Manual Técnico

```
private void almacenar() {...57 lines }
```

El método almacenar(), lee el archivo según la ruta del campo de texto, entrando en un bucle donde convierte cada línea separada por "," como un arreglo de dos datos, el primero el país y el segundo el valor.

Se verifica si el nombre del país no esta repetido en el arreglo, si esta repetido, junto a su nombre se le agregara un número consecutivo y se agrega al arreglo, si este no esta repetido, simplemente se agrega al arreglo.

```
public void Graficar(db_datos[] arr) {...26 lines }
```

El método Graficar(), toma como parámetro el arreglo de datos que se va a ordenar, borrando el contenido del panel donde se graficara, y creando un bucle recorriendo el arreglo para obtener los países y los valores que se van a graficar.

```
public void limpiar() {...24 lines }
```

El método limpiar(), se encarga de limpiar todos los datos al seleccionar un nuevo archivo, este deselecciona los checkbox, oculta el botón "Generar Reporte", borra el contenido del campo de texto de ruta y del título, así también borra el contenido del panel a graficar.

```
public static void desactivar() {...16 lines }
```

El método desactivar(), se encarga de desactivar todos los campos de texto, botones y checkbox al momento de ordenar la gráfica.

```
public static void desactivarOrden() {...7 lines }
```

El método desactivarOrden(), se encarga de desactivar los checkbox y el botón de ordenar si no se ha graficado ningún archivo CSV.

```
public static void activar() {...16 lines }
```

El método activar(), se encarga de activar todos los campos de texto, botones y checkbox al finalizar el ordenamiento.

```
public static void activarOrden() {...7 lines }
```

El método activarOrden(), se encarga de activar los checkbox y el botón de ordenar al momento de graficar un archivo CSV.

```
public Cronometro(int minutos, int segundos) {...5 lines }
```

El constructor Cronometro(), pide como parámetros los minutos y segundos en los que se va a iniciar el conteo, por defecto este se inicia en 0 minutos 0 segundos.

```
public synchronized void parar() {...3 lines }
```

El método parar(), cambia el se encarga de parar el conteo del cronometro, este se llama al finalizar cualquier método de organización.

```
@Override
public void run() {...32 lines }
```

El método run(), se encarga de insertar el contador de “minutos : segundos” con formato “00:00”, en el label tiempo para mostrar el tiempo transcurrido.

```
public static db_datos[] datos = new db_datos[2];
public static db_datos[] original;
```

Se crean dos arreglos estáticos del constructor db_datos, el arreglo “datos “, será utilizado como los datos a ordenar, y el arreglo “original” almacenara los datos originales del archivo.

```
public static db_datos[] redimOrdenados(db_datos[] arr) {...7 lines }
```

El método redimOrdenados(), pide como parámetro el arreglo de datos a ordenar, este se encargara de aumentar en 1, la capacidad que este puede almacenar.

```
public static boolean capacidadOrdenados(db_datos[] arr) {...10 lines }
```

El método capacidadOrdenados(), pide como parámetro el arreglo de datos a ordenar, este se encarga de verificar si el arreglo ya llego a su máxima capacidad.

```
private int metodo = -1;
private int orden = -1;
private db_datos[] arr = null;
```

Se crearon tres variables privadas que se encarga de almacenar, el tipo de método a utilizar, el tipo de Orden a utilizar y el arreglo de datos a ordenar.

```
public void Ordenar(int _metodo, int _orden, db_datos[] _arr) {...6 lines }
```

El método Ordenar(), pide como parámetro el método a utilizar, el tipo de orden a utilizar y el arreglo a ordenar para definir las variables privadas anteriores.

```
private void EmpezarOrden(int metodo, int orden, db_datos[] arr) {...15 lines }
```

El método EmpezarOrden(), recibe como parámetros las variables privadas anteriores, este método es llamado en el método run(), este se encarga ejecutar el método dependiendo el tipo de orden y el tipo de método a utilizar que se pedía en el método Ordenar().

```
private void shellSort(boolean orden, db_datos[] arr) {...71 lines }
```

El método shellSort(), pide como parámetro el tipo de orden y el arreglo, si parámetro orden es verdadero, este ejecuta el método de ordenamiento de manera ascendente, si el parámetro orden es falso, este se encarga de ejecutar el método de ordenamiento de manera descendente.

```
private void burbuja(boolean orden, db_datos[] arr) {...70 lines }
```

El método burbuja(), pide como parámetro el tipo de orden y el arreglo, si parámetro orden es verdadero, este ejecuta el método de ordenamiento de manera ascendente, si el parámetro orden es falso, este se encarga de ejecutar el método de ordenamiento de manera descendente.

```
public void Graficar(db_datos[] arr) {...23 lines }
```

El método Graficar(), pide como parámetro el arreglo a graficar, este es llamado en los métodos de ordenamiento dentro de su bucle, cada vez que el método de ordenamiento hace un cambio en el arreglo, este lo envía al método graficar y este se encarga de graficar los nuevos valores con sus posiciones cambiadas.