CLASES:

- Main
- MenuPrincipal
- BotonesMenu (ENUM)
- ParqueMovil
- Vehiculo
- Marca (ENUM)
- Utilidades

Main:

```
public class Main {
    public static void main(String[] args) {

        ParqueMovil parqueMovil = new ParqueMovil();

        MenuPrincipal menuPrincipal = new MenuPrincipal();

        menuPrincipal.iniciarMenu(parqueMovil);
        menuPrincipal.ejecutarMenuPincipal(parqueMovil);
    }
}
```

MenuPrincipal:

```
import javax.swing.*;

public class MenuPrincipal {
    private boolean comprobarSalir = false;

    public MenuPrincipal() {
    }

    public void iniciarMenu(ParqueMovil parqueMovil) {
        if (JOptionPane.showConfirmDialog(null, "Desea inicializar el
parque con 5 vehiculos?", null, JOptionPane.YES_NO_OPTION) == 0) {
            parqueMovil.ParqueMovilPorDefecto();
            parqueMovil.ordenaParquePorMatricula();
            parqueMovil.mostrarListado();
        }
    }

    public void ejecutarMenuPincipal(ParqueMovil parqueMovil) {
        BotonesMenu presionado;

        do {
            int opcionElegida = JOptionPane.showOptionDialog(null, "Elige
una opcion", "MENU", JOptionPane.DEFAULT_OPTION,
JOptionPane.QUESTION_MESSAGE, null, BotonesMenu.values(),
BotonesMenu.ALTA);
```

```
            if (opcionElegida < 0) {
                presionado = BotonesMenu.SALIR;
            } else {
                presionado = BotonesMenu.values()[opcionElegida];
            }

            switch (presionado) {
                case ALTA:
                    parqueMovil.anyadeVehiculo();
                    parqueMovil.ordenaParquePorMatricula();
                    break;
                case BAJA:
                    parqueMovil.borrarVehiculo();
                    break;
                case DATOS:
                    parqueMovil.mostrarDatosVehiculoSeleccionado();
                    break;
                case LISTADO:
                    parqueMovil.mostrarListado();
                    break;
                case SALIR:
                    comprobarSalir = Utilidades.comprobarSalir();
                default:
                    break;
            }
        } while (!comprobarSalir);
    }
}
```

BotonesMenu:

```
public enum BotonesMenu {
    ALTA("ALTA DE VEHICULO"),
    BAJA("BAJA DE VEHICULO"),
    DATOS("DATOS DEL VEHICULO SELECCIONADO"),
    LISTADO("LISTA DE VEHICULOS EN PARQUE"),
    SALIR("SALIR");

    String nombre;

    BotonesMenu(String nombre) {
        this.nombre = nombre;
    }
}
```

ParqueMovil:

```
import javax.swing.*;
import java.util.Arrays;

public class ParqueMovil {
```

```
    private static final int MAX_VEHICULOS_EN_PARQUE = 50;
    private static final int VEHICULOS_POR_DEFECTO = 5;
    private static final String MENSAJE_PARQUE_VACIO = "No hay vehículos
en el parque";
    private Vehiculo[] parque = new Vehiculo[MAX_VEHICULOS_EN_PARQUE];
    private static int cantidadVehiculos = 0;


    public ParqueMovil() {
    }

    public void ParqueMovilPorDefecto() {
        this.parque = Vehiculo.porDefecto();
        cantidadVehiculos = VEHICULOS_POR_DEFECTO;
    }

    public Vehiculo[] parqueMovilRecortado(Vehiculo[] parque) {
        Vehiculo[] parqueMovilRecortado;
        parqueMovilRecortado = Arrays.copyOf(parque, cantidadVehiculos);
        return parqueMovilRecortado;
    }

    public int getCantidadVehiculos() {
        return cantidadVehiculos;
    }

    @Override
    public String toString() {
        String salida = String.format("%10s %10s %10s %10s", "MATRÍCULA",
"MARCA", "CARGA", "KM") + "\n";
        for (Vehiculo b :
                this.parque) {
            salida += b + "\n";
        }
        return salida;
    }

    private boolean existeMatriculaEnParque(Vehiculo v) {
        if (v == null) {
            return false;
        }
        int posicionMatricula =
Utilidades.buscarPosicionEnArray(this.parque, v.getMatricula());
        if (posicionMatricula >= 0) {
            return true;
        }
        return false;
    }

    public boolean anyadeVehiculo() {
```

```
        if (this.getCantidadVehiculos() >= MAX_VEHICULOS_EN_PARQUE) {
            JOptionPane.showMessageDialog(null, "El parque esta lleno");
            return false;
        }

        Vehiculo v = Utilidades.entradaVehiculo();
        boolean existeVehiculo = existeMatriculaEnParque(v);
        if (v == null) {
            JOptionPane.showMessageDialog(null, "Entrada cancelada");
            return false;
        } else if (existeVehiculo) {
            JOptionPane.showMessageDialog(null, "La matrícula introducida
ya existe");
            return false;
        } else {
            int posicionLibre =
Utilidades.buscaPosicionVacia(this.parque);
            this.parque[posicionLibre] = v;
            JOptionPane.showMessageDialog(null, "Vehículo añadido con
éxito");
            cantidadVehiculos++;
            return true;
        }
    }

    public boolean borrarVehiculo() {
        if (getCantidadVehiculos() <= 0) {
            JOptionPane.showMessageDialog(null, MENSAJE_PARQUE_VACIO);
            return false;
        }

        Vehiculo vehiculoABorrar = (Vehiculo)
JOptionPane.showInputDialog(null, "Selecciona un vehículo", "MATRÍCULA",
JOptionPane.DEFAULT_OPTION, null, parqueMovilRecortado(this.parque),
parqueMovilRecortado(this.parque)[0]);
        if (vehiculoABorrar == null) {
            JOptionPane.showMessageDialog(null, "Borrado cancelado");
            return false;
        } else {
            String matricula = vehiculoABorrar.getMatricula();
            int posicionVehiculoBorrar =
Utilidades.buscarPosicionEnArray(parque, matricula);
            for (int i = posicionVehiculoBorrar; i < parque.length - 1;
i++) {
                parque[i] = parque[i + 1];
            }
            cantidadVehiculos--;
            JOptionPane.showMessageDialog(null, "Vehículo borrado con
éxito");
            return true;
        }
```

```
    }

    private String obtenListaVehiculos() {
        Vehiculo[] v = parqueMovilRecortado(this.parque);

        String salida = "";
        salida += "<HTML><h2>LISTADO DE VEHÍCULOS:</h2>";
        salida += "<hr/>";
        salida += "<pre style=\"font-size:130%\">";
        salida += String.format("%10s %10s %10s %5s", "MATRÍCULA",
"MARCA", "CARGA", "KM");
        salida += "<hr/>";
        for (int i = 0; i < cantidadVehiculos; i++) {
            salida += Utilidades.salidaVehiculo(v[i]);
            salida += "<br/>";
        }
        salida += "</pre><hr/></html>";
        return salida;
    }

    public boolean mostrarListado() {
        if (cantidadVehiculos <= 0) {
            JOptionPane.showMessageDialog(null, MENSAJE_PARQUE_VACIO);
            return false;
        } else {
            JOptionPane.showMessageDialog(null, obtenListaVehiculos());
            return true;
        }
    }

    public boolean mostrarDatosVehiculoSeleccionado() {
        if (cantidadVehiculos <= 0) {
            JOptionPane.showMessageDialog(null, MENSAJE_PARQUE_VACIO);
            return false;
        }
        Vehiculo seleccionado = (Vehiculo)
JOptionPane.showInputDialog(null, "Selecciona una Vehículo", null,
JOptionPane.DEFAULT_OPTION, null, parqueMovilRecortado(this.parque),
parqueMovilRecortado(this.parque)[0]);
        if (seleccionado == null) {
            return false;
        } else {
            JOptionPane.showMessageDialog(null,
seleccionado.datosVehiculo());
            return true;
        }
    }

    public void ordenaParquePorMatricula() {
        Utilidades.ordenaArray(this.parque,cantidadVehiculos);
    }
```

```
}
```

Vehiculo:

```
public class Vehiculo {

    private final String matricula; //AAA-9999
    private final Marca marca;
    private int cargaMaxima; //kilo
    private int kilometraje; //km

    public Vehiculo(String matricula, Marca marca, int cargaMaxima, int
kilometraje) {
        this.matricula = matricula;
        this.marca = marca;
        this.cargaMaxima = cargaMaxima;
        this.kilometraje = kilometraje;
    }

    public Vehiculo(String matricula, Marca marca, int cargaMaxima) {
        this(matricula, marca, cargaMaxima, 0);
    }

    public String getMatricula() {
        return this.matricula;
    }

    public Marca getMarca() {
        return this.marca;
    }

    public int getCargaMaxima() {
        return this.cargaMaxima;
    }

    public int getKilometraje() {
        return this.kilometraje;
    }

    public void setCargaMaxima(int cargaMaxima) {
        this.cargaMaxima = cargaMaxima;
    }

    public void setKilometraje(int kilometraje) {
        this.kilometraje = kilometraje;
    }

    public String datosVehiculo() {
        return String.format("%10s %10s %8dKg %5dKm", this.matricula,
this.marca, this.cargaMaxima, this.kilometraje);
    }
```

```
    @Override
    public String toString() {
        return this.matricula;
    }

    public static final Vehiculo[] porDefecto() {
        Vehiculo[] porDefecto = new Vehiculo[50];
        porDefecto[0] = new Vehiculo("2222-BBB", Marca.MERCEDES, 5000);
        porDefecto[1] = new Vehiculo("1111-AAA", Marca.FORD, 3500);
        porDefecto[2] = new Vehiculo("5555-EEE", Marca.FORD, 1000);
        porDefecto[3] = new Vehiculo("4444-DDD", Marca.MERCEDES, 3500);
        porDefecto[4] = new Vehiculo("3333-CCC", Marca.PEGASO, 4000);
        return porDefecto;
    }

    // pruebas
    public static void main(String[] args) {
        Vehiculo[] v = Vehiculo.porDefecto();
        for (Vehiculo b :
                v) {
            System.out.println(b);
        }
    }
}
```

Marca:

```
public enum Marca {
    MERCEDES("Mercedes"), FORD("Ford"), PEGASO("Pegaso");
    private final String cadenaTipo;

    Marca(String nombre) {
        this.cadenaTipo = nombre;
    }

    @Override
    public String toString() {
        return cadenaTipo;
    }
}
```

Utilidades:

```
import javax.swing.*;

public class Utilidades {
    public Utilidades() {
    }

    public static int buscaPosicionVacia(Vehiculo[] v) {
```

```
        for (int i = 0; i < v.length - 1; i++) {
            if (v[i] == null) {
                return i;
            }
        }
        return -1;
    }

    public static int buscarPosicionEnArray(Vehiculo[] v, String
matricula) {
        for (int i = 0; i < v.length; i++) {
            if (v[i] == null) {
                return -1;
            }
            if (v[i].getMatricula().equals(matricula)) {
                return i;
            }
        }
        return -1;
    }

    public static Vehiculo entradaVehiculo() {
        String matricula = JOptionPane.showInputDialog(null, "Introduzca
una matricula con el siguiente formato 1111-XXX");
        if (matricula == null) {
            return null;
        } else {
            Marca marca = (Marca) JOptionPane.showInputDialog(null,
"Seleccione una marca", "MARCA", 0, null, Marca.values(),
Marca.MERCEDES);
            if (marca == null) {
                return null;
            } else {
                String cargaString = JOptionPane.showInputDialog(null,
"Carga Maxima:");
                if (cargaString == null) {
                    return null;
                } else {
                    int cargaMaxima = Integer.parseInt(cargaString);
                    Vehiculo v = new Vehiculo(matricula, marca,
cargaMaxima);
                    return v;
                }
            }
        }

    }

    public static String salidaVehiculo(Vehiculo v) {
        String salida = "";
```

```
        salida += "<hr/>";
        salida += v.datosVehiculo();

        return salida;
    }

    public static void ordenaArray(Vehiculo[] v, int longitudOrdenacion)
{
        int menor;
        Vehiculo aux;

        for (int i = 0; i < longitudOrdenacion - 1; i++) {
            menor = i;
            for (int j = i + 1; j < longitudOrdenacion; j++) {
                if
(v[j].getMatricula().compareTo((v[menor].getMatricula())) < 0) {
                    menor = j;
                }
            }
            if (menor != i) {
                aux = v[menor];
                v[menor] = v[i];
                v[i] = aux;
            }
        }
    }

    public static boolean comprobarSalir() {
        int respuesta = JOptionPane.showConfirmDialog(null, "¿Esta
seguro?", "Alerta!", JOptionPane.YES_NO_OPTION);
        if (respuesta == 0) {
            return true;
        }
        return false;
    }
}
```