pasamos el código y la probabilidad introducidas por el usuario. Transformamos la probabilidad a decimal.

```
public class BorrónRandom {  
  
    private String codigo;  
    private double probabilidad;  
  
    public BorrónRandom(String c, String p) {  
        this.codigo = c;  
        this.probabilidad = Double.valueOf(p);  
    }  
}
```

Convertimos el código en un array para cada posición, creamos para cada carácter un número aleatorio. Si ese número es menor que la probabilidad introducimos un borrón y si no nada.

```
    public String borrar() {  
  
        char[] caracteres = codigo.toCharArray();  
  
        for(int i=0; i<caracteres.length; i++) {  
            double aleat = Math.random();  
            if(aleat < probabilidad)  
                caracteres[i] = 'X';  
        }  
  
        return String.valueOf(caracteres);  
    }  
}
```

## 10. Interfaz



El código fuente del Interfaz gráfico es el siguiente:

```
public class Interfaz extends JFrame implements ActionListener{

    private static final long serialVersionUID = 1L;
    private JLabel lblPedirCodigo;
    private JTextField txtCodigoBarras;
    private JLabel lblCorrecto;
    private JLabel lblIncorrecto;
    private JPanel panelOpciones;
    JButton btnValidar;
    private JButton btnDigitoControl;
    private JButton btnDigitoBorrado;
    private JPanel panelErrores;
    private JButton btnDigitosConsecutivos;
    private JSeparator separator;
    private JLabel lblIntercambio;
    private JLabel lblDifieren;
    private JLabel lblBorrón;
    private JTextField txtProbabilidad;
    private JLabel lblNewLabel;
    private JButton btnBorrarAleatorio;
    private JLabel lblDiferencia;

    public Interfaz() {

        setIconImage(Toolkit.getDefaultToolkit().getImage(Interfaz.class.getResource(
            e("/es/unileon/si/images/Barcode.png"))));
        this.setTitle("Codigos de barras");
        this.setBounds(100, 100, 417, 393);
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

```

        this.getContentPane().setLayout(null);

        lblPedirCodigo = new JLabel("Introduzca el código de barras:");
        lblPedirCodigo.setBounds(10, 11, 196, 35);
        this.getContentPane().add(lblPedirCodigo);

        txtCodigoBarras = new JTextField();
        txtCodigoBarras.setBounds(190, 18, 159, 20);
        getContentPane().add(txtCodigoBarras);

        lblCorrecto = new JLabel("");
        lblCorrecto.setIcon(new
        ImageIcon(Interfaz.class.getResource("/es/unileon/si/images/ok.png")));
        lblCorrecto.setBounds(359, 11, 30, 29);
        //no se ve, se pasa a true en caso que la validacion sea correcta
        lblCorrecto.setVisible(false);
        getContentPane().add(lblCorrecto);

        lblIncorrecto = new JLabel("");
        lblIncorrecto.setIcon(new
        ImageIcon(Interfaz.class.getResource("/es/unileon/si/images/fail.png")));
        lblIncorrecto.setBounds(359, 11, 30, 29);
        //no se ve, se pasa a true en caso que la validacion sea incorrecta
        lblIncorrecto.setVisible(false);
        getContentPane().add(lblIncorrecto);

        panelOpciones = new JPanel();
        panelOpciones.setBorder(new
        TitledBorder(UIManager.getBorder("TitledBorder.border"), "Opciones",
        TitledBorder.LEADING, TitledBorder.TOP, null, null));
        panelOpciones.setLayout(null);
        panelOpciones.setBounds(8, 246, 381, 97);
        getContentPane().add(panelOpciones);

        btnValidar = new
        JButton("<html><center>Validar<br>Codigo</center></html>");
        btnValidar.setBounds(10, 28, 115, 52);
        panelOpciones.add(btnValidar);

        btnDigitoControl = new JButton("<html><center>Calcular digito<br>de
        control</center></html>");
        btnDigitoControl.setBounds(135, 28, 115, 52);
        panelOpciones.add(btnDigitoControl);

        btnDigitoBorrado = new JButton("<html><center>Recuperar<br>digito
        borrado</center></html>");
        btnDigitoBorrado.setBounds(256, 28, 115, 52);
        panelOpciones.add(btnDigitoBorrado);

        panelErrores = new JPanel();
        panelErrores.setBorder(new TitledBorder(null, "Errores",
        TitledBorder.LEADING, TitledBorder.TOP, null, null));
        panelErrores.setBounds(10, 52, 379, 183);
        panelErrores.setLayout(null);
        getContentPane().add(panelErrores);

        btnDigitosConsecutivos = new JButton("Intercambiar");
        btnDigitosConsecutivos.setBounds(254, 142, 115, 30);
        panelErrores.add(btnDigitosConsecutivos);

```

```

        separator = new JSeparator();
        separator.setBounds(10, 106, 359, 9);
        panelErrores.add(separator);

        lblIntercambio = new JLabel("Intercambio de dos dígitos consecutivos");
        lblIntercambio.setBounds(10, 106, 234, 50);
        panelErrores.add(lblIntercambio);

        lblDifieren = new JLabel("Difieren en:");
        lblDifieren.setBounds(10, 134, 200, 38);
        panelErrores.add(lblDifieren);

        lblBorrón = new JLabel("Probabilidad borrar más de un dígito");
        lblBorrón.setBounds(10, 24, 247, 20);
        panelErrores.add(lblBorrón);

        txtProbabilidad = new JTextField();
        txtProbabilidad.setText("0.05");
        txtProbabilidad.setBounds(227, 24, 44, 20);
        panelErrores.add(txtProbabilidad);
        txtProbabilidad.setColumns(10);

        lblNewLabel = new JLabel("Borrón aleatorio de dígitos");
        lblNewLabel.setBounds(10, 50, 193, 50);
        panelErrores.add(lblNewLabel);

        btnBorrarAleatorio = new JButton("Borrar");
        btnBorrarAleatorio.setBounds(254, 64, 115, 30);
        panelErrores.add(btnBorrarAleatorio);

        lblDiferencia = new JLabel("");
        lblDiferencia.setBounds(84, 145, 25, 17);
        panelErrores.add(lblDiferencia);

        //escuchar evento boton cambio digitos
        this.btnDigitosConsecutivos.addActionListener(this);

        //escuchar evento boton borrón aleatorio
        this.btnBorrarAleatorio.addActionListener(this);

        //calcular dígito de control
        this.btnDigitoControl.addActionListener(this);

        //escuchar evento boton validar código
        this.btnValidar.addActionListener(this);

        this.btnDigitoBorrado.addActionListener(this);
    }

```

```

@Override
public void actionPerformed(ActionEvent e) {
    //accion evento cambio digitos
    if(e.getSource() == this.btnDigitosConsecutivos)
    {

```

```

        IntercambioDigitos cambio=new
IntercambioDigitos(this.txtCodigoBarras.getText());
        cambio.intercambio();
        this.txtCodigoBarras.setText(cambio.getCodigo());
        //mostramos la diferencia entre los digitos intercambiados
para ver si fucionara la validacion
        this.lblDiferencia.setText(cambio.getDiferencia()+"");
    }
    else if(e.getSource()== this.btnBorrarAleatorio) {
        BorronRandom borron = new
BorronRandom(txtCodigoBarras.getText(), txtProbabilidad.getText());
        txtCodigoBarras.setText(borron.borrar());
    }
    else if(e.getSource() == this.btnDigitoControl) {
        CalculoDigitoDeControl control = new
CalculoDigitoDeControl(this.txtCodigoBarras.getText());
        txtCodigoBarras.setText(control.mostrar());

    }else if(e.getSource() == this.btnValidar) {
        ValidarCodigo validar = new
ValidarCodigo(this.txtCodigoBarras.getText());
    }else if(e.getSource() == this.btnDigitoBorrado){
        RecuperarDigito recuperar = new RecuperarDigito();

        txtCodigoBarras.setText(recuperar.recuperarDigito(this.txtCodigoBarras.getT
ext()));
    }
}
}
}

```

## 11. Bibliografía

- [http://es.wikipedia.org/wiki/C%C3%B3digo\\_de\\_barras](http://es.wikipedia.org/wiki/C%C3%B3digo_de_barras)
- <http://www.monografias.com/trabajos11/yantucod/yantucod.shtml>
- <http://www.informatica-hoy.com.ar/informatica-tecnologia-empresas/Tipos-de-codigo-de-barras-y-sus-ventajas.php>
- <http://www.ajpdsoft.com/modules.php?name=news&file=article&sid=207>
- <http://www.timingargentina.com.ar/component/content/article/80-uncategorised/239-codigo-de-barras14>
- <http://www.delitosinformaticos.com/04/2014/noticias/detenido-un-hombre-por-falsificar-codigos-de-barras>
- [http://elpais.com/diario/1999/08/19/madrid/935061863\\_850215.html](http://elpais.com/diario/1999/08/19/madrid/935061863_850215.html)
- <https://www.lagranepoca.com/archivo/31986-fabricante-leche-tacha-numero-666-codigo-barra-sus-envases.html>