```java
package etsjavadocs;

/**
 * Esta clase emula los atributos y metodos
 * de un vuelo
 * @author Daniel Castro Cruz
 */
public class vueloJavadocs {

  private int idvuelo;
  private String aerolinea;
  private String origen;
  private String destino;
  private int asientostotales;
  private int asientosdisponibles;
  private boolean escala;
  /*Se establece que el número de asientos
  totales de todo vuelo sea como máximo 800*/

  /**
   * Metodo constructor con parametros
   * @param nidvuelo Id del vuelo
   * @param naerolinea Nombre de la aerolinea
   * @param norigen Nombre del lugar de origen
   * @param ndestino Nombre del lugar de destino
   * @param nasientostotales Numero de asientos totales
   * @param tieneescala Booleano que indica si el vuelo tiene
escala
   */
  public vueloJavadocs(int nidvuelo, String naerolinea, String
norigen,
      String ndestino, int nasientostotales, boolean tieneescala) {
    idvuelo = nidvuelo;
    aerolinea = naerolinea;
    origen = norigen;
    destino = ndestino;
    if (nasientostotales < 0) {
      nasientostotales = 0;
    }
    if (nasientostotales > 800) {
      nasientostotales = 800;
    }
    asientostotales = nasientostotales;
    asientosdisponibles = nasientostotales;
    escala = tieneescala;
  } /* Fin del constructor de Vuelo
  El nuevo valor debe ser superior al que
  había previamente y no debe ser mayor de 800*/
```

```java
/**
* Metodo para ampliar la capacidad del vuelo
* @param nuevovalortotal Numero de la nueva capacidad del vuelo
*/
public void ampliarcapacidad(int nuevovalortotal) {
  if (nuevovalortotal < 800) {
    if (nuevovalortotal > asientostotales) {
      asientostotales = nuevovalortotal;
    } else {
      System.out.println("Debe introducir una "
          + "nueva capacidad mayor a la actual");
    }
  } else {
    System.out.println("La nueva capacidad no puede "
        + "ser superior a 80");
  }
}

/**
* Metodo para modificar si el vuelo tiene escala
* @param nescala Booleano que indica si el vuelo tiene escala
*/
public void modificarescala(boolean nescala) {
  if ((nescala == true) && (escala == false)) {
    escala = true;
  } else if ((nescala == false) && (escala == true)) {
    escala = false;
  } else {
    System.out.println("El valor "
        + "introducido es el existente previamente");
  }
}

/**
* Metodo para reservar asientos del vuelo
* @param cantidad Numero que indica el numero de asientos
* que se reservarán
*/
public void reservarasiento(int cantidad) {
  if (cantidad > 0) {
    if (asientosdisponibles - cantidad >= 0) {
      asientosdisponibles = asientosdisponibles - cantidad;
    } else {
      System.out.println("No hay asientos suficientes");
    }
  } else {
    System.out.println("La cantidad introducida no es valida");
  }
} // Fin del método reservar asiento
```

```java
/**
 * Metodo para cancelar la reserva del vuelo
 * @param cantidad Numero que indica la cantidad de asientos
 * sobre los que se cancela la reserva
 */
public void devolverreserva(int cantidad) {
  if (cantidad > 0) {
    if (asientosdisponibles + cantidad <= asientostotales) {
      asientosdisponibles = asientosdisponibles + cantidad;
    } else {
      System.out.println("La cantidad introducida "
          + "excede el numero de asientos totales");
    }
  } else {
    System.out.println("Debe devolver al menos un asiento");
  }
} // Fin del método devolver reserva

/**
 * Metodo para retornar el id del vuelo
 * @return Retorna el id del vuelo
 */
public int getid() {
  return idvuelo;
}

/**
 * Metodo para retornar el nombre de la aerolinea
 * @return Retorna en nombre de la aerolinea
 */
public String getaerolinea() {
  return aerolinea;
}

/**
 * Metodo para retornar el nombre del lugar
 * de origen del vuelo
 * @return Retorna el lugar de origen del vuelo
 */
public String getorigen() {
  return origen;
}
```

```java
/**
 * Metodo para retornar el nombre del lugar
 * de destino del vuelo
 * @return Retorna el lugar de destino del vuelo
 */
public String getdestino() {
  return destino;
}

/**
 * Metodo para retornar el numero de asientos
 * totales del vuelo
 * @return Retorna el numero de asientos totales
 */
public int getasientostotales() {
  return asientostotales;
}

/**
 * ´Metodo para retornar el numero de asientos
 * disponibles del vuelo
 * @return Retorna en numero de asientos disponibles
 */
public int getasientosdisponibles() {
  return asientosdisponibles;
}

/**
 * Metodo para retornar si el vuelo tiene o no escala
 * @return Retorna si el vuelo tiene o no escala
 */
public boolean getescala() {
  return escala;
}

/**
 * Metodo para modificar el valor del id del vuelo
 * @param nuevoid Numero del nuevo id del vuelo
 */
public void settid(int nuevoid) {
  idvuelo = nuevoid;
}
```

```java
  /**
   * Metodo para modificar el nombre de la aerolinea
   * @param nuevaaerolinea Nombre de la nueva aerolinea
   */
  public void setaerolinea(String nuevaaerolinea) {
    aerolinea = nuevaaerolinea;
  }

  /**
   * Metodo para modificar el origen del vuelo
   * @param nuevoorigen String del nuevo origen del vuelo
   */
  public void setorigen(String nuevoorigen) {
    origen = nuevoorigen;
  }

  /**
   * Metodo para modificar el destino del vuelo
   * @param nuevodestino String del nuevo destino del vuelo
   */
  public void setdestino(String nuevodestino) {
    destino = nuevodestino;
  }

  /**
   * Metodo que modifica el numero de asientos totales del vuelo
   * @param nuevovalor Numero de la nueva cantidad de asientos
totales
   */
  public void setasientostotales(int nuevovalor) {
    asientostotales = nuevovalor;
  }

  /**
   * Metodo que modifica el numero de asientos disponibles en el
vuelo
   * @param nuevovalor Numero que asigna una
   * nueva cantidad de asientos diponibles
   */
  public void setasientosdisponibles(int nuevovalor) {
    asientosdisponibles = nuevovalor;
  }
}
```