En este Apéndice se presenta el programa de Java que realiza la interfase para facilitar al usuario modificar la frecuencia de modulación.

```
import java.io.*;
import java.net.*;
import java.io.*;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

// método que realiza la ventana en la cual se introduce la frecuencia de modulación
public class moduladorSwing extends JFrame implements ActionListener{
JButton calcular;
JButton cerrar;
JTextField frecuenciaTextF;

public static void main(String args[])
    {
      float fc, coeff;
      moduladorSwing mod = new moduladorSwing();
      mod.ventanaFrecuencia();    }

private void ventanaFrecuencia()
          {
          JFrame frame = new JFrame("Cálculo del coeficiente para la frecuencia de modulación
          deseada");
          Container c = frame.getContentPane();
          JLabel frecuenciaLabel = new JLabel("Introduzca la frecuencia de modulación en Hz: ");
          frecuenciaTextF = new JTextField("0");
          calcular = new JButton("Calcular coeficiente");
          calcular.setActionCommand("calcular");
          calcular.addActionListener(this);
          cerrar = new JButton("Cerrar Ventana");
          cerrar.setActionCommand("cerrar");
          cerrar.addActionListener(this);
          c.setLayout(new GridLayout(2,2));
          c.add(frecuenciaLabel);
          c.add(frecuenciaTextF);
          c.add(calcular);
          c.add(cerrar);

          //frame.pack();
          frame.setSize(450,130);
          frame.setVisible(true); }

private void showInfoDialog( String dialogo){
                    JOptionPane.showMessageDialog(new JFrame(),dialogo); }

  public void actionPerformed( ActionEvent e ) {
                    String eventoStr = e.getActionCommand();
                              if (eventoStr.equals("calcular")){
                              if (!(frecuenciaTextF.getText()).equals("")){
                                        float f=(new Float(frecuenciaTextF.getText())).floatValue();
```

```
                        System.out.println("--calcular: " + f);
                        compara(f);
                    }
                    else
                      showInfoDialog("No se ha introducido ninguna frecuencia");
                    }
                if (eventoStr.equals("cerrar")){
                    System.out.println("--cerrar");
                    System.exit(0);
                }
        }
```

// método que lee la frecuencia de modulación introducida por el usuario

```
private float readf()
   {
        String a="";
        System.out.flush();
        DataInputStream ds= new DataInputStream(System.in);
        try
        {
                a=ds.readLine();
        }
        catch (IOException e)  {
                System.out.println(e);
        }
        float f=(new Float(a)).floatValue();
        return f;
   }
```

// realiza la comparación de la frecuencia introducida; dependiendo del rango manda modulo1 (en el caso de 1428 Hz-2856) o modulo 2 (para el rango de 2857 Hz-6666 Hz)

```
private void compara(float fc){
        if( (fc<1427) || (fc>6666))
          System.out.println("Frecuencia fuera del rango de modulación");
        else {
                if( (fc>=1428) && (fc<2857) )
                   modulo1(fc);
                else if ( (fc>=2857) && (fc<=6666))
                   modulo2(fc);}
  }
```

// calcula el valor de COEFF para el rango de modulación 1 aplicando la ecuación de la recta (tal como se explicó en la sección 5.3)

```
private void modulo1(float fc)
  {
        float coeff;
        float m;
        int x1=0,x2=0,y1=0,y2=0;

        if( (fc<1666) && (fc>=1428) )
        {
                x1 = 1428;
                x2 = 1666;
                y1 = 27000;
                y2 = 26000;
        }
```

```
        else if( (fc<1818) && (fc>=1666) )
                 {
                x1 = 1666;
                x2 = 1818;
                y1 = 25000;
                y2 = 24000;
        }
        else if ( (fc<1923) && (fc>=1818) )
        {
                x1 = 1818;
                x2 = 1923;
                y1 = 24000;
                y2 = 23000;
        }
        else if( (fc<2083) && (fc>=1923) )
        {
                x1 = 1923;
                x2 = 2083;
                y1 = 23000;
                y2 = 22000;
        }
        else if( (fc<2173) && (fc>=2083) )
        {
                x1 = 2083;
                x2 = 2173;
                y1 = 22000;
                y2 = 21000;
        }
        else if( (fc<2272) && (fc>=2173) )
        {
                x1 = 2173;
                x2 = 2272;
                y1 = 21000;
                y2 = 20000;
        }
        else if( (fc<2380) && (fc>=2272) )
        {
                x1 = 2272;
                x2 = 2380;
                y1 = 20000;
                y2 = 19000;
        }
        else if( (fc<2500) && (fc>=2380) )
        {
                x1 = 2380;
                x2 = 2500;
                y1 = 19000;
                y2 = 18000;
}
        else if( (fc<2631) && (fc>=2500) )
        {
                x1 = 2500;
                x2 = 2631;
```

```
                y1 = 17000;
                y2 = 16000;
        }
        else if( (fc<2702) && (fc>=2631) )
        {
                x1 = 2631;
                x2 = 2702;
                y1 = 16000;
                y2 = 15000;
        }
        else if( (fc<2777) && (fc>=2702) )
        {
                x1 = 2702;
                x2 = 2777;
                y1 = 15000;
                y2 = 14000;
        }
        else if( (fc<2857) && (fc>=2777) )
        {
                x1 = 2777;
                x2 = 2857;
                y1 = 14000;
                y2 = 13000;
        }

        m = calculaPendiente(x1,x2,y1,y2);
        coeff = Math.round(m*(fc-x1)+y1);
        System.out.println("coeff ="+ coeff);
        escribeCoeff("MOD1.asm", (int)coeff);
  }
// calcula el valor de COEFF para el rango 2 aplicando la ecuación de la recta
private void modulo2(float fc)
  {
        float coeff;
        float m;
        int x1=0,x2=0,y1=0,y2=0;
        if( (fc<3225) && (fc>=2857) )
        {
                x1 = 2857;
                x2 = 3225;
                y1 = 29000;
                y2 = 28000;
        }
        else if( (fc<3571) && (fc>=3225) )
        {
                x1 = 3225;
                x2 = 3571;
                y1 = 28000;
                y2 = 27000;
        }
        else if( (fc<3846) && (fc>=3571) )
        {
                x1 = 3571;
```

```
        x2 = 3846;
        y1 = 27000;
        y2 = 26000;
}
else if( (fc<4166) && (fc>=3846) )
{
        x1 = 3846;
        x2 = 4166;
        y1 = 26000;
        y2 = 25000;
}
else if( (fc<4347) && (fc>=4166) )
{
        x1 = 4166;
        x2 = 4347;
        y1 = 25000;
        y2 = 24000;
}
else if( (fc<4761) && (fc>=4347) )
{
        x1 = 4347;
        x2 = 4761;
        y1 = 24000;
        y2 = 23000;
}
else if( (fc<5000) && (fc>=4761) )
{
        x1 = 4761;
        x2 = 5000;
        y1 = 23000;
        y2 = 22000;
}
else if( (fc<5263) && (fc>=5000) )
{
        x1 = 5000;
        x2 = 5263;
        y1 = 22000;
        y2 = 21000;
}
else if( (fc<5555) && (fc>=5263) )
{
        x1 = 5263;
        x2 = 5555;
        y1 = 21000;
        y2 = 20000;
}
else if( (fc<5882) && (fc>=5555) )
{
        x1 = 5555;
        x2 = 5882;
        y1 = 19000;
        y2 = 18000;
}
```

```
        else if( (fc<6060) && (fc>=5882) )
        {
                x1 = 5882;
                x2 = 6060;
                y1 = 18000;
                y2 = 17000;
        }
        else if( (fc<6451) && (fc>=6060) )
        {
                x1 = 6060;
                x2 = 6451;
                y1 = 17000;
                y2 = 16000;
        }
        else if( (fc<=6666) && (fc>=6451) )
         {
                x1 = 6451;
                x2 = 6666;
                y1 = 15000;
                y2 = 14000;
        }
        m = calculaPendiente(x1,x2,y1,y2);
        coeff = Math.round(m*(fc-x1)+y1);
        System.out.println("coeff ="+ coeff);
        escribeCoeff("MOD2.asm", (int)coeff);
 }
```
// calcula la pendiente tomando el P1(x1,y1) y el P2(x2,y2) dependiendo del rango en el que se encuentre la frecuencia introducida por el usuario
```
private float calculaPendiente(float x1,float x2,float y1,float y2)
  {
        float m;
        m = (y2 - y1)/(x2-x1);
        System.out.println("la pendiente calculada es ="+ m);
        return m;
  }
```
// en esta parte del programa se escribe la leyenda que inicializa el valor de COEFF en el archivo
```
private void escribeCoeff(String nombreArchivo, int coeff)
  {
         String texto = "COEFF    .word    " + coeff;
```
// lee línea por línea el archivo correspondiente (MOD1.asm o MOD2.asm) y cuando encuentra el símbolo &, pega la leyenda que inicializa COEFF
```
         String checa = "&";
         BufferedReader br=null;
         FileInputStream in = null;
         String linea;
         PrintStream misalida=null;
         FileOutputStream fout=null;
         String s = null;
         try {
       in = new FileInputStream("C:/tms/" +nombreArchivo);
       br = new BufferedReader(new InputStreamReader(in));
       fout = escritura(nombreArchivo);
       misalida = new PrintStream(fout);
```

```
        while ((linea= br.readLine()) != null )
        {
         if (!linea.equals("")) {
                        StringBuffer checaDato = new StringBuffer(linea);
                         charDato= checaDato.charAt(0);
                        switch(charDato)
                        {
                                case '&':
                                        linea=texto;
                                        System.out.println("--  & Encontrado--");
                                break;
                        }
                 }
                escribir(misalida,linea);
       }
       System.out.println("-- Se termino de escribir en el archivo ---");
       if (fout!=null)
                        fout.close();
                  if (misalida!=null)
// cierra el archivo MODUL1.asm o MODUL2.asm
                        misalida.close();
                   if (br!=null)
                        br.close();

                  if (in!=null)
                        in.close();
                System.out.println("-- Se cerro  el archivo --- ");
 // abre el sistema operativo para ensamblar el archivo
                  if (nombreArchivo.equals("MOD1.asm")) {
                   System.out.println("== ensamblar modul1");
                try{
 // escribe el comnado para ensamblar el archivo
                Process proceso = Runtime.getRuntime().exec("dsk5aMODUL1.asm");
                BufferedReader stdInput = new BufferedReader(new
                InputStreamReader(proceso.getInputStream()));
                BufferedReader stdError = new BufferedReader(new
                  InputStreamReader(proceso.getErrorStream()));
                  // read the output from the command
                System.out.println("Here is the standard output of the command:\n");
                                 while ((s = stdInput.readLine()) != null) {
                    System.out.println(s);
                        }
                                 // read any errors from the attempted command
                                System.out.println("Here is the standard error of the command (if any):\n");
                            while ((s = stdError.readLine()) != null) {
                                System.out.println(s);
                        }
                }
                catch (Exception e){
                System.out.println("error al ejecutar el comando en MS-DOS para ensamblar MODUL1.asm");
                        }
                   }
                   else  {
```

```
                    System.out.println("== ensamblar modul2");
                            try{
                      Process proceso = Runtime.getRuntime().exec("dsk5a MODUL2.asm");
                    BufferedReader stdInput = new BufferedReader(new
                    InputStreamReader(proceso.getInputStream()));
                    BufferedReader stdError = new BufferedReader(new
                    InputStreamReader(proceso.getErrorStream()));
                    // read the output from the command
                    System.out.println("Here is the standard output of the command:\n");
                    while ((s = stdInput.readLine()) != null) {
                    System.out.println(s);
                    }
                    // read any errors from the attempted command
                    System.out.println("Here is the standard error of the command (if any):\n");
                    while ((s = stdError.readLine()) != null) {
                            System.out.println(s);
                    }  }
                    catch (Exception e)
                    {System.out.println("error al ejecutar el comando en MS-DOS para ensamblar
MODUL2.asm");
                            }
                }
            }
                    catch (Exception mue) {
                    System.out.println("Error al leer el archivo");
                    System.out.println(mue);  }
        }
  private void escribir(PrintStream salida, String texto){
            PrintStream salidaTxt = new PrintStream(salida);
            salidaTxt.println(texto);
  }
  // guarda el archivo en el cual se realizó la copia con el nombre MODUL1.asm
  private FileOutputStream escritura(String nombreArchivo){
            FileOutputStream fout = null;
            PrintStream misalida = null;
            String rutaSalida="";

    if (nombreArchivo.equals("MOD1.asm")) {
        rutaSalida = "C:/tms/MODUL1.asm";
        System.out.println("Soy mod1.txt");
    }
    else
// guarda el archivo con el nombre MODUL2.asm
            rutaSalida = "C:/tms/MODUL2.asm";
    try {
        fout = new FileOutputStream(rutaSalida,false);
    }
    catch(Exception e){
            System.out.println("-- Error el escribir al archivo");
            System.out.println(e);
    }
    return fout; }
}
```