```java
package Ejercicio03;

public class ColaArrayObj implements ColaObj {

    //variables
    private static boolean flag = false; //para controlar la impresion de la linea

    //variables miembro
    private static final int LONGITUD_POR_DEFECTO = 10;
    private int maxLongitud;        //Tamaño por defecto
    private int cabeza;             //indice del primero de la cola
    private int fin;                //indice del ultimo de la cola
    private Object[] datos;            //array que almacena elementos

    //constructores
    public ColaArrayObj() { this(LONGITUD_POR_DEFECTO);}

    public ColaArrayObj(int max) {
        this.maxLongitud = max + 1;  //un espacio extra
        this.fin = 0;
        this.cabeza = 1;
        datos = new Object[maxLongitud];
    }

    //getters
    public int getFin() {return this.fin;}
    public int getCabeza() {return this.cabeza;}
    public int getMaxLongitud() {return maxLongitud;}

    //metodos
    public void vaciar() {
        this.fin = 0;
        this.cabeza = 1;
        flag = false;
    }

    /** Añadir a la cola e */
    public boolean encolar(Object o) {
        //si no se cumple esta condicion, la cola esta llena
        if ((this.fin + 2) % this.maxLongitud != this.cabeza){
            this.fin = (this.fin + 1) % this.maxLongitud; // Incremento circular
            this.datos[this.fin] = o;
            flag = true;
            return true;
        }
        return false;
    }

    /** Eliminar y devolver el primer elemento (cabeza) */
     public Object desencolar() {
        //si no se cumple, la cola esta vacia
```

```
52          if (this.longitud() != 0){
53              Object o = this.datos[this.cabeza];
54              this.cabeza = (this.cabeza + 1) % this.maxLongitud; // Incremento Circular
55              return o;
56          }
57          return null;
58      }
59
60      /** @return primer valor */
61       public Object primero() {
62          //sino se cumple: "La cola está vacía";
63          if(this.longitud() != 0) return this.datos[this.cabeza];
64          return null;
65      }
66
67      /** @return Cantidad de elementos en la cola */
68       public int longitud(){
69          return ((this.fin + this.maxLongitud) - this.cabeza + 1) % this.maxLongitud;
70      }
71
72     @Override
73     public String toString(){
74       String cadena="";
75
76       if (flag == true){
77
78          //mientras la posicion de cabeza no sea mayor que la posicion del fin de la cola
79          if (cabeza <= fin){
80              for(int i = cabeza; i <= fin; i++){
81                  if (datos[i].getClass().getName().equals("Ejercicio03.Persona")){
82                      Persona temp = (Persona)datos[i];
83                      cadena = cadena + temp.getNombre() + " " + temp.getEdad() + " ";
84                  }
85                  else cadena = cadena + datos[i] + " ";
86              }
87          }
88
89          //como es una cola circular se puede dar el caso de que el fin de la cola
90          //sea una posicion menor que el inicio de la cola
91          //debemos leer desde el inicio de la cola hasta el final del array
92          //y posteriormente desde el incicio del array hasta el fin de la colas
93          else {
94              for(int i = cabeza; i < maxLongitud; i++){
95                  if (datos[i].getClass().getName().equals("Ejercicio03.Persona")){
96                      Persona temp = (Persona)datos[i];
97                      cadena = cadena + temp.getNombre() + " " + temp.getEdad() + " ";
98                  }
99                  else cadena = cadena + datos[i] + " ";
100             }
101             for (int i = 0; i <= fin; i++){
102                 if (datos[i].getClass().getName().equals("Ejercicio03.Persona")){
```

```
103                     Persona temp = (Persona)datos[i];
104                     cadena = cadena + temp.getNombre() + " " + temp.getEdad() + " ";
105                 }
106                 else cadena = cadena + datos[i] + " ";
107             }
108         }
109     }
110     return cadena;
111
112
113   }
114
115
116
117 }
118
119
120
```