La protectora de animales sin ánimo de lucro “Peluditos” quiere encargar una aplicación para gestionar los animales que tiene en acogida. Actualmente sólo acoge a perros y a gatos. De todos los animales debe guardar la siguiente información: identificador, que será un número asignado por el programa de forma que no pueda haber dos iguales, nombre, edad en meses y estado, que será un booleano que indique si el animal está reservado para su adopción o no. Se creará una clase para los perros y otra para los gatos. El programa principal contará como atributo con una lista de todos los animales que tiene la protectora y nos mostrará un menú con las siguientes funciones: 1. Añadir animal: Preguntará si se va a añadir un perro o a un gato, pedirá sus datos y lo añadirá a la lista de animales. 2. Mostrar todos los animales: mostrará un listado de todos los animales indicando para cada uno de ellos si se trata de un perro o de un gato y si está reservado o no. Se dará la opción de volcar dicho listado a un fichero de texto cuyo nombre será proporcionado por el usuario. 3. Mostrar listado de perros sin reserva: mostrará todos los perros que no están reservados 4. Mostrar listado de gatos sin reserva: mostrará todos los gatos que no están reservados 5. Reservar: se preguntará por el identificador del animal que se desea reservar y se cambiará su estado a reservado 6. Anular reserva: se preguntará por el identificador del animal del cuál se desea anular la reserva y se cambiará su estado a no reservado 7. Adoptar: se preguntará por el identificador del animal a adoptar y si existe, se elimina de la lista. Sólo podrán adoptarse animales que se encuentren en estado de reserva. Si el identificador no se corresponde con ningún animal o éste no está reservado deberá indicarlo y no lo eliminará de la lista. 8. Salir Al finalizar la ejecución del programa se deberá guardar en un fichero binario llamado “peluditos.dat” el listado de animales que estén actualmente en el sistema. Al iniciar el programa se leerá el fichero y recuperará el listado de animales previamente guardado. Cuando no exista el fichero el listado estará vacío. Para aprobar el examen será imprescindible que éste no tenga errores de compilación. Se deberán controlar todas las posibles excepciones que pudiesen tener lugar.

MAIN JAVA

```
* To change this license header, choose License Headers in Project
Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package peluditos;

import java.io.*;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.Scanner;

/**
 *
 * @author anusk
 */
public class Peluditos {


    public static ArrayList<Animal> animales = new ArrayList<Animal>();
    public static final String FICHERO = "peluditos.dat";

    public static int leerInt()
    {
        Scanner sc = new Scanner(System.in);
        while (true)
        {
            try
            {
                return sc.nextInt();
            }
            catch (Exception e)
            {
                System.out.println("NÃºmero no vÃ¡lido");
                sc.nextLine();
            }
        }
    }

    public static int menu()
    {
        int op = 0;
        while (op<1 || op>8)
        {
            System.out.println("1. AÃ±adir animal");
            System.out.println("2. Mostrar todos los animales");
            System.out.println("3. Mostrar listado de perros sin reserva");
            System.out.println("4. Mostrar listado de gatos sin reserva");
            System.out.println("5. Reservar");
            System.out.println("6. Anular reserva");
            System.out.println("7. Adoptar");
            System.out.println("8. Salir");
            op = leerInt();
        }
        return op;
```

```
    }

    public static void anadirAnimal()
    {
        Scanner sc = new Scanner(System.in);
        int tipo = 0;
        while (tipo!=1 && tipo!=2)
        {
            System.out.println("Â¿QuÃ© animal desea aÃ±adir?\n1. Perro\n2.
Gato");
            tipo = leerInt();
        }
        System.out.print("Nombre: ");
        String nombre = sc.nextLine();
        System.out.print("Edad (en meses): ");
        int edad = leerInt();
        if (tipo==1)
            animales.add(new Perro(nombre,edad));
        else
            animales.add( new Gato(nombre,edad));
    }

    public static void guardarListadoEnFicheroTexto()
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("Nombre del fichero: ");
        String fichero = sc.nextLine();
        BufferedWriter bw = null;
        try        {
            bw = new BufferedWriter(new FileWriter(fichero));
            Iterator <Animal> it = animales.iterator();
            while (it.hasNext())
            {
                bw.write(it.next().toString());
                bw.newLine();

            }
        }
        catch (IOException e)
        {
            System.out.println("Error escribiendo en el fichero de texto");
        }
        finally
        {
            try
            {
                if (bw!=null)
                    bw.close();
            }
            catch (IOException e)
            {
                System.out.println("Error cerrando el fichero");
            }
        }

    }

    public static void mostrarAnimales()
    {
```

```
        Iterator <Animal> it = animales.iterator();
        while (it.hasNext())
        {
            System.out.println(it.next());
        }
        int op = 0;
        while (op!=1 && op!=2)
        {
            System.out.println("Â¿Desea volcar el listado a un fichero de
texto? \n1. SÃ-\n2. No");
            op = leerInt();
        }
        if (op==1)
            guardarListadoEnFicheroTexto();
    }

    public static void mostrarPerrosSinReserva()
    {
        Iterator <Animal> it = animales.iterator();
        while (it.hasNext())
        {
            Animal a = it.next();
            if (a instanceof Perro && !a.isReservado())
                System.out.println(a);
        }
    }

    public static void mostrarGatosSinReserva()
    {
        Iterator <Animal> it = animales.iterator();
        while (it.hasNext())
        {
            Animal a = it.next();
            if (a instanceof Gato && !a.isReservado())
                System.out.println(a);
        }
    }

    public static Animal buscar(int id)
    {
        Iterator <Animal> it = animales.iterator();
        while (it.hasNext())
        {
            Animal a = it.next();
            if (a.getId() == id)
                return a;
        }
        return null;
    }

    public static void reservar()
    {
        System.out.print("Identificador: ");
        int id = leerInt();
        Animal a = buscar(id);
        if (a == null)
            System.out.println("Animal no encontrado");
        else if (a.isReservado())
```

```
            System.out.println("No se puede reservar un animal ya
reservado");
        else
        {
            a.reservar();
            System.out.println("Animal reservado correctamente");
        }
    }

    public static void anulaReserva()
    {
        System.out.print("Identificador: ");
        int id = leerInt();
        Animal a = buscar(id);
        if (a == null)
            System.out.println("Animal no encontrado");
        else if (a.isReservado())
        {
            a.cancelarReserva();
            System.out.println("Reserva anulada");
        }
        else
            System.out.println("El animal no estaba reservado");
    }

    public static void adoptar()
    {
        System.out.print("Identificador: ");
        int id = leerInt();
        Animal a = buscar(id);
        if (a == null)
            System.out.println("Animal no encontrado");
        else if (a.isReservado())
        {
            animales.remove(a);
            System.out.println("El animal ha sido adoptado");
        }
        else
            System.out.println("El animal no estÃ¡ reservado y no se puede
adoptar");
    }

    public static void guardarDatos()
    {
        ObjectOutputStream oos = null;
        try
        {
            oos = new ObjectOutputStream(new FileOutputStream(FICHERO));
            oos.writeObject(animales);
            oos.writeInt(Animal.contador);
        }
        catch (IOException e)
        {
            System.out.println("Error guardando los datos");
        }
        finally
        {
            try
            {
```

```
            if (oos!=null)
                oos.close();
        }
        catch (IOException e)
        {
            System.out.println("Error cerrando el fichero");
        }
    }
}

public static void recuperarDatos()
{
    ObjectInputStream ois = null;
    try
    {
        ois = new ObjectInputStream(new FileInputStream(FICHERO));
        animales = (ArrayList<Animal>) ois.readObject();
        Animal.setContador(ois.readInt());
    }
    catch (ClassNotFoundException e)
    {
        System.out.println("Error en los datos");
    }
    catch (FileNotFoundException e)
    {
        System.out.println("Ejecutando por primera vez el programa");
    }
    catch (IOException e)
    {
        System.out.println("Error leyendo los datos");
    }
    catch (Exception e)
    {
        System.out.println("Error en los datos");
    }
    finally
    {
        try
        {
            if (ois!=null)
                ois.close();
        }
        catch (IOException e)
        {
            System.out.println("Error cerrando el fichero");
        }
    }
}

public static void main(String[] args) {

    recuperarDatos();
    int op = menu();

    while (op!=8)
    {
        switch (op)
        {
            case 1: anadirAnimal(); break;
```

```
                case 2: mostrarAnimales(); break;
                case 3: mostrarPerrosSinReserva(); break;
                case 4: mostrarGatosSinReserva(); break;
                case 5: reservar(); break;
                case 6: anulaReserva(); break;
                case 7: adoptar(); break;
            }

            op = menu();
        }
        guardarDatos();

    }

}
```

ANIMAL.JAVQ

```
ackage peluditos;

import java.io.Serializable;

/**
 *
 * @author anusk
 */
public abstract class Animal implements Serializable{

    protected int id;
    protected String nombre;
    protected int edad;
    protected boolean reservado;

    protected static int contador = 1;

    public Animal(String n, int e)
    {
        id = contador++;
        nombre = n;
        edad = e;
        reservado = false;
    }

    public int getId() {
        return id;
    }

    public boolean isReservado() {
        return reservado;
    }

    public void setReservado(boolean reservado) {
        this.reservado = reservado;
    }

    public void reservar() {
        this.reservado = true;
    }

    public static int getContador() {
        return contador;
    }

    public static void setContador(int contador) {
        Animal.contador = contador;
    }

    public void cancelarReserva() {
        this.reservado = false;
    }

    @Override
```

```
    public String toString()
    {
        return id+"\t"+nombre+"\t"+edad+"
meses\t"+((reservado)?"Reservado":"No reservado");
    }

}
```

```java
package peluditos;

/**
 *
 * @author anusk
 */
public class Gato extends Animal{
    public Gato(String n, int e)
    {
        super(n,e);
    }

    @Override
    public String toString()
    {
        return "Gato: "+super.toString();
    }

}
```

```java
*
 * @author anusk
 */
public class Perro extends Animal{
    public Perro(String n, int e)
    {
        super(n,e);
    }

    @Override
    public String toString()
    {
        return "Perro: "+super.toString();
    }

}
```