Nuestro amigo Luis, dueño de un bar, dispone en el sótano de un local para celebraciones. Nos ha pedido hacer una pequeña aplicación para gestionar las reservas de dicho local. El local sólo lo reserva los sábados por la noche por lo que, para facilitar la aplicación se usará como identificador de la reserva el año en el que va a tener lugar la celebración seguido por un número de dos dígitos que identificará la semana de dicho año. Así, si por ejemplo, la celebración va a tener lugar el tercer sábado de enero del año 2023, el identificador será 202303. Cuando se vaya a dar de alta una reserva se solicitará al cliente el año y el número de semana que será un número entre 1 y 53, que es el número máximo de sábados que puede tener un año (no será necesario comprobar si el año indicado tiene 52 o 53 sábados). De una reserva se deberá almacenar el identificador y el nombre del cliente. Se puede hacer dos tipos de reserva, reserva de sólo el local que tiene un precio fijo de 250€ o con catering y barra libre. En el de las reservas con catering y barra libre se debe indicar además el número asistentes que estará comprendido entre 15 y 30 personas y, en este caso, el precio será de 25€ por persona en el que se incluye 2 horas de barra libre. Si se desea ampliar la barra libre, la ampliación costará 5€ por hora adicional, por lo que se deberá almacenar cuántas horas totales de barra libre se van a contratar (contando las dos que están incluidas en el precio dado). Todas las reservas incluirán un método llamado calcularPrecio que nos dará el precio total de la reserva. El programa principal contará como atributo con una lista de todas las reservas y nos mostrará un menú con las siguientes funciones: 1. Añadir reserva: preguntará por el tipo de reserva y pedirá los datos necesarios para añadirla a la lista de reservas. Antes de añadir una nueva reserva se comprobará que no existe ya una reserva con el mismo identificador, lo que indicaría que esa fecha ya está reservada. 2. Mostrar todas las reservas: mostrará un listado de las reservas en las que se mostrará el número de reserva, el nombre del cliente, en el caso de las reservas con catering mostrará además el número de personas y las horas de barra libre y, por último, mostrará el precio total de la reserva. Se dará la opción de volcar dicho listado a un fichero de texto cuyo nombre será proporcionado por el usuario. 3. Mostrar listado de las reservas con catering: mostrará únicamente aquellas reservas que tengan contratado el servicio de catering con toda la información indicada en la opción 2 del menú. 4. Anular reserva: se preguntará por el año y el número de semana de la reserva y, en caso de existir una reserva para esa fecha, se eliminará. 5. Salir Al finalizar la ejecución del programa se deberá guardar en un fichero binario llamado “reservas.bin” el listado de las reservas para recuperarlas al iniciar nuevamente la aplicación. Para aprobar el examen será imprescindible que éste no tenga errores de compilación. Se deberán controlar todas las posibles excepciones que pudiesen tener lugar.

```
package barluis;

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


public class BarLuis {


   public static ArrayList<Reserva> reservas = new ArrayList<Reserva>();
   public static final String FICHERO = "reservas.bin";

   public static int leerInt()
   {
      Scanner sc = new Scanner(System.in);
      while (true)
      {
         try
         {
            return sc.nextInt();
         }
         catch (Exception e)
         {
            System.out.println("NÃºmero incorrecto");
            sc.nextLine();
         }
      }
   }
```

```
public static int menu()
{
   int op = 0;
   while (op<1 || op>5)
   {
      System.out.println("1. AÃ±adir reserva");
      System.out.println("2. Mostrar todas las reservas");
      System.out.println("3. Mostrar listado de las reservas con catering");
      System.out.println("4. Anular reserva");
      System.out.println("5. Salir");
      op = leerInt();
   }
   return op;
}

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

public static void anadirReserva()
{
   Scanner sc = new Scanner(System.in);
   int op = 0;
```

```
    while (op!=1 && op!=2)
    {
       System.out.println("Â¿QuÃ© tipo de reserva quiere realizar?\n1. SÃ³lo local\n2. Con catering y barra libre");
       op = leerInt();
    }
    System.out.print("AÃ±o: ");
    int anio = leerInt();
    int sem = 0;
    while (!Reserva.esSemanaValida(sem))
    {
       System.out.print("Semana: ");
       sem = leerInt();
    }

    if (buscar(Reserva.generarId(anio, sem))== null)
    {

       System.out.print("Cliente: ");
       String cliente = sc.nextLine();
       if (op==2)
       {
          int personas = 0;
          while (!ReservaCatering.esPersonasValido(personas))
          {
             System.out.print("NÃºmero de personas: ");
             personas = leerInt();
          }
          System.out.print("Horas contratadas: ");
          int horas = leerInt();
          reservas.add(new ReservaCatering(anio,sem,cliente,personas,horas));
```

```
        }
        else
        {
            reservas.add(new ReservaLocal(anio,sem,cliente));
        }
    }
    else
        System.out.println("Esa fecha ya estÃ±a reservada.");
}

public static void guardarFicheroTexto()
{
    Scanner sc = new Scanner(System.in);
    System.out.print("Nombre del fichero: ");
    String f = sc.nextLine();
    BufferedWriter bw = null;
    try
    {
        bw = new BufferedWriter(new FileWriter(f));
        Iterator<Reserva> it = reservas.iterator();
        while (it.hasNext())
        {
            bw.write(it.next().toString()+"\n");
        }
    }
    catch (IOException e)
    {
        System.out.println("Error escribiendo el fichero de texto");
    }
    finally
    {
```

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

public static void guardarDatos()
{
    ObjectOutputStream oos = null;
    try
    {
        oos = new ObjectOutputStream(new FileOutputStream(FICHERO));
        oos.writeObject(reservas);
    }
    catch (IOException e)
    {
        System.out.println("Error guardando 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));
        reservas = (ArrayList<Reserva>) ois.readObject();
    }
    catch (FileNotFoundException e)
    {

    }
    catch (ClassNotFoundException e)
    {
        System.out.println("Error en los datos");
    }
    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 mostrarReservas()
{
    Iterator<Reserva> it = reservas.iterator();
    while (it.hasNext())
    {
       System.out.println(it.next());

    }
    int op = 0;
    while (op!=1 && op!=2)
    {
       System.out.println("Â¿Desea guardar el listado en un fichero?\n1. Si\n2. No");
       op = leerInt();
    }
    if (op==1)
    {
```

```
        guardarFicheroTexto();
    }

}

public static void mostrarReservasCatering()
{
    Iterator<Reserva> it = reservas.iterator();
    while (it.hasNext())
    {
        Reserva r = it.next();
        if (r instanceof ReservaCatering)
            System.out.println(r);
    }
}

public static void anularReserva()
{
    System.out.print("AÃ±o: ");
    int anio = leerInt();
    int sem = 0;
    while (!Reserva.esSemanaValida(sem))
    {
        System.out.print("Semana: ");
        sem = leerInt();
    }

    Reserva r = buscar(Reserva.generarId(anio, sem));
    if (r==null)
        System.out.println("No he encontrado ninguna reserva para esa fecha");
    else
```

```
        {
            reservas.remove(r);
            System.out.println("La reserva ha sido anulada correctamente");
        }
    }



    public static void main(String[] args) {

        recuperarDatos();
        int op = menu();
        while (op!=5)
        {
            switch (op)
            {
                case 1: anadirReserva(); break;
                case 2: mostrarReservas(); break;
                case 3: mostrarReservasCatering(); break;
                case 4: anularReserva(); break;
            }
            op = menu();
        }
        guardarDatos();
    }

}
```

```java
package barluis;

import java.io.Serializable;

/**
 *
 * @author anusk
 */
public abstract class Reserva implements Serializable{
    protected int id;
    protected String cliente;

    public Reserva (int anio, int sem, String c)
    {
        id = generarId(anio,sem);
        cliente = c;
    }

    public static int generarId(int a,int s)
    {
        return a*100 + s;
    }

    public static boolean esSemanaValida(int s)
    {
        return s>=1 && s<=53;
    }

    public int getId()
    {
        return id;
    }

    @Override
    public String toString()
    {
        return id+"\t"+cliente;
    }

    public abstract double calcularPrecio();


}
```

```java
package barluis;

/**
 *
 * @author anusk
 */
public class ReservaCatering extends Reserva{

    private int nPersonas;
    private int horas;
    private static final double PRECIO_PERSONA = 25;
    private static final int HORAS_MIN = 2;
    private static final double HORA_ADC = 5;
    private static final double PERSONAS_MIN = 15;
    private static final double PERSONAS_MAX = 30;

    public ReservaCatering(int anio, int sem, String c,int nPersonas, int
horas)
    {
        super(anio,sem,c);
        this.nPersonas = nPersonas;
        if (horas < 2)
            this.horas = 2;
        else
            this.horas = horas;
    }

    public static boolean esPersonasValido(int p)
    {
        return p>=PERSONAS_MIN && p<=PERSONAS_MAX;
    }

    @Override
    public double calcularPrecio()
    {
        return nPersonas*PRECIO_PERSONA + (horas-
HORAS_MIN)*HORA_ADC*nPersonas;
    }

    @Override
    public String toString()
    {
        return super.toString()+"\t"+nPersonas+" personas\t"+horas+"
horas\t"+calcularPrecio()+" â,¬";

    }

}
```

```java
package barluis;

/**
 *
 * @author anusk
 */
public class ReservaLocal extends Reserva{

    private static final double PRECIO = 250;

    public ReservaLocal(int anio, int sem, String c)
    {
        super(anio,sem,c);
    }

    @Override
    public double calcularPrecio()
    {
        return PRECIO;
    }

    @Override
    public String toString()
    {
        return super.toString()+"\t"+calcularPrecio()+" â,¬";
    }
}
```