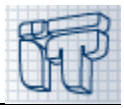


NOMBRE:
APELLIDOS:
NIA:
GRUPO:

2ª Parte: Problemas (7 puntos sobre 10)

Duración: 70 minutos
Puntuación máxima: 7 puntos
Fecha: 15 marzo 2019

Instrucciones para el examen:

- No se permite el uso de libros o apuntes, ni tener teléfonos móviles u otros dispositivos electrónicos encendidos. Incumplir cualquiera de estas normas puede ser motivo de expulsión inmediata del examen.
- Rellena tus datos personales antes de comenzar a realizar el examen.
- Utiliza el espacio de los recuadros en blanco para responder a bolígrafo a cada uno de los apartados de los problemas (no usar lápiz para las respuestas finales).

NOTA: No está permitido crear más atributos ni métodos de los que figuran en el enunciado (no son necesarios).

Problema 1 (7 puntos)

Un equipo de informativos decide crear un programa para gestionar mejor a los empleados que se desplazan a cubrir diferentes reportajes.

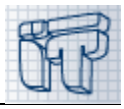
- Cada equipo tiene periodistas (`Periodista`) de diferentes categorías. De momento sólo se han implementado dos tipos de periodistas: `Redactor` para modelar a los empleados que escriben el texto de las noticias y `ReporteroGrafico` para aquellos que hacen fotografías o grabaciones de cámara.
- Todos los periodistas independientemente de su categoría (`categoria`) tienen además un nombre (`nombre`) y un número de empleado (`numEmpleado`).
- El número de empleado (`numEmpleado`) se asigna de forma consecutiva cada vez que se contrata a un nuevo periodista. También es necesario llevar el registro del número total de periodistas contratados (`empleadosTotales`).
- Cada periodista empleado en el equipo realiza un trabajo diferente que depende de su categoría.
- El responsable de informativos formará equipos (`Equipo`) con periodistas de diferentes tipos (`Redactores y Reporteros Gráficos`) dependiendo de la noticia a cubrir.

Para colaborar en el modelado de los diferentes elementos del sistema deberás realizar las tareas indicadas en cada apartado.

Apartado 1: Interfaz Empleable (0,2 puntos)

Declara la interfaz `Empleable` y el método `trabajar` que no recibe parámetros ni tiene tipo de retorno.

```
Apartado 1 (0,2 puntos)
```

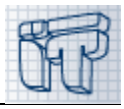


Apartado 2: Clase Periodista (1,5 puntos)

Declara la clase `Periodista` que implemente la interfaz declarada en el apartado anterior teniendo en cuenta las siguientes especificaciones:

- Declara tres atributos que no puedan ser accedidos ni modificados por ninguna otra clase: `Nombre` de tipo cadena, `categoria` de tipo carácter y `numEmpleado` de tipo entero.
- Declara un atributo `empleadosTotales` que sea el mismo para todos los objetos de la clase.
- Crea tres constantes que puedan ser accedidas por cualquier clase que se llamen: `REDACTOR`, `REPORTER_GRAFICO` y `NOT_ASIGNED` que tomen los valores '`r`', '`g`' y '`x`' respectivamente para representar los tres tipos de clientes permitidos por la aplicación.
- Crea un método `getCategoria` y un método `setCategoria` que permitan devolver y asignar el valor a la variable `categoria`. El método `setCategoria` deberá además controlar que el valor del tipo asignado es uno de los 3 valores permitidos. Si se introduce un valor incorrecto el método `setCategoria` asignará por defecto el valor '`x`' (`NOT_ASIGNED`) al.
- La clase `Periodista` no dispone de información suficiente para implementar el método `trabajar()` de la interfaz

```
Apartado 2 (1,5 puntos)
```



Apartado 3: Constructores y método toString clase Periodista (1 punto)

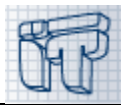
Implementa dos constructores para la clase `Periodista`

- El primer constructor deberá recibir como parámetros el nombre (`nombre`) y la categoría del periodista (`Categoria`) y asignar valor a los atributos, `nombre`, `categoria`, `numEmpleado` y `empleadosTotales` asegurándose sin repetir código que el valor asignado para la categoría es el valor correcto.
- Recuerda que el número de empleado (`numEmpleado`) se asigna de forma consecutiva cada vez que se contrata a un nuevo periodista pero también es necesario llevar un contador genérico del número total de empleados (`empleadosTotales`) que es el mismo para todos los objetos de la clase.
- El segundo constructor es un constructor por defecto que no recibe parámetros y asigna a cada atributo su valor por defecto. Este constructor deberá llamar al anterior para no repetir código.

Implementa el método `toString` de la clase `Periodista` que imprima la información del cliente en siguiente formato:

```
Nombre: <nombre>
Categoría: <categoria>
NumEmpleado:<employeeNum>
```

Apartado 3 (1 puntos)

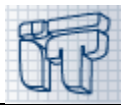


Apartado 4: Clase Redactor (constructores y métodos) (1 punto)

Declara la clase `Redactor` que hereda de la clase `Periodista`

- Declara dos constructores con el mismo número y tipo de parámetros que los constructores de la clase padre y que realicen la misma función que ellos (sin repetir código).
- Implementa el método trabajar de la clase `Redactor` este método simplemente imprime en pantalla el mensaje: `"Redactor trabajando"`. Del mismo modo, el método trabajar de la clase `ReporteroGrafico` imprimiría: `"Reportero Gráfico trabajando"`, pero este segundo método trabajar no es necesario que lo implementes.

```
Apartado 4 (1 puntos)
```

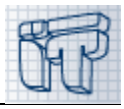


Apartado 5: Clase Equipo (1,6 puntos)

Programa una clase `Equipo` que implemente la interfaz `Empleable` teniendo en cuenta las siguientes especificaciones:

- Declara un atributo `equipo` que permita almacenar un array de `Periodistas` y un constructor que reciba como parámetro un array de periodistas y lo asigne al atributo correspondiente.
- Declara un método `toString()` a la clase equipo que recorra el array y devuelva el resultado de invocar al método `toString()` de cada uno de los miembros del equipo.
- Implementa el método trabajar de la clase `Equipo` de modo que recorra el array invocando al método `trabajar()` de todos los miembros del equipo.

```
Apartado 5 (1,6 puntos)
```



Apartado 6: Clase Test y Método Main (1,2 puntos)

Programa una clase `Test` que contenga un método `main` que realice las siguientes operaciones:

- Cree un objeto de tipo Redactor y otro de tipo Reportero Gráfico
- Cree un equipo con ambos componentes.
- Imprima en pantalla la información de cada método del equipo proporcionada por el método toString e invoque al método trabajar de cada uno de sus miembros.

El resultado de ejecutar el método main mostraría en pantalla el siguiente resultado:

```
Nombre: Pedro R.
Categoría: r
NumEmpleado:1
Nombre: Juan J.
Categoría: g
NumEmpleado:2
Redactor trabajando
Reportero gráfico trabajando
```

Apartado 6 (1,2 puntos)

Apartado 7: Pruebas de programa: Cobertura (0,5 puntos)

- En función de tu propia implementación de los métodos `main` y `setCategoria` indica qué cobertura de ramas has conseguido para el método `setCategoria` tras ejecutar todas las sentencias del método `main`.

Apartado 7 (0,5 puntos)

Criterios de corrección

Apartado 1 Interfaz (0,2 puntos)

- (0,2) Declaración de interfaz y método abstracto.
  - (0 si no demuestra conocimiento de interfaz porque pone abstract en la declaración o implementa el método o pone {} en vez de ;)
  - No penalizar si pone public en el método aunque al ser una interfaz no es necesario porque todos sus métodos lo son.

Apartado 2 Clase Abstracta (1,5 puntos)

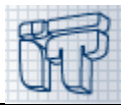
- (0,2) Declaración de la clase abstracta que implementa la interfaz
  - 0 si no pone abstract o no demuestra conocimiento de lo que es una clase abstracta. No es necesario que ponga el método de la interfaz. Pero no penalizar si lo pone de forma correcta.
  - Penalizar con -0,1 si pone abstract pero no pone implements
- (0.3) Declaración de los atributos privados nombre, categoria y numEmpleado.
  - 0 si hay algún fallo en los modificadores o tipos de datos.
- (0,2) Declaración del atributo static
  - 0 si hay algún fallo en modificadores o tipos de datos
- (0,2) Declaración de constantes
  - 0 si hay algún fallo en modificadores o tipos de datos
  - -0.1 si le falta alguno de los tres valores pero los demás son correctos.
- (0,2) Método get() (0 si tiene cualquier fallo)
- (0,4) Método set()
  - (-0.1) si controla todos los valores menos el valor por defecto.
  - No penalizar si lo hace de forma correcta aunque ineficiente

Apartado 3. Constructor y método toString de la clase abstracta (1 punto)

- (0,6) Constructor con parámetros
  - (0,1) declaración
  - (0,1) manejo y asignación del nombre
  - (0,2) manejo y asignación de categoría invocando a set (0 si lo hace directamente).
  - (0,1) manejo y asignación del atributo estático
  - (0,1) manejo y asignación del numEmpleado
- (0,2) Constructor sin parámetros
  - 0.1 declaración
  - 0.1 asignación
  - Penalizar con -0,1si hace asignación correcta pero repitiendo código (sin llamar al otro constructor)
  - 0 si no controla que el valor de categoría sea uno de los correctos.
- (0,2) Método toString. (0 si tiene algún fallo importante como no devolver un String o si la declaración no es correcta).
  - 0,1 declaración
  - 0,1 contenido
  - Penalizar con -0,1 si tiene algún fallo menor como olvidar los saltos de línea.

Apartado 4: Clase derivada (1 punto)

- (0,25) Declaración de la clase
- (0,25) Constructor con parámetros
  - 0 si declaración no es correcta o si no hace llamada a super porque no hay otro modo de asignar valor a los atributos al ser privados porque nombre no tiene método set.
- (0,25) Constructor sin parámetros
  - considerar válido llamar al constructor de la misma clase (si es correcto) o al constructor de la clase padre
  - 0 si declaración no es correcta o si hace asignación directamente porque los atributos son privados
- (0,25) Declaración e implementación del método trabajar



Apartado 5: Clase agregada Equipo (1,6 puntos)

- (0,25) Declaración de la clase
- (0,1) Declaración del atributo
- (0,25) Constructor
- (1) Metodo toString() y método trabajar()
    - (0,25) Recorrer el array en los métodos toString() y trabajar() (0 si está mal el recorrido o los límites son incorrectos)
    - (0,25) Llamada a toString() de los elementos del array
    - (0,25) Concatenar los resultados de las llamadas a toString() y devolver el valor correcto.
    - (0,25) Llamada a trabajar() de los elementos del array

Apartado 6: Clase Test y Método main (1,2 punto)

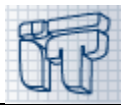
- (0,2) Declaración de la clase y método main
- (0,2) Declaración y creación de objetos Reportero y ReporteroGrafico (0 si hay algún fallo en la declaración o la llamada a los constructores)
- (0,2) Declaración y asignación de valores al array que se va a pasar como parámetro
    - (0 si no conoce manejo de arrays).
    - Sumar al apartado siguiente si lo hace directamente al crear el equipo.
- (0,2) Declaración y creación del equipo.
- (0,2) Imprimir información del equipo
    - 0 si invocan directamente a equipo.toString() sin ponerlo dentro de System.out.println() o almacenando el valor en una variable que luego imprimen.
    - Considerar correcto tanto System.out.println(equipo) como System.out.println(equipo.toString())
- (0,2) Invocar correctamente al método trabajar

Apartado 7: Pruebas de programa (0,5 punto)

- Puntuar sólo si la respuesta es correcta y consistente con el código creado.

```
Apartado 1 (0,2 puntos)
public interface Empleable{
   public void trabajar();
}
```

```
Apartado 2 (1,5 puntos)
public abstract class Periodista implements Empleable{
  private String nombre;
  private char categoria;
  private int numEmpleado;
  public static int empleadosTotales;
  public static final char REDACTOR = 'r';
  public static final char REPORTERO_GRAFICO = 'g';
  public static final char NO_ASIGNADO = 'x';
  public char getCategoria(){
    return categoria;
  }
  public void setCategoria(char categoria){
    if(categoria == REDACTOR || categoria == REPORTERO_GRAFICO){
        this.categoria = categoria;
    }else{
        this.categoria = NO_ASIGNADO;
    }
  }
```



```
Apartado 3 (1 puntos)
  public Periodista(String nombre, char categoria){
    this.nombre = nombre;
      setCategoria(categoria);
      empleadosTotales++;
      numEmpleado = empleadosTotales;
  }
  public Periodista(){
    this(null, NO_ASIGNADO);
  }
  public String toString(){
    String resultado = "Nombre: "+ nombre + "\n" +
                               "Tipo: " + categoria + "\n" +
                               "ID: " + numEmpleado + "\n" ;
      return resultado;
  }
```

```
Apartado 4 (1 puntos)
public class Redactor extends Periodista{
  public Redactor(String nombre, char categoria){
    super(nombre, categoria);
  }
  public Redactor(){
    this(null,'x');
  }
  public void trabajar(){
   System.out.println("Redactor trabajando");
  }
}
```

```
Apartado 5 (1,6 puntos)
public class Equipo implements Empleable{
  private Periodista[] equipo;
  public Equipo (Periodista[] equipo){
    this.equipo = equipo;
  }
  public String toString(){
    String resultado = "Equipo formado por: "+"\n";
    for(int i=0; i<equipo.length; i++){
        resultado = resultado + equipo[i].toString();
      }
      return resultado;
  }
  public void trabajar(){
    for(int i=0; i<equipo.length; i++){
        equipo[i].trabajar();
      }
  }
}
```

Apartado 6 (1,2 puntos)

```
public class Test{
  public static void main(String[] args){
    Redactor r = new Redactor("Pedro P.",Periodista.REDACTOR);
      ReporteroGrafico rg = new ReporteroGrafico ("Juan J.",
Periodista.REPORTERO_GRAFICO);
      Periodista[] datos = {r, rg};
      Equipo equipo = new Equipo(datos);
      System.out.println(equipo);
      equipo.trabajar();
  }
}
```

Apartado 7 (0,5 puntos)