



UNIVERSIDAD DE MALAGA
DEPARTAMENTO DE LENGUAJES Y
CIENCIAS DE LA COMPUTACION

**PROGRAMACION ORIENTADA A OBJETOS
CONVOCATORIA DE FEBRERO DE 2.002
INGENIERIA TECNICA EN INFORMATICA**

NOMBRE:..... APELLIDOS:.....

CURSO/PLAN:..... MÁQUINA:.....

Para realizar una prueba de tipo test se generan varios modelos cada uno de los cuales incluyen un número determinado de preguntas cuyas respuestas pueden ser A, B, C, D. Dos modelos de un mismo test se diferencian en el orden de las preguntas y en el orden de las cuatro respuestas ofrecidas en cada pregunta.

- 1) (2 ptos.) Crear la clase **Test** cuyos objetos mantienen información sobre los datos que conforman un test. Dichos datos son:
- Número de modelos distintos que se generan de un test.
 - Número de preguntas que contienen cada modelo de este test. Siempre se ofertan 4 respuestas posibles por pregunta.
 - Respuestas correctas a cada modelo del test. Los modelos de un test se numeran de 1 en adelante.

EJEMPLO: Un test concreto podrá contener:

- 4 modelos distintos.
- 12 preguntas por modelo.
- El modelo 1 tiene como respuestas "ABBACDDCABCD"
- El modelo 2 tiene como respuestas "BBBAAACCCDDD"
- El modelo 3 tiene como respuestas "CDCDACACBBAA"
- El modelo 4 tiene como respuestas "AABBAACCAADD"

Además, común a todos los test debe disponerse del valor que tiene una pregunta acertada y el valor que tiene una pregunta fallada. Por ejemplo: las preguntas acertadas valen 3 puntos y las falladas -1 punto (Estos valores serán tomados por defecto).

Dar los métodos necesarios para crear un objeto de la clase **Test**.

Crear también el método con prototipo **int corrigeModelo(int,String)** que permite corregir una prueba realizada por alguien. El primer argumento es el número del modelo realizado y el segundo argumento son las respuestas proporcionadas a ese modelo. El resultado es la puntuación obtenida. En la respuesta proporcionada, puede haber alguna respuesta en blanco. Esta no se contará ni como acertada ni como fallada (0 puntos). Por ejemplo, si **test** referencia a un objeto de la clase **Test** con la información antes descrita, entonces la respuesta a

test.corrigeModelo(3,"CDC ABAAB AA") será **22**

- 2) (1.5 ptos.) Crear la clase **Alumno** que contiene la información de un alumno que realiza una prueba de test. Los objetos de esta clase mantienen el nombre del alumno, el número de modelo que ha realizado y su respuesta a ese modelo. Dar los métodos necesarios para crear un objeto de esta clase. Además se debe disponer de un método que permite conocer la puntuación que dicho alumno ha obtenido; tendrá como prototipo **int corrige(Test)** donde el argumento es una instancia de **Test** con la que se debe corregir.

- 3) (2.5 ptos.) Para simular la realización de un test, se van generar un número de alumnos (instancias de la clase **Alumno**) con respuestas aleatorias. El nombre de estos alumnos siempre será el vuestro y tanto el modelo de test realizado como las respuestas serán aleatorias teniendo en cuenta que las cinco posibles respuestas (A,B,C,D o en blanco) deben tener la misma probabilidad así como el número de modelo realizado. Crear la clase **SimTest** en cuyo constructor se le proporcione una instancia de **Test** con los datos de un test. Por otro lado, el método con prototipo **void simula(int)** realizará una simulación (eliminando todo rastro de cualquier simulación anterior) donde el argumento es el número de alumnos que se deben simular. Este método debe guardar la información adecuada para que posteriormente pueda mostrarse a través del método **listado** que se define a continuación.

El método con prototipo **void listado(PrintWriter)** cuyo argumento es un flujo de datos, produce el siguiente listado sobre el flujo argumento con la información previamente almacenada:

Puntuación	Num. Alumnos
-3	5
-1	2
0	7
3	12
....	

(alumnos con puntuación -3 hay 5, con puntuación -1 hay 2, etc)

La suma de la última columna siempre debe coincidir con el número de alumnos que realizan la simulación. Además, el listado está ordenado ascendentemente por la puntuación. Crear una aplicación para probar la simulación.

- 4) (3 ptos.) Se dispone de un fichero como el siguiente:

3	valor resp. acertada
-1	valor resp. fallada
4	número de test distintos
12	preguntas por modelo
ABBACDDCABCD	respuestas correctas al modelo 1
BBBAAACCCDDD	respuestas correctas al modelo 2
CDCDACACBBAA	respuestas correctas al modelo 3
AABBAACCAADD	respuestas correctas al modelo 4
2:BB ABAC CDDD:López Hermoso, Juan	respuestas del alumno 1
1: ACD BCD:García García, Andrés	respuestas del alumno 2
3:CDCDACACBBAA:Romero López, Luisa	respuestas del alumno 3
...	

y se pretende generar otro fichero que contenga un listado ordenado por apellidos de las personas junto con la puntuación obtenida:

Nombre Alumno	Puntos
García García, Andrés	18
López Hermoso, Juan	26
Romero López, Luisa	36
....	

Crear en la clase **ListaTest** una aplicación que tome dos argumentos, el fichero de entrada y el fichero de salida. Tomar las decisiones necesarias sobre métodos, variables, clase padre, etc. para la realización de este ejercicio. (Un fichero que vale de ejemplo de entrada es C:\JAVA\alumnos.txt). Controlar las posibles excepciones que se puedan producir.

PROGRAMACION ORIENTADA A OBJETOS

NOTAS PARA LA REALIZACION DEL EXAMEN

(Leer detenidamente antes del examen)

1. Para la realización del ejercicio puede usarse el entorno que se quiera.
2. Borrar todo el contenido del directorio **C:\JAVA**. Es en este directorio donde se debe realizar el ejercicio.
3. Copiar el contenido del directorio **F:\USERS\JAVA\EXAMEN y sus subdirectorios** al directorio **C:\JAVA** (Si no estáis en Windows no tenéis visibilidad sobre ese directorio).

Al finalizar el examen, el profesor correspondiente pasará por cada puesto. Para cuando llegue, el directorio **C:\JAVA** deberá contener **EXCLUSIVAMENTE**:

Las clases:	Test.java Alumno.java SimTest.java ListaTest.java
Los ficheros:	El fichero de salida del ejercicio 4

SOLO ESTO será salvado por el profesor. Revisar que los nombres coinciden. Cualquier otro fichero será borrado.

Para las salidas con formato, usar la clase **ES.class** que se encuentra en el paquete **es** (que se debe haber copiado cuando se realizó lo indicado en el apartado 3). Hay documentación de esta clase en el mismo directorio en formato HTML.

Pueden construirse otros métodos además de los que se indican en cada pregunta. A veces, es necesario.

IDENTIFICAR CADA FICHERO CON EL NOMBRE, CURSO, PLAN Y NÚMERO DE EQUIPO.

TODAS LAS PREGUNTAS INDICAN SU VALOR. Se concederá 1 punto por la claridad de los algoritmos, del código y la correcta elección de las estructuras de datos.

PODEIS:

- Consultar la página <http://www.lcc.uma.es/~pacog/> en donde se encuentran los apuntes.
- Consultar el API que se encuentra en `c:\jdk1.3\docs\api\index.html`.
- Consultar el tutorial que se encuentra en `c:\jdk1.3\tutorial\index.html`

NO PODEIS:

- Utilizar otra documentación electrónica o impresa.
- Intercambiar documentación entre vosotros.
- Utilizar soportes magnéticos.
- Copiar vuestro ejercicio al final del examen. Si alguien desea una copia de su ejercicio, que se pase posteriormente por el despacho del profesor a retirarla.

IMPORTANTE: APAGAD LOS DISPOSITIVOS ELECTRÓNICOS DE COMUNICACIÓN

Hoja en blanco