

**PRUEBA 3 PROGRAMACIÓN
Junio 2007
INGENIERÍA INFORMÁTICA**



UNIVERSIDAD CARLOS III DE MADRID

LEA ATENTAMENTE ESTAS INSTRUCCIONES ANTES DE COMENZAR LA PRUEBA:

- Rellene todas las hojas a bolígrafo, tanto los datos personales como las respuestas. No use bolígrafo rojo.
- No olvide rellenar el NIA y el grupo real al que pertenece.
- El tiempo máximo de realización es de 1 hora.
- El único material permitido sobre la mesa es la hoja de test y un bolígrafo

NO PASE DE ESTA HOJA, hasta que se le indique

| | | |
|------------------|---------------|--------------|
| Apellidos | Nombre | |
| | | |
| Firma | NIA | Grupo |
| | | |

PARTE 1: CUESTIONES

Pregunta 1 (1 Punto).- Indicar si la siguiente afirmación es cierta, y explicar brevemente por qué.

“Es obligatorio o gestionar o lanzar todas las excepciones que puedan aparecer en un programa Java, si no el programa no compila.”

Pregunta 2 (1 Punto).- Indicar si la siguiente afirmación es cierta, y explicar brevemente por qué.

“Si en un bloque `try/catch` existen varias sentencias `catch` para capturar excepciones, es obligatorio ordenarlas poniendo antes las excepciones más particulares y después las más generales (clases hijas antes que clases madres).”

Pregunta 3 (1 Punto).- Indicar si la siguiente afirmación es cierta, y explicar brevemente por qué.

“La manera más adecuada para manejar flujos de datos en modo texto es utilizar las clases `FileInputStream` y `FileOutputStream`.”

Pregunta 4 (1 Punto).- Dado el siguiente método:

```
public void metodo1 () {
    try {
        File directorio1 = new File ("miDirectorio");
        File directorio2= new File (directorio1,"directorioHijo");
        File fichero1 = new File (directorio1,"fichero.dat");
        File fichero2 = new File (directorio2, "fichero.dat");
        if (!directorio2.exists()) directorio2.mkdir();
        if (!directorio1.exists()) directorio1.mkdir();
        fichero1.createNewFile();
        fichero1.renameTo(fichero2);
        fichero2.createNewFile();
        System.out.println(fichero1.exists());
        System.out.println(fichero2.exists());
    }
    catch (IOException e){
        System.out.println("Ha ocurrido una excepción de E/S");
    }
}
```

Y suponiendo que al principio no existe ninguno de los directorios ni ficheros. **Explicar** cuál sería el resultado por pantalla y cómo quedarían los ficheros y directorios tras la ejecución del método.

Respuesta:

Pregunta 5 (1 Punto).- Dado el siguiente método

```
public String metodo2(){
    String resultado = "";
    File fichero = new File ("fichero.dat");
    try {
        DataOutputStream dos = new DataOutputStream(new
FileOutputStream (fichero,true));
        dos.writeInt(4);
        dos.writeInt(6);
        dos.close();
        dos.writeInt(8);
    }
    catch (IOException e){
        resultado =";Vaya, algo falló!";
    }
    try {
        DataInputStream dis = new DataInputStream (new
FileInputStream (fichero));
        int b;
        b= dis.readInt();
        b+= dis.readInt();
        return resultado+String.valueOf(b);
    }
    catch (IOException e) {
        System.out.println("Otro error de E/S");
    }
    return resultado;
}
```

Explicar cuál sería el resultado por pantalla al ejecutar

```
System.out.println(metodo2());
```

Respuesta:

PARTE 2: PROBLEMAS

Problema 1 (3 Puntos).- Dado el siguiente código java:

```
public class Mueble {  
    private String material;  
    private float precio;  
    protected static int unidadesEnStock;  
    private int id;  
  
    public Mueble (String mat, float prec){  
        material = mat;  
        precio = prec;  
        unidadesEnStock++;  
        id = unidadesEnStock;  
    }  
  
    public Mueble (){  
        this ("indeterminado",-1.0F);  
    }  
}
```

1. Crear la clase `Mesa` que hereda de `Mueble` y tiene dos atributos privados `largo` y `ancho` de tipo `int`.
2. Crear un constructor para la clase `Mesa` que recibe valor para todos los parámetros (incluidos los heredados). Hacer otro constructor por defecto sin parámetros que use el constructor con parámetros anterior.
3. Modificar la clase `Mesa` y la clase `Mueble` para que se puedan serializar los objetos de estas clases.
4. Crear un método `public void guardar (String fichero)` en la clase `Mesa` que guarde un objeto de la clase `Mesa` en el fichero que se le pase por parámetro. Se deberán gestionar las excepciones pertinentes.
5. Crear un método `public Mesa leer (String fichero)` de la clase `Mesa` que lea un objeto de la clase `Mesa` del fichero que se le pase por parámetro. Se deberán gestionar las excepciones pertinentes.

Problema 2 (2 Puntos).- Dado el siguiente código java:

```
public int leeAño () {
    int resultado;
    try {
        BufferedReader br = new BufferedReader (new InputStreamReader
                                                (System.in));

        String lectura;
        lectura = br.readLine();
        resultado = Integer.parseInt(lectura);
        // Añadir el código que lanza la excepción aquí

    }
    catch (IOException e) {
        System.out.println ("Error al leer del teclado");
        return 1900;
    }
    catch (NumberFormatException e) {
        System.out.println ("Error al convertir a entero");
        return 1900;
    }
    //Añadir la gestión de la excepción aquí

}
return resultado;
}
```

1. Crear una excepción propia denominada `MiExcepcion` que tenga como atributo privado un número entero que se llame `valor` y un método `public int dameValor()` que devuelva el valor del atributo.
2. Crear un constructor para esta excepción que reciba como parámetro el valor del atributo `valor`. Crear también un constructor por defecto que ponga este valor a 0.
3. Cambiar el código del método `leeAño` para que si el valor de resultado es mayor que 2007, salte la excepción `MiExcepcion`.
4. Añadir al método `leeAño` el código necesario para que al saltar la excepción `miExcepcion` se imprima por pantalla "el año <valor> no es correcto" y el resultado que devuelva el método sea 1900.