```
package Examen.QuienNoCorreVuela;

public class Administrador extends Usuario {

	public Administrador(String u, String p) {
		super(u, p);
	}

}
```

```
package Examen.QuienNoCorreVuela;

import java.io.Serializable;

public class Aeropuerto implements Serializable {

	private String codigo;
	private String nombre;
	private String ciudad;

	public Aeropuerto(String cod, String n, String ciu) {
		codigo = cod;
		nombre = n;
		ciudad = ciu;
	}

	public String getCodigo() {
		return codigo;
	}

	public String toString() {
		return codigo + " - " + nombre + " - " + ciudad;
	}

}
```

```
package Examen.QuienNoCorreVuela;

public class Agente extends Usuario {

	public Agente(String u, String p) {
		super(u, p);
	}

}
```

```
package Examen.QuienNoCorreVuela;

import java.io.Serializable;

public class Fecha implements Serializable {
	private int dia;
	private int mes;
	private int anio;

	public Fecha(int d, int m, int a) {
		dia = d;
		mes = m;
		anio = a;
	}

	public int comparar(Fecha f) {
		if (anio < f.anio)
			return -1;
		else if (anio > f.anio)
			return 1;
		else if (mes < f.mes)
			return -1;
		else if (mes > f.mes)
			return 1;
		else if (dia < f.dia)
			return -1;
		else if (dia > f.dia)
			return 1;
		else
			return 0;
	}

	public String toString() {
		String texto = "";
		if (dia < 10)
			texto += "0" + dia + "/";
		else
			texto += dia + "/";
		if (mes < 10)
			texto += "0" + mes + "/" + anio;
		else
			texto += mes + "/" + anio;

		return texto;
	}

}
```

```java
package Examen.QuienNoCorreVuela;

import java.util.Scanner;

public class Lectura {

	public static int leerInt() {
		Scanner sc = new Scanner(System.in);

		while (true) {
			try {
				int n = sc.nextInt();
				return n;
			} catch (Exception e) {
				System.out.println("Debe escribir un número entero");
				sc.nextLine();
			}
		}
	}

	public static double leerDouble() {
		Scanner sc = new Scanner(System.in);

		while (true) {
			try {
				double n = sc.nextDouble();
				return n;
			} catch (Exception e) {
				System.out.println("Debe escribir un número double");
				sc.nextLine();
			}
		}
	}

	public static String leerLinea() {
		Scanner sc = new Scanner(System.in);

		return sc.nextLine();
	}

}
```

```
package Examen.QuienNoCorreVuela;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;

public class Principal {

	public static String fDatos = "datos.bin";
	public static QuienNoCorreVuela gestor = new QuienNoCorreVuela();

	public static void recuperarDatos() {
		ObjectInputStream ois = null;
		File f = new File(fDatos);
		if (f.exists()) {
			try {
				ois = new ObjectInputStream(new FileInputStream(fDatos));
				gestor = (QuienNoCorreVuela) ois.readObject();
			} catch (IOException e) {
				System.err.println("Error leyendo el fichero");
			} catch (ClassNotFoundException e) {
				System.err.println("Error leyendo los datos");
			} finally {
				try {
					if (ois != null)
						ois.close();

				} catch (IOException e) {
					System.err.println("Error cerrando el fichero");
				}
			}

		} else {
			gestor.altaAdmin();
			System.out.println("Ahora debe escribir su usuario y contraseña");
		}
		gestor.cambiarUsuario();
		gestor.menu();
	}

	public static void guardarDatos() {
		ObjectOutputStream oos = null;

		try {
			oos = new ObjectOutputStream(new FileOutputStream(fDatos));
			oos.writeObject(gestor);
```

```
            } catch (IOException e) {
                System.err.println("Error escribiendo en el fichero");
            }

            finally {
                try {
                    if (oos != null)
                        oos.close();

                } catch (IOException e) {
                    System.err.println("Error cerrando el fichero");
                }
            }

        }

        public static void main(String[] args) {

            recuperarDatos();
            guardarDatos();

        }

    }
```

```
package Examen.QuienNoCorreVuela;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.Iterator;

public class QuienNoCorreVuela implements Serializable {

	private ArrayList<Usuario> usuarios = new ArrayList<Usuario>();
	private ArrayList<Aeropuerto> aeropuertos = new ArrayList<Aeropuerto>();
	private ArrayList<Vuelo> vuelos = new ArrayList<Vuelo>();
	private ArrayList<Reserva> reservas = new ArrayList<Reserva>();

	private Usuario userActual = null;

	public QuienNoCorreVuela() {
	}

	public boolean menuAdmin() {
		int op = 0;
		boolean salir = false;
		while (op < 1 || op > 6) {
			System.out.println("MENU ADMINISTRADOR");
			System.out.println("1. Dar de alta aeropuerto");
			System.out.println("2. Dar de alta administrador");
			System.out
					.println("3. Dar de alta responsable de compañía aérea");
			System.out.println("4. Dar de alta agente de viajes");
			System.out.println("5. Cambiar de usuario");
			System.out.println("6. Salir");
			op = Lectura.leerInt();
		}
		switch (op) {
			case 1 :
				altaAeropuerto();
				break;
			case 2 :
				altaAdmin();
				break;
			case 3 :
				altaResponsable();
				break;
			case 4 :
				altaAgente();
				break;
			case 5 :
				cambiarUsuario();
				break;
			case 6 :
```

```
                salir = true;

        }
        return salir;

    }

    public boolean menuResponsable() {
        boolean salir = false;
        int op = 0;
        while (op < 1 || op > 6) {
            System.out.println("MENU RESPONSABLE");
            System.out.println("1. Añadir vuelo");
            System.out.println("2. Cancelar vuelo");
            System.out.println("3. Modificar hora de un vuelo");
            System.out.println("4. Cambiar de usuario");
            System.out.println("5. Consultar aeropuertos");
            System.out.println("6. Salir");
            op = Lectura.leerInt();
        }
        switch (op) {
            case 1 :
                anadirVuelo();
                break;
            case 2 :
                cancelarVuelo();
                break;
            case 3 :
                modificarVuelo();
                break;
            case 4 :
                cambiarUsuario();
                break;
            case 5 :
                consultarAeropuertos();
                break;
            case 6 :
                salir = true;
        }

        return salir;
    }

    public boolean menuAgente() {
        boolean salir = false;
        int op = 0;
        while (op < 1 || op > 8) {
            System.out.println("MENU AGENTE");
            System.out.println("1. Realizar búsqueda");
```

```
            System.out.println("2. Reservar vuelo");
            System.out.println("3. Cancelar reserva");
            System.out.println("4. Listar todas las reservas");
            System.out.println("5. Listar por viajero");
            System.out.println("6. Cambiar de usuario");
            System.out.println("7. Consultar aeropuertos");
            System.out.println("8. Salir");
            op = Lectura.leerInt();
        }
        switch (op) {
            case 1 :
                realizarBusqueda();
                break;
            case 2 :
                reservarVuelo();
                break;
            case 3 :
                cancelarReserva();
                break;
            case 4 :
                listarReservas();
                break;
            case 5 :
                listarPorViajero();
                break;
            case 6 :
                cambiarUsuario();
                break;
            case 7 :
                consultarAeropuertos();
                break;
            case 8 :
                salir = true;
        }

        return salir;
    }

    public void menu() {
        boolean salir = false;

        while (!salir) {
            if (userActual instanceof Administrador)
                salir = menuAdmin();
            else if (userActual instanceof Agente)
                salir = menuAgente();
            else
                salir = menuResponsable();
        }
```

```
}

public Usuario buscarUsuario(String n) {

    boolean encontrado = false;

    Iterator<Usuario> it = usuarios.iterator();
    while (it.hasNext()) {
        Usuario u = it.next();
        if (u.getUser().equals(n)) {
            return u;
        }
    }
    return null;

}

public void cambiarUsuario() {
    System.out.println("Usuario: ");
    String user = Lectura.leerLinea();
    System.out.println("Password: ");
    String pass = Lectura.leerLinea();

    Usuario u = buscarUsuario(user);
    if (u != null && u.getPass().equals(pass)) {
        System.out.println("Se ha conectado correctamente");
        userActual = u;
    } else
        System.out.println("El usuario/password son incorrectos");

}

public void altaAeropuerto() {
    System.out.println("Código: ");
    String cod = Lectura.leerLinea();
    System.out.println("Nombre: ");
    String nom = Lectura.leerLinea();
    System.out.println("Ciudad: ");
    String ciud = Lectura.leerLinea();

    aeropuertos.add(new Aeropuerto(cod, nom, ciud));

}

public void consultarAeropuertos() {
    Iterator<Aeropuerto> it = aeropuertos.iterator();
    while (it.hasNext())
        System.out.println(it.next());
}
```

```
public void altaAdmin() {
    System.out.println("Usuario: ");
    String user = Lectura.leerLinea();
    Usuario u = buscarUsuario(user);
    while (u != null) {
        System.out.println("Nombre de usuario existente, elija otro");
        System.out.println("Usuario: ");
        user = Lectura.leerLinea();
        u = buscarUsuario(user);
    }
    System.out.println("Password: ");
    String pass = Lectura.leerLinea();
    usuarios.add(new Administrador(user, pass));

}

public void altaAgente() {
    System.out.println("Usuario: ");
    String user = Lectura.leerLinea();
    Usuario u = buscarUsuario(user);
    while (u != null) {
        System.out.println("Nombre de usuario existente, elija otro");
        System.out.println("Usuario: ");
        user = Lectura.leerLinea();
        u = buscarUsuario(user);
    }
    System.out.println("Password: ");
    String pass = Lectura.leerLinea();
    usuarios.add(new Agente(user, pass));

}

public void altaResponsable() {
    System.out.println("Usuario: ");
    String user = Lectura.leerLinea();
    Usuario u = buscarUsuario(user);
    while (u != null) {
        System.out.println("Nombre de usuario existente, elija otro");
        System.out.println("Usuario: ");
        user = Lectura.leerLinea();
        u = buscarUsuario(user);
    }
    System.out.println("Password: ");
    String pass = Lectura.leerLinea();
    usuarios.add(new Responsable(user, pass));

}
```

```
public Aeropuerto buscarAeropuerto(String cod) {

	Iterator<Aeropuerto> it = aeropuertos.iterator();
	while (it.hasNext()) {
		Aeropuerto a = it.next();
		if (a.getCodigo().equals(cod)) {
			return a;
		}
	}
	return null;
}

public void anadirVuelo() {

	System.out.println("Código de vuelo");
	String cod = Lectura.leerLinea();
	System.out.println("Día: ");
	int dia = Lectura.leerInt();
	System.out.println("Mes: ");
	int mes = Lectura.leerInt();
	System.out.println("Año: ");
	int anio = Lectura.leerInt();
	System.out.println("Hora: ");
	String hora = Lectura.leerLinea();
	System.out.println("Código del aeropuerto de origen: ");
	String origen = Lectura.leerLinea();
	Aeropuerto or = buscarAeropuerto(origen);
	while (or == null) {
		System.out.println("Código no válido, vuelva a escribirlo: ");
		origen = Lectura.leerLinea();
		or = buscarAeropuerto(origen);
	}
	System.out.println("Código del aeropuerto de destino: ");
	String destino = Lectura.leerLinea();
	Aeropuerto dest = buscarAeropuerto(destino);
	while (dest == null) {
		System.out.println("Código no válido, vuelva a escribirlo: ");
		destino = Lectura.leerLinea();
		dest = buscarAeropuerto(destino);
	}
	System.out.println("Duración en minutos: ");
	int duracion = Lectura.leerInt();
	System.out.println("Plazas: ");
	int plazas = Lectura.leerInt();

	vuelos.add(new Vuelo(cod, new Fecha(dia, mes, anio), hora, or, dest,
			duracion, plazas));

}
```

```
public void cancelarVuelo() {
    System.out.println("Indique el código del vuelo");
    String cod = Lectura.leerLinea();

    Iterator<Vuelo> it = vuelos.iterator();
    while (it.hasNext()) {
        Vuelo v = it.next();
        if (v.getCodigo().equals(cod)) {
            v.cancelar();
        }
    }
}

public void modificarVuelo() {
    System.out.println("Indique el código del vuelo");
    String cod = Lectura.leerLinea();
    System.out.println("Indique la nueva hora");
    String hora = Lectura.leerLinea();

    Iterator<Vuelo> it = vuelos.iterator();
    while (it.hasNext()) {
        Vuelo v = it.next();
        if (v.getCodigo().equals(cod)) {
            v.setHora(hora);
        }
    }
}

public void realizarBusqueda() {
    System.out.println("Código del aeropuerto de origen: ");
    String origen = Lectura.leerLinea();
    Aeropuerto or = buscarAeropuerto(origen);
    while (or == null) {
        System.out.println("Código no válido, vuelva a escribirlo: ");
        origen = Lectura.leerLinea();
        or = buscarAeropuerto(origen);
    }
    System.out.println("Código del aeropuerto de destino: ");
    String destino = Lectura.leerLinea();
    Aeropuerto dest = buscarAeropuerto(destino);
    while (dest == null) {
        System.out.println("Código no válido, vuelva a escribirlo: ");
        destino = Lectura.leerLinea();
        dest = buscarAeropuerto(destino);
    }
    System.out.println("Fecha inicial");
    System.out.println("Día: ");
    int diaI = Lectura.leerInt();
```

```
        System.out.println("Mes: ");
        int mesI = Lectura.leerInt();
        System.out.println("Año: ");
        int anioI = Lectura.leerInt();
        Fecha inicial = new Fecha(diaI, mesI, anioI);
        System.out.println("Fecha final");
        System.out.println("Día: ");
        int diaF = Lectura.leerInt();
        System.out.println("Mes: ");
        int mesF = Lectura.leerInt();
        System.out.println("Año: ");
        int anioF = Lectura.leerInt();
        Fecha Ffinal = new Fecha(diaF, mesF, anioF);
        Iterator<Vuelo> it = vuelos.iterator();
        while (it.hasNext()) {
            Vuelo v = it.next();
            if (v.getOrigen() == or && v.getDestino() == dest)
                if (v.getSalida().comparar(inicial) == 0
                        || v.getSalida().comparar(inicial) > 0)
                    if (v.getSalida().comparar(Ffinal) == 0
                            || v.getSalida().comparar(Ffinal) < 0) {
                        System.out.println(v);
                    }
        }

    }
    public void reservarVuelo() {
        System.out.println("Código del vuelo:");
        String cod = Lectura.leerLinea();
        Iterator<Vuelo> it = vuelos.iterator();

        Vuelo vuelo = null;
        while (it.hasNext() && vuelo == null) {
            Vuelo v = it.next();
            if (v.getCodigo().equals(cod)) {
                vuelo = v;
            }
        }

        if (vuelo == null)
            System.out.println("El vuelo no existe");
        else if (vuelo.getPlazas() < vuelo.getReservas())
            System.out.println("No hay plazas disponiblñes en ese vuelo");
        else {
            vuelo.setReservas(vuelo.getReservas() + 1);
            System.out.println("Nombre del pasajero:");
            String pasajero = Lectura.leerLinea();
            reservas.add(new Reserva(vuelo, pasajero));
        }
```

```
    }
    public void cancelarReserva() {
        System.out.println("Indique el código de la reserva");
        int cod = Lectura.leerInt();

        Iterator<Reserva> it = reservas.iterator();
        while (it.hasNext()) {
            Reserva r = it.next();
            if (r.getCodigo() == cod) {
                it.remove();;
            }
        }
    }

    public void listarReservas() {
        Iterator<Reserva> it = reservas.iterator();
        while (it.hasNext()) {
            System.out.println(it.next());
        }
    }
    public void listarPorViajero() {
        System.out.println("Indique nombre a buscar");
        String n = Lectura.leerLinea();

        Iterator<Reserva> it = reservas.iterator();
        while (it.hasNext()) {
            Reserva r = it.next();
            if (r.getViajero().toUpperCase().indexOf(n.toUpperCase()) != -1) {
                System.out.println(r);
            }
        }
    }

}
```

```
package Examen.QuienNoCorreVuela;

import java.io.Serializable;

public class Reserva implements Serializable {

	private Vuelo vuelo;
	private String viajero;
	private int codigo;
	private static int num_reser = 0;

	public Reserva(Vuelo vu, String via) {
		codigo = ++num_reser;
		vuelo = vu;
		viajero = via;

	}

	public boolean contiene(String n) {
		return (viajero.toUpperCase().indexOf(n.toUpperCase()) != -1);
	}

	public String toString() {
		return codigo + " - " + vuelo.getCodigo() + " " + vuelo.getSalida()
				+ " " + vuelo.getHora() + " - " + viajero + " - "
				+ ((vuelo.cancelado()) ? "CANCELADO" : "RESERVADO");
	}

	public Vuelo getVuelo() {
		return vuelo;
	}

	public int getCodigo() {
		return codigo;
	}

	public String getViajero() {
		return viajero;
	}

}
```

```
package Examen.QuienNoCorreVuela;

public class Responsable extends Usuario {

	public Responsable(String u, String p) {
		super(u, p);
	}

}
```

```
package Examen.QuienNoCorreVuela;

import java.io.Serializable;

public abstract class Usuario implements Serializable {

	protected String user;
	protected String pass;

	public Usuario(String u, String p) {
		user = u;
		pass = p;
	}

	public String getUser() {
		return user;
	}

	public String getPass() {
		return pass;
	}

}
```

```
package Examen.QuienNoCorreVuela;

import java.io.Serializable;

public class Vuelo implements Serializable {

	private String codigo;
	private Fecha salida;
	private String hora;
	private Aeropuerto origen;
	private Aeropuerto destino;
	private int duracion; // en minutos
	private int plazas;
	private int reservas;
	private String estado = "OPERATIVO";

	public Vuelo(String c, Fecha s, String h, Aeropuerto o, Aeropuerto d,
			int dur, int p) {
		codigo = c;
		salida = s;
		hora = h;
		origen = o;
		destino = d;
		duracion = dur;
		plazas = p;
		reservas = 0;
	}

	public String getHora() {
		return hora;
	}

	public void setHora(String hora) {
		this.hora = hora;
	}

	public int getReservas() {
		return reservas;
	}

	public int getPlazas() {
		return plazas;
	}

	public void setReservas(int reservas) {
		this.reservas = reservas;
	}

	public String getCodigo() {
```

```
        return codigo;
    }

    public Fecha getSalida() {
        return salida;
    }

    public Aeropuerto getOrigen() {
        return origen;
    }

    public Aeropuerto getDestino() {
        return destino;
    }

    public void cancelar() {
        estado = "CANCELADO";
    }

    public boolean cancelado() {
        return estado.equals("CANCELADO");
    }

    public String toString() {
        return "Código: " + codigo + "\nSalida: " + salida + "\nHora: " + hora
                + "\nOrigen: " + origen + "\nDestino: " + destino
                + "\nDuración: " + duracion + "min" + "\nPlazas: " + plazas
                + "\nReservas: " + reservas;
    }

}
```