

1ª Evaluación. Ejercicio 2

Nombre: _____

Implementa la clase **Mediciones** que guarda el instante de tiempo en el que se mide la audiencia de televisión (hora y minutos), la cadena de televisión observada y el número de personas (en decenas de miles) de los distintos segmentos de población que estaban viendo esa cadena.

```
public class Medicion implements Serializable {
    int horas;
    int minutos;
    String nombre;
    int []audiencia_seg=new int[4];

    public Medicion(int horas, int minutos, String nombre,int[]audiencia_seg) {
        this.horas = horas;
        this.minutos = minutos;
        this.nombre = nombre;
        this.audiencia_seg=audiencia_seg;
    }

    public int getHoras() {
        return horas;
    }

    public void setHoras(int horas) {
        this.horas = horas;
    }

    public int getMinutos() {
        return minutos;
    }

    public void setMinutos(int minutos) {
        this.minutos = minutos;
    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public int[] getAudiencia_seg() {
        return audiencia_seg;
    }

    public void setAudiencia_seg(int[] audiencia_seg) {
        this.audiencia_seg = audiencia_seg;
    }
}
```

Contamos con el archivo de texto ya creado (se proporciona) de nombre **mediciones.txt** que tiene el siguiente contenido.

```
00,15,cad1,0.8,25.2,22.43,4.81
16,30,cad1,35.0,5.4,17.06,19.41
12,10,cad2,23.5,11.4,14.5,13.1
18,05,cad3,25.2,7.2,17.2,14.6
22,01,cad1,25.4,9.1,45.1,2.4
20,09,cad2,10.3,16.4,90.3,9.3
21,00,cad2,10.0,32.2,35.2,4.7
23,00,cad3,9.0,14.8,16.1,11.1
23,45,cad1,0.5,21.9,14.3,8.9
23,12,cad3,0.2,50.3,43.5,6.3
20,00,cad3,11.3,61.2,18.9,4.1
20,20,cad2,18.9,18.6,16.3,20.8
21,18,cad1,25.2,20.3,13.9,45.9
22,12,cad2,1.9,14.2,10.2,63.2
21,00,cad2,1.2,19.9,7.3,4.1
03,30,cad3,0.02,20.0,9.4,13.2
10,45,cad2,4.8,32.1,18.1,27.1
08,22,cad3,2.2,35.4,23.2,19.4
07,18,cad3,0.9,16.2,14.9,45.1
12,00,cad1,11.3,14.8,6.4,18.4
13,10,cad2,15.2,21.3,13.7,21.3
13,05,cad3,16.9,4.5,20.0,14.9
```

Por ejemplo la primera línea mide la audiencia de la cadena de televisión de nombre cad1, a las 00:15, en los distintos segmentos de audiencia (en decenas de mil).

Segmentos : NIÑOS, JÓVENES, ADULTOS, MAYORES, TODOS

En este caso:

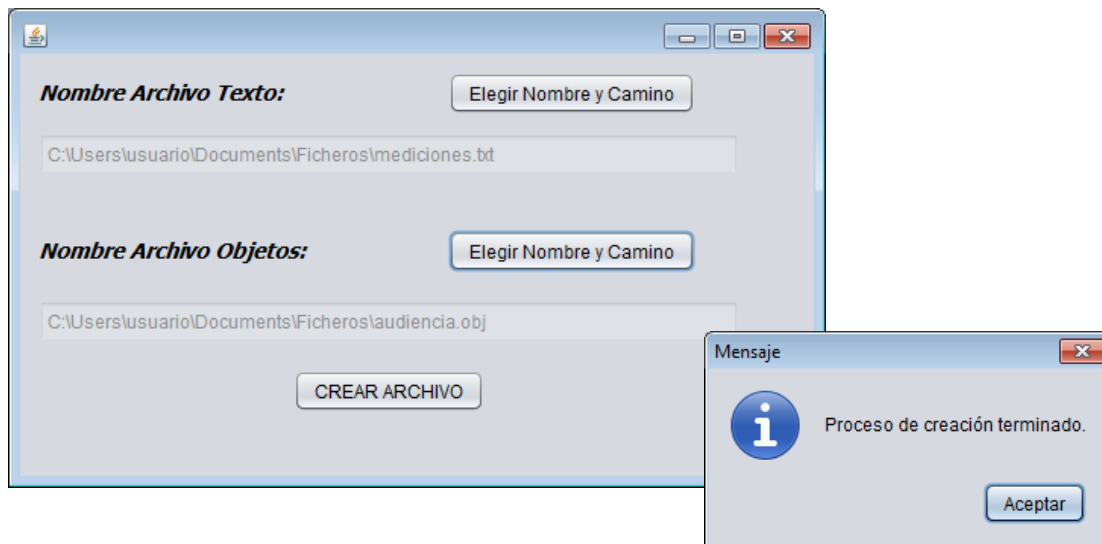
0.8*10000= 8.000 niños,
25.2*10000= 252.000 jóvenes,
22.43*10000=224.300 adultos,
4.81*10000=4.8100 mayores.

La audiencia total (segmento TODOS) no está en el fichero y se calcula como la suma de los cuatro valores.

Puede haber varias mediciones a la misma hora para la misma o distintas cadenas.

Las cadenas que existen son: **cad1, cad2, cad3**.

1.- Diseña un programa java (**Crear.java**) que a partir del archivo de texto proporcionado (mediciones.txt) cree el fichero binario de objetos **Medicion** de nombre **audiencia.obj**

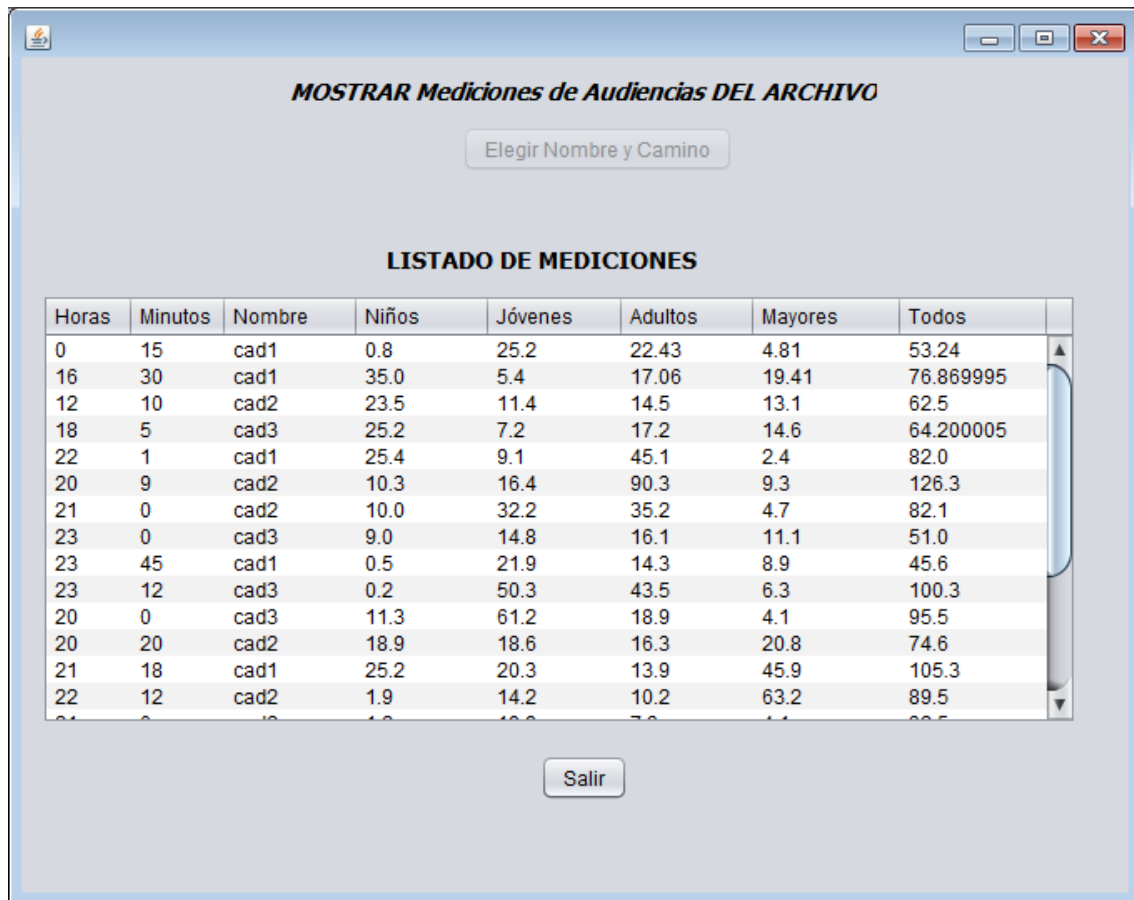


Métodos:

```
public void crear(String fichorigen, String fichdestino ) { ... }
```

El fichero origen es de texto y el fichero destino es de objetos. Por cada línea del fichero de texto se crea un objeto.

2.- Diseña un programa java (Listar.java) que muestre el contenido de **audiencia.obj** en una tabla



Métodos:

```
public ArrayList paraMostrar (String nomfich) { ...}
```

Crea una lista de objetos a partir de un archive de objetos.

```
public void LlenarTabla (String nombrefich, DefaultTableModel m, JTable  
tabla){...}
```

A partir de una lista creada con el método anterior, muestra su contenido en un JTable.

3.- Una vez creado el archivo **audiencia.obj** escribe un programa java **CuotasPantalla.java** que consulte este archivo y nos muestre el total de personas que han visto cada cadena ese día por franja horaria y el total. Utiliza una matriz de contadores de 3 x 4

	MADRUGADA	MAÑANA	TARDE	PRIME TIME	TOTAL
cad1					
cad2					
cad3					

Las franjas horarias son:

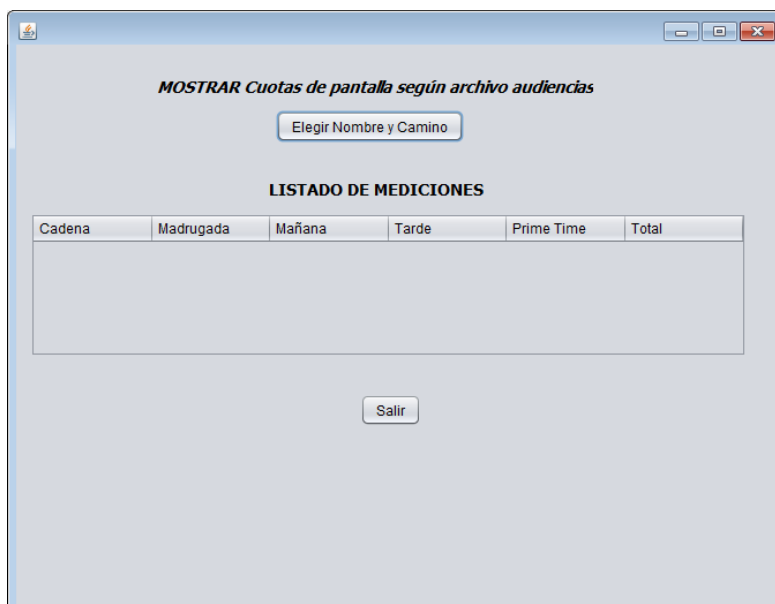
MADRUGADA: 00:00 a 07:00
 MAÑANA: 07:00 a 14:00
 TARDE: 14:00 a 20:00
 PRIME TIME: 20:00 a 00:00


```
public int[][] CargarMatriz (String nomfich){ }
```




Abre fichero de objetos para leer, se recorre y se van incrementando las posiciones de la matriz de contadores. La columna TOTAL es calculada.

```
public void MostrarMatriz (String nombrefich,DefaultTableModel m, JTable  
tabla){ }
```

Mostramos los datos de la matriz de contadores en un Jtable.







MOSTRAR Cuotas de pantalla según archivo audiencias

Elegir Nombre y Camino

LISTADO DE MEDICIONES

Cadena	Madrugada	Mañana	Tarde	Prime Time	Total
cad1	532400	509000	768700	2329000	4139100
cad2	0	2161000	0	4050000	6211000
cad3	426200	1766000	642000	2468000	5302200

Salir

SOLUCIÓN:

```
public class ControlArchivo {

    public void crear(String fichorigen,String fichdestino ) {

        File ftexto=new File(fichorigen);
        if (ftexto.exists()){
            FileReader fr = null;
            BufferedReader bf=null;
            FileOutputStream fs=null;
            ObjectOutputStream os = null;
            try {
                fr = new FileReader(fichorigen);
                bf=new BufferedReader(fr);
                fs=new FileOutputStream(fichdestino);
                os = new ObjectOutputStream(fs);

                String linea,token;
                StringTokenizer st;
                while((linea=bf.readLine())!=null)
                {
                    //descomponer la linea en tokens

                    st = new StringTokenizer(linea,"");
                    int horas;token=st.nextToken();horas=Integer.parseInt(token);
                    int minutos;token=st.nextToken();minutos=Integer.parseInt(token);
                    String nombre=st.nextToken();
                    float []audiencia_seg=new float[4];
                    token=st.nextToken();audiencia_seg[0]=Float.parseFloat(token);
                    token=st.nextToken();audiencia_seg[1]=Float.parseFloat(token);
                    token=st.nextToken();audiencia_seg[2]=Float.parseFloat(token);
                    token=st.nextToken();audiencia_seg[3]=Float.parseFloat(token);
                    //crear objeto y añadir
                    Medicion obj=new Medicion(horas,minutos,nombre,audiencia_seg);
                    System.out.println(obj.toString());
                    //añadir al fichero de objetos
                    os.writeObject(obj);
                    //JOptionPane.showMessageDialog(null, "Medición guardada.");

                }
                JOptionPane.showMessageDialog(null, "Proceso de creación terminado.");
            } catch (IOException ex) {
                Logger.getLogger(ControlArchivo.class.getName()).log(Level.SEVERE, null,
ex);
            } finally {
                try {
                    fr.close();bf.close();
                    fs.close();os.close();
                } catch (IOException ex) {
```

```

        Logger.getLogger(ControlArchivo.class.getName()).log(Level.SEVERE,
null, ex);
    }
}
else{

    JOptionPane.showMessageDialog(null, "El archivo de texto no existe.");
}
}

```

```

public ArrayList paraMostrar(String nomfich) {

    //Recorremos el archivo y añadimos cada objeto a una lista
    ArrayList<Medicion> lista = new ArrayList<Medicion>();

    FileInputStream fs;
    ObjectInputStream is;

    try{

        fs=new FileInputStream(nomfich);
        is = new ObjectInputStream(fs);
        Medicion obj=null;
        boolean b=true;
        try{
            obj=(Medicion)is.readObject();
            //while(obj!=null)
            while(b)
            {
                if (obj instanceof Medicion)
                {
                    lista.add(obj);

                }
                //is = new ObjectInputStream(fs);
                obj=(Medicion)is.readObject();
            }
            is.close();
        }catch EOFException eot){

            //JOptionPane.showMessageDialog(null, "Fin de fichero.");
        }
        catch (ClassNotFoundException ex) {

            JOptionPane.showMessageDialog(null, "Clase no encontrada.");
        }
        finally {
            //JOptionPane.showMessageDialog(null, "Proceso finalizado.");
            is.close();
        }
    }
}

```



```

    } catch (IOException ex) {
        JOptionPane.showMessageDialog(null, "Problema con el fichero (N0 existe.)");
        return null;
    }
    System.out.println("tamaño: "+lista.size());
    return lista;
}

public void LlenarTabla(String nombrefich,DefaultTableModel m, JTable tabla){
    //PrepararTabla();
    ArrayList <Medicion> lista=paraMostrar(nombrefich);
    Medicion md;
    Object [] fila=new Object[8];
    for (int i=0; i<lista.size();i++)
    {
        md=lista.get(i);
        fila[0]=md.getHoras();
        fila[1]=md.getMinutos();
        fila[2]=md.getNombre();
        float []audiencia_seg=md.getAudiencia_seg();
        fila[3]=audiencia_seg[0];//niños
        fila[4]=audiencia_seg[1];//jóvenes
        fila[5]=audiencia_seg[2];//adultos
        fila[6]=audiencia_seg[3];//mayores
        float total=0;
        for(int j=0; j<4;j++){
            total+=audiencia_seg[j];
        }
        fila[7]=total;//todos
        m=(DefaultTableModel)tabla.getModel();//recoge el modelo de la tabla
        m.addRow(fila);
    }
}

public int[][] CargarMatriz (String nomfich){
    int[][] matriz=new int [3][4];
    int fila=0,columna=0;
    //Abrir fichero para leer, recorrer e ir incrementando contadores

    FileInputStream fs;
    ObjectInputStream is;

    try{

        fs=new FileInputStream(nomfich);
        is = new ObjectInputStream(fs);
        Medicion obj=null;
        boolean b=true;
        try{
            obj=(Medicion)is.readObject();
            //while(obj!=null)
            while(b)

```

```

{
    int hora=obj.getHoras();
    int minutos=obj.getMinutos();
    String cadena=obj.getNombre();
    float []audiencia_seg=obj.getAudiencia_seg();
    //calcular total audiencia para esa medición en personas
    int total=0;
    for (int i=0;i<4;i++)
    {
        total+=audiencia_seg[i]*10000;
    }

    //calcular fila
    if (cadena.equals("cad1"))
        fila=0;
    else if (cadena.equals("cad2"))
        fila=1;
    else //if (cadena.equals("cad3"))
        fila=2;
    //calcular columna
    if (hora>=0 && hora<7) columna=0;
    else if (hora>=7 && hora<14) columna=1;
    else if (hora>=14 && hora <20) columna=2;
    else columna=3; //hora >=20

    System.out.print(matriz[fila][columna]+"\\t");
    matriz[fila][columna]+=total;
    //is = new ObjectInputStream(fs);
    obj=(Medicion)is.readObject();
    System.out.println(cadena+" "+fila+" "+columna+"
"+total+"\\t"+matriz[fila][columna]+" ");
}
//pintar prueba

//is.close();
} catch (EOFException eot){

    //JOptionPane.showMessageDialog(null, "Fin de fichero.");
}
catch (ClassNotFoundException ex) {

    JOptionPane.showMessageDialog(null, "Clase no encontrada.");
}
finally {
    //JOptionPane.showMessageDialog(null, "Proceso finalizado.");
    is.close();
    /*for (int i=0;i<3;i++)
    {
        for (int j=0;j<=3;j++)
        {
            System.out.print(matriz[i][j]+" * ");
        }
    }
}

```

```

        System.out.println("");
    }*/
}

} catch (IOException ex) {
    JOptionPane.showMessageDialog(null, "Problema con el fichero (N0 existe.)");
    return null;
}
return matriz;

}

public void MostrarMatriz (String nombrefich,DefaultTableModel m, JTable
tabla){
    String[] nombre_cad={"cad1","cad2","cad3"};
    int[][] matriz=CargarMatriz (nombrefich);
    for (int i=0; i<matriz.length;i++)
    {

        Object [] fila=new Object[6];
        fila[0]=nombre_cad[i];
        int total_cadena=0;

        for (int j=0;j<matriz[0].length;j++)
        {

            fila[j+1]=matriz[i][j];
            total_cadena+=matriz[i][j];
        }
        fila[5]=total_cadena;//todos
        m=(DefaultTableModel)tabla.getModel();//recoge el modelo de la tabla
        m.addRow(fila);
    }
}
}

```

CREAR:

```
public class Crear extends javax.swing.JFrame {
    String nombrefich1="";
    String nombrefich2="";

    ControlArchivo miarchivo= new ControlArchivo();

    public Crear() {
        initComponents();
    }

    .....

    private void btnNombre1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        //btnNombre2.setEnabled(true);
        int resp;
        resp=elegirFichero.showOpenDialog(Panel1);
        if(resp==JFileChooser.APPROVE_OPTION){
            nombrefich1=elegirFichero.getSelectedFile().toString();
            nombre1.setText(nombrefich1);
            nombre1.setVisible(true);

        }else if (resp==JFileChooser.CANCEL_OPTION){
            JOptionPane.showMessageDialog(null, "Se pulsó la opción Cancelar.");
        }
    }

    private void btnNombre2ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        int resp;
        resp=elegirFichero.showOpenDialog(Panel1);
        if(resp==JFileChooser.APPROVE_OPTION){
            nombrefich2=elegirFichero.getSelectedFile().toString();
            nombre3.setText(nombrefich2);
            nombre3.setVisible(true);
            btnCrear.setEnabled(true);

        }else if (resp==JFileChooser.CANCEL_OPTION){
            JOptionPane.showMessageDialog(null, "Se pulsó la opción Cancelar.");
        }
    }

    private void btnCrearActionPerformed(java.awt.event.ActionEvent evt) {

        miarchivo.crear(nombrefich1, nombrefich2);

        btnCrear.setEnabled(false);
    }
}
```

LISTAR:

```
public class Listar extends javax.swing.JFrame {
    String nombrefich="";
    ControlArchivo miarchivo= new ControlArchivo();
    DefaultTableModel m;

    public Listar() {
        initComponents();
        PrepararTabla();
        jPanel1.setVisible(true);
        jPanel2.setVisible(false);
    }
    .....
    private void btnNombreActionPerformed(java.awt.event.ActionEvent evt) {
        int resp;
        resp=elegirFichero.showOpenDialog(jPanel1);
        if(resp==JFileChooser.APPROVE_OPTION){
            nombrefich=elegirFichero.getSelectedFile().toString();
            //jPanel1.setVisible(false);
            btnNombre.setEnabled(false);
            jPanel2.setVisible(true);

            System.out.println(nombrefich);

            miarchivo.LlenarTabla(nombrefich,m,tabla);

        }else if (resp==JFileChooser.CANCEL_OPTION){
            JOptionPane.showMessageDialog(null, "Se pulsó la opción Cancelar.");
        }
    }
    private void PrepararTabla()
    {
        String
        titulos[]={ "Horas","Minutos","Nombre","Niños","Jóvenes","Adultos","Mayores","Todos" };
        m=new DefaultTableModel(null,titulos);
        tabla.setModel(m);

    }
    private void SalirActionPerformed(java.awt.event.ActionEvent evt) {

        this.dispose();
    }
}
```

CUOTAS DE PANTALLA:

```
public class CuotasPantalla extends javax.swing.JFrame {

    String nombrefich="";
    ControlArchivo miarchivo= new ControlArchivo();
    DefaultTableModel m;

    public CuotasPantalla() {
        initComponents();
        PrepararTabla();
        jPanel1.setVisible(true);
        //jPanel2.setVisible(false);
    }

    private void PrepararTabla()
    {
        String titulos[]={ "Cadena", "Madrugada", "Mañana", "Tarde", "Prime
Time", "Total" };
        m=new DefaultTableModel(null,titulos);
        tabla1.setModel(m);

    }

    private void btnNombreActionPerformed(java.awt.event.ActionEvent evt) {

        int resp;
        resp=elegirFichero.showOpenDialog(jPanel1);
        if(resp==JFileChooser.APPROVE_OPTION){
            nombrefich=elegirFichero.getSelectedFile().toString();
            btnNombre.setEnabled(false);
            jPanel3.setVisible(true);

            System.out.println(nombrefich);

            miarchivo.MostrarMatriz (nombrefich,m,tabla1);

        }else if (resp==JFileChooser.CANCEL_OPTION){
            JOptionPane.showMessageDialog(null, "Se pulsó la opción Cancelar.");
        }
    }

    private void Salir1ActionPerformed(java.awt.event.ActionEvent evt) {

        this.dispose();
    }
}
```