Tarea5

Nombre alumno o alumna: Podadera González, Andrés Samuel

Fecha de entrega: jueves, 12 de enero de 2023, 22:11

Retroalimentación global de la actividad:

Hola, Andrés. Muy bien resuelto. Te indico algunos detalles que debes revisar. Por lo demás, magnífico trabajo.
Ahora, a seguir trabajando con los ejercicios del foro de entrenamiento.

Puntuación orientativa de la actividad: **9,90**

RESULTADOS DE APRENDIZAJE EVALUADOS EN ESTA ACTIVIDAD

RA3 ***Escribe y depura código, analizando y utilizando las estructuras de control del lenguaje.***

		Punt. Máx	Punt. Obt.	Punt. Final (sobre 10)
RA3.d	Se ha escrito código utilizando control de excepciones.	19	19,00	**10,0**

RA4 ***Desarrolla programas organizados en clases analizando y aplicando los principios de la programación orientada a objetos.***

		Punt. Máx	Punt. Obt.	Punt. Final (sobre 10)
RA4.a	Se ha reconocido la sintaxis, estructura y componentes típicos de una clase.	34	32,50	**9,6**
RA4.b	Se han definido clases.	27	26,50	**9,8**
RA4.c	Se han definido propiedades y métodos.	64	62,00	**9,7**
RA4.d	Se han creado constructores.	6	6,00	**10,0**
RA4.e	Se han desarrollado programas que instancien y utilicen objetos de las clases creadas anteriormente.	5	5,00	**10,0**
RA4.f	Se han utilizado mecanismos para controlar la visibilidad de las clases y de sus miembros.	34	34,00	**10,0**
RA4.h	Se han creado y utilizado métodos estáticos.	5	5,00	**10,0**

CORRECCIÓN POR EJERCICIO/ACTIVIDAD/ELEMENTO

Ejercicio 1. Atributos de la clase.

Puntos de control (elementos evaluables)		Punt. Máx.	Punt. Obt.	CE	CE	CE	CE	CE	CE
PC1	*Declaración de los atributos de clase constantes (public final static): AFORO_MAX, AFORO_MIN, DEFAULT_AFORO*	2	2,0	RA4.a	RA4.c	RA4.f			
PC2	*Declaración de los atributos variables de clase (estáticos) de tipo "contador": numTeatros, numObras y entradasVendidasTotales*	2	2,0	RA4.a	RA4.c	RA4.f			
PC3	*Declaración de los atributos de objeto constantes (como constantes final): codigoTeatro, nombreTeatro y aforo.*	2	2,0	RA4.a	RA4.c	RA4.f			
PC4	*Declaración de los atributos de objeto variables: obra y entradasVendidas*	1	1,0	RA4.a	RA4.c	RA4.f			
PC5	*Los nombres de los identificadores (clase, atributos, variables, constantes, etc.) son representativos de la información que están almacenando y cumplen las convenciones establecidas para el lenguaje Java (uso de mayúsculas, minúsculas, guiones, etc. dependiendo de la función de ese identificador: variable, constante, clase, etc.).*	1	1,0	RA4.a	RA4.b				
PC6	*A la clase no le falta ningún elemento necesario y no incluye elementos extraños, innecesarios o sin sentido.*	2	1,5	RA4.a	RA4.b				

Retroalimentación:

Tienes un atributo redundante: `teatrosTotales` y `codigosTeatros` hacen prácticamente lo mismo. Solo con un atributo sería suficiente.

Ejercicio 2. Constructores.

Puntos de control (elementos evaluables)		Punt. Máx.	Punt. Obt.	CE	CE	CE	CE	CE	CE
PC1	*Comprobación de que el parámetro nombre no sea null: si es null, se lanza una excepción NullPointerException con un mensaje de error apropiado.*	1	1,0	RA3.d					
PC2	*Comprobación de que los parámetros de entrada (nombre y aforo) son correctos. Si no es así, se lanza una excepción IllegalArgumentException con un mensaje de error apropiado.*	2	2,0	RA3.d					
PC3	*Inicialización de los atributos de objeto, tanto inmutables (codigoTeatro, nombreTeatro y aforo) como de estado (obra y entradasVendidas).*	2	2,0	RA4.a	RA4.c				
PC4	*Actualización del atributo de clase numTeatros*	1	1,0	RA4.a	RA4.c				
PC5	*Incluye en la cabecera de los constructores las posibles excepciones: throws IllegalArgumentException, NullPointerException.*	1	1,0	RA3.d	RA4.d				
PC6	Se utilizan los atributos constantes de clase (AFORO_MIN, AFORO_MAX) para realizar las comprobaciones de rango y coherencia del parámetro AFORO.	1	0,0	RA4.a	RA4.c				
PC7	*Los valores por omisión asignados a los atributos son los que se especifican en el enunciado: Teatro.DEFAULT_AFORO y construcción del nombre del teatro según el patrón "Teatro "+ (Teatro.numTeatros +1).*	1	1,0	RA4.a	RA4.c				
PC8	La cabecera de los constructores está correctamente definida y se realiza una implementación apropiada de la sobrecarga.	1	1,0	RA4.b	RA4.d	RA4.f	RA3.d		
PC9	Invocación apropiada a otros constructores, *evitándose código redundante.*	2	2,0	RA4.b	RA4.d				
PC10	*No se incluyen elementos extraños, innecesarios o sin sentido.*	2	2,0	RA4.b	RA4.d				

Retroalimentación:

En el **constructor**: deberías evitar el uso de constantes literales (300, 1000). Para eso precisamente se han definido los atributos AFORO_MIN, AFORO_MAX, etc.

Ejercicio 3. Métodos de consulta.									
Puntos de control (elementos evaluables)		**Punt. Máx.**	**Punt. Obt.**	**CE**	**CE**	**CE**	**CE**	**CE**	**CE**
PC1	*Implementación correcta de los métodos getter: getNombreTeatro y getCodigoTeatro.*	2	2,0	RA4.c	RA4.f				
PC2	*Implementación correcta de los métodos getter: getAforo y getEntradasVendidas.*	2	2,0	RA4.c	RA4.f				
PC3	*Implementación correcta de los métodos getter: tieneObra y getObra.*	2	2,0	RA4.c	RA4.f				
PC4	*Implementación correcta de los métodos estáticos para la devolución de información de la clase: getTeatrosTotales, getObrasActivas y getEntradasVendidasTotales.*	3	3,0	RA4.c	RA4.f	RA4.h			
PC5	*No se incluyen elementos extraños, innecesarios o sin sentido.*	2	2,0	RA4.c	RA4.f	RA4.h			

Retroalimentación:

Correcto.

Ejercicio 4. Asignar y Terminar obra.									
Puntos de control (elementos evaluables)		**Punt. Máx.**	**Punt. Obt.**	**CE**	**CE**	**CE**	**CE**	**CE**	**CE**
PC1	*Método asignarObra: Se lanzan las excepciones necesarias, con un mensaje de error apropiado, para el control del parámetro de entrada (NullPointerException e IllegalArgumentException).*	1	1,0	RA3.d					
PC2	*Método asignarObra: Se lanza la excepción IllegalStateException si el teatro ya tiene una obra asignada, con un mensaje de error apropiado.*	1	1,0	RA3.d					
PC3	*Método asignarObra: Se actualizan los atributos de estado necesarios (obra).*	2	2,0	RA4.c					
PC4	*Método asignarObra:Se actualizan los atributos de clase si es necesario (numObras).*	1	1,0	RA4.c					
PC5	*Método terminarObra: Se lanza la excepción IllegalStateException si el teatro no tiene una obra asignada, con un mensaje de error apropiado.*	1	1,0	RA3.d					
PC6	*Método terminarObra: Se actualizan los atributos de estado necesarios (obra y entradasVendidas)*	2	1,0	RA4.c					
PC7	*Método terminarObra:Se actualizan los atributos de clase si es necesario (numObras)*	1	1,0	RA4.c					
PC8	*Se define la cabecera de los métodos correctamente: nombre, modificadores, visibilidad, valor devuelto, posibles parámetros, excepciones (throws),…*	1	1,0	RA3.d	RA4.c	RA4.f			
PC9	*No se incluyen elementos extraños, innecesarios o sin sentido.*	2	2,0	RA4.c	RA4.f				

Retroalimentación:

En lugar de indicar “//Métodos VOID”, quizá podrías indicar “//Métodos de acción”. Se entenderá mejor.
Al terminar una obra, deberías actualizar el número de entradas vendidas a cero. Si no, al iniciar otra obra, tendrás ya entradas vendidas, lo cual no tiene mucho sentido. Otra opción podría ser ponerlas a cero al asignar una nueva obra. Pero en algún momento hay que hacerlo. Es mejor hacerlo al terminar una obra para que el método `getEntradasVendidas()` no devuelva algo mayor que cero cuando no hay una obra asignada.

Ejercicio 5. Compra y devolución de entradas.									
Puntos de control (elementos evaluables)		**Punt. Máx.**	**Punt. Obt.**	**CE**	**CE**	**CE**	**CE**	**CE**	**CE**
PC1	*Método comprarEntradas: Se lanzan las excepciones necesarias, con un mensaje de error apropiado, para el control tanto del parámetro de entrada como del estado del objeto (IllegalStateException e IllegalArgumentException).*	1	1,0	RA3.d					
PC2	*Método comprarEntradas: Se actualizan los atributos de estado y de clase necesarios (entradasVendidas y entradasVendidasTotales).*	2	2,0	RA4.c					
PC3	*Método comprarEntrada:Se implementa mediante una invocación al método correspondiente (comprarEntradas), evitando así la redundancia de código.*	1	1,0	RA4.c					
PC4	*Método devolverEntradas: Se lanzan las excepciones necesarias, con un mensaje de error apropiado, para el control tanto del parámetro de entrada como del estado del objeto (IllegalStateException y IllegalArgumentException).*	1	1,0	RA3.d					
PC5	*Método devolverEntradas: Se actualizan los atributos de estado y de clase necesarios (entradasVendidas y entradasVendidasTotales).*	2	2,0	RA4.c					
PC6	*Método devolverEntrada:Se implementa mediante una invocación al método correspondiente (devolverEntradas), evitando así la redundancia de código.*	1	1,0	RA4.c					
PC7	*Método llenarTeatro: Se lanzan las excepciones necesarias, con un mensaje de error apropiado, para el control del estado del objeto (IllegalStateException).*	1	1,0	RA3.d					
PC8	*Método llenarTeatro:Se implementa mediante una invocación al método correspondiente (comprarEntradas), evitando así la redundancia de código.*	2	2,0	RA4.c					
PC9	*Método vaciarTeatro: Se lanzan las excepciones necesarias, con un mensaje de error apropiado, para el control del estado del objeto (IllegalStateException).*	1	1,0	RA3.d					
PC10	*Método vaciarTeatro:Se implementa mediante una invocación al método correspondiente (devolverEntradas), evitando así la redundancia de código.*	2	2,0	RA4.c					
PC11	*Se define la cabecera de los métodos correctamente: nombre, modificadores, visibilidad, valor devuelto, posibles parámetros, excepciones (throws),…*	3	3,0	RA3.d	RA4.c	RA4.f			
PC12	*No se incluyen elementos extraños, innecesarios o sin sentido.*	2	2,0	RA4.c	RA4.f				

Retroalimentación:

Correcto.

Ejercicio 6. Traspasar obra.									
Puntos de control (elementos evaluables)		**Punt. Máx.**	**Punt. Obt.**	**CE**	**CE**	**CE**	**CE**	**CE**	**CE**
PC1	*Se comprueba que el parámetro que se pasa no sea null: si es null, se lanza una excepción NullPointerException con un mensaje de error apropiado.*	1	1,0	RA3.d	RA4.e				
PC2	*Se comprueba correctamente el estado del teatro origen (con obra asignada) y del teatro destino (sin obra asignada), lanzándose las excepciones correspondientes (IllegalStateException) si el estado de alguno no es correcto y con un mensaje de error apropiado.*	1	1,0	RA3.d	RA4.e				
PC3	*Se actualizan correctamente los atributos necesarios en el teatro origen (this.obra y this.entradasVendidas).*	2	2,0	RA4.c					
PC4	*Se actualizan correctamente los atributos necesarios en el teatro destino (otroTeatro.obra y otroTeatro.entradasVendidas).*	2	2,0	RA4.c	RA4.e				
PC5	*Se define la cabecera del método correctamente: nombre, modificadores, visibilidad, valor devuelto, posibles parámetros, excepciones (throws),…*	1	1,0	RA3.d	RA4.c	RA4.e	RA4.f		
PC6	*No se incluyen elementos extraños, innecesarios o sin sentido.*	2	2,0	RA4.c	RA4.f				

Retroalimentación:
Correcto.

Ejercicio 7. Método toString.									
Puntos de control (elementos evaluables)		**Punt. Máx.**	**Punt. Obt.**	**CE**	**CE**	**CE**	**CE**	**CE**	**CE**
PC1	*Se incluyen en la cadena de salida todas las características y valores de estado solicitados.*	1	1,0	RA4.c					
PC2	*Se respeta el formato solicitado, devolviendo la información de forma correcta.*	3	3,0	RA4.c					
PC3	*Se ha utilizado el método String.format para la construcción de la cadena de salida.*	2	2,0	RA4.c					
PC4	*Se define la cabecera del método correctamente: nombre, modificadores, visibilidad, valor devuelto, etc.*	2	2,0	RA4.c	RA4.f				
PC5	*No se incluyen elementos extraños, innecesarios o sin sentido.*	2	2,0	RA4.c	RA4.f				

Retroalimentación:
Correcto.

Ejercicio 8. Documentación javadoc.									
Puntos de control (elementos evaluables)		**Punt. Máx.**	**Punt. Obt.**	**CE**	**CE**	**CE**	**CE**	**CE**	**CE**
PC1	*Se documenta apropiadamente la clase en general.*	2	2,0	RA4.a	RA4.b				
PC2	*Se documentan apropiadamente los atributos públicos estáticos (constantes). Uso de @value.*	2	2,0	RA4.a	RA4.b				
PC3	*Se documentan apropiadamente los constructores.*	3	3,0	RA4.a	RA4.b				
PC4	*Se documentan apropiadamente los getters.*	2	2,0	RA4.a	RA4.b				
PC5	*Se documentan apropiadamente los métodos de acción: asignarObra, terminarObra, comprarEntradas, comprarEntrada, devolverEntradas, devolverEntrada, llenarTeatro, vaciarTeatro y traspasarObra.*	3	3,0	RA4.a	RA4.b				
PC6	*Se documenta apropiadamente el método toString.*	1	1,0	RA4.a	RA4.b				
PC7	*No se incluyen elementos extraños, innecesarios o sin sentido.*	2	2,0	RA4.a	RA4.b				
PC8	No se producen errores o warning durante la generación de la documentación javadoc.	4	4,0	RA4.a	RA4.b				

Retroalimentación:
Correcto.