**Parciales de Módulo Programación Orientadas a Objetos.**

**Parcial 1.**

Se desea implementar un sistema que permita administrar los discos y canciones de una banda de música. La banda de música se caracteriza por tener un nombre, la ciudad donde se formó y el estilo musical. La banda ademas tiene información de todos sus discos sacados a la venta (a lo sumo 20). Un disco se caracteriza por tener un nombre, la fecha de salida al mercado (por simplicidad la fecha es un int) y las canciones (a lo sumo 25). Una cancion se caracteriza por tener un nombre y una duración en segundos.

1. Realice el modelo de clases. Implemente las clases con sus atributos y métodos para obtener/ modificar el valor de los mismos, ademas de los constructores necesarios.

2. Implemente la posibilidad de poder agregar discos a una banda y de igual manera agregar canciones a un disco.

3. Implemente los métodos necesarios (en las clases que corresponden) para poder conocer:

   i. La cantidad de discos de una banda.

   ii. Dado el nombre de un disco y el nombre de una canción la duración del mismo.

   iii. El nombre del tema más largo de una banda.

   iv. La cantidad de temas del disco más viejo.

**Notas:**

- Los métodos NO deben imprimir nada, solo devolver lo solicitado.
- Piense que todo lo pedido debe poder ser obtenido a partir de una banda, es decir: si en la función main se instancia una banda, a partir de ella se debe poder conocer/ hacer todo lo pedido.

**Parcial 2.**

Un cine dese un sistema para poder vender entradas para las películas en cada una de sus salas. El cine se caracteriza por tener un nombre, una dirección y el nombre del dueño. El cine además tiene información de sus salas (a lo sumo 5). Una sala se caracteriza por tener un numero de sala, el cupo de entradas y la cantidad vendida. Además de tener informacion de la película proyectada en esa sala. Una película se caracteriza por tener un nombre, un director y la duración en minutos.

1. Realice el modelo de clases. Implemente las clases con sus atributos y métodos para obtener/modificar el valor de los mismos, además de los constructores necesarios.

2. Implemente la posibilidad de poder agregar salas al cine y una película a una sala. También se debe implementa un método que permita la venta de entradas de una determinada sala, como así también la devolución de una entrada.

3. Implemente los métodos necesarios (en las clases que correspondan) para poder conocer:

   i. Dado un numero de sala, saber si quedan entradas disponibles.

   ii. Dado el nombre de una película saber en qué sala se proyecta.

   iii. La sala con mayor cantidad de entradas vendidas.

   iv. El nombre de la película que más entradas vendió.

**Notas:**

- Los métodos NO deben imprimir nada, solo devolver lo solicitado.
- Piense que todo lo pedido debe poder ser accedido a partir de un cine, es decir: si en la función main instancia un cine, a partir de él se debe poder conocer/hacer todo lo solicitado.

**Parcial 3.**

Una discográfica desea un sistema para poder manejar discos de artistas solistas y de bandas. La discográfica se caracteriza por tener un nombre, la ciudad de residencia y el nombre del dueño. La discográfica además tiene información de todos los discos de solistas y bandas que representa (a lo sumo 50 de cada uno). Un artista solista se caracteriza por tener un nombre, instrumento que toca y la cantidad de temas producidos. Una banda se caracteriza por tener un nombre, la ciudad de formación y la cantidad de integrantes.

1. Realice el modelo de clases. Implemente las clases con sus atributos y métodos para obtener/modificar el valor de los mismos, además de los constructores necesarios.

2. Implemente la posibilidad de poder agregar artistas solistas y bandas a una discográfica.

3. Implemente los métodos necesarios (en las clases que correspondan) para poder conocer:

   **i.** La cantidad de bandas representadas por una discográfica.

   **ii.** Dado un instrumento, la cantidad de artistas solistas que tocan dicho instrumento.

   **iii.** El nombre de la banda con más integrantes.

   **iv.** La cantidad de bandas formadas en la misma ciudad que la discográfica.

**Notas:**

- Los métodos NO deben imprimir nada, solo devolver lo solicitado.

- Piense que todo lo pedido debe poder ser obtenido a partir de una discográfica, es decir: si en la función main se instancia una discográfica, a partir de ella se debe poder conocer/hacer todo lo solicitado.

**Parcial 4.**

**1.**

**a).** Genere una clase para representar espectadores, que se caracterizan por nombre, DNI y edad. Incorpore getters y setters, y un constructor que permita iniciar el espectador con un nombre, DNI y edad recibida.

**b).** Generar una clase para representar funciones de teatro. Una función de teatro se caracteriza por mantener: Titulo, Fecha, Hora, una estructura que representa la sala (20 filas y 10 butacas por fila) que almacenará los espectadores registrados para la función, y una estructura que almacena para cada fila la cantidad de butacas ocupadas.

**i-** Defina métodos getters/ setters para los atributos que considere adecuado.

**ii-** Implemente un constructor que inicie la función de teatro con un título, fecha y hora recibidos por parámetro. La sala inicialmente debe estar vacía.

**iii-** Incorpore los siguientes métodos:

**#validarFila:** recibe un nro. "F" y devuelve un boolean que indica si "F" es un nro. de fila valida o no.

**#hayButacaLibreEnFila**: recibe un nro. de fila valido "F" y devuelve un boolean que indica si hay una butaca libre en la fila "F" o no.

**#agregarEspectadorAFila:** recibe un nro. de fila valido "F" y un espectador "E", y agregar a "E" en la primera butaca libre de la fila "F", debiendo retornar el nro. de butaca asignado.

**#calcularBuracasLibres**: calcula y devuelve la cantidad total de butacas libres para la función.

**#calcularEdadPromEspectadores**: calcula y devuelve la edad promedio de los espectadores registrados.

**#estaRegistradoEspectador:** recibe un nro. de DNI "D" y devuelve un boolean que indica si existe un espectador registrado en la sala con DNI igual a "D".

2. Escriba un programa que instancie una función de teatro para la obra "Cazafantasmas" con hora "20:00" y fecha "22-10-2018". Luego simule la venta de localidades de la siguiente manera. Leer DNI de personas hasta ingresar el 0 o hasta que no queden butacas libres para la función. A la persona se le solicita además nombre, edad, y el nro. de fila que prefiere. En caso que el nro. de fila sea válido, exista una butaca libre en esa fila y no exista un espectador registrado en la función con ese DNI, agregar a la persona como espectador en la fila solicitada e imprimir el nro. de butaca asignado. Caso contrario, informar el error correspondiente. Al

finalizar la venta informar la cantidad de butacas libres y edad promedio de los espectadores registrados para la función.

**Parcial 5.**

**1)**. La UNLP desea un sistema que le permita administrar diferentes subsidios (de estadía y de bienes) pedidos por sus investigadores. De todo subsidio se conoce el nombre y apellido del investigador, el plan de trabajo actual (un String) y la fecha solicitada (un String). De los subsidios de estadía se conoce el lugar de destino, el costo del pasaje de ida y vuelta, la cantidad de días del viaje y el monto de hotel por día. De los subsidios de bienes se poseen los detalles de cada bien solicitado: descripción, tipo de bien (de uso o de consumo), cantidad y costo por unidad.

**2)**. Implemente el modelo de clases teniendo en cuenta:

**a)-** Un subsidio de estadía solo debería poder construirse con el nombre y apellido del investigador, la fecha, el lugar y la cantidad de días del viaje.

**b)-** Un subsidio de bienes solo debería poder construirse con el nombre y apellido del investigador, la fecha y la cantidad de bienes a pedir.

**c)-** La posibilidad de agregar un bien al subsidio de bienes, estableciendo la descripción, tipo de bien, cantidad y costo por unidad.

**d)-** La posibilidad de poder modificar cualquier atributo descripto en el punto 1).

**3)**. Implemente los métodos:

**#double montoTotalSolicitado();**

- devuelve el monto total del subsidio:

  // Monto_total_subsidio_viajes = costo_pasaje + (monto_hotel_por_dia * dias_viaje).

  // Monto_total_subsidio_bien = sumatoria (costo_bien_i * cantidad_bien_i).

**#String toString();**

- devuelve un String con los detalles del subsidio:

  // nombre del investigador, plan de trabajo, la fecha, monto total y el ítem más caro: viaje o la estadía en un tipo de subsidio o el ítem más caro en el otro.

**4).** Escriba un programa principal que instancie dos subsidios (uno de cada tipo), establezca todos los datos necesarios para cada uno de ellos y luego muestre por consola la descripción de cada uno.

**Parcial 6.**

Un teatro desea un sistema para poder manejar la venta de localidades de las obras que se brindan. El teatro se caracteriza por tener un nombre, una dirección y el nombre del dueño. El teatro además tiene información de sus obras (a lo sumo 5). Una obra se caracteriza por tener un nombre, el nombre del director y los actores que actúan en dicha obra (a lo sumo 100). Un actor tiene un nombre, un apellido, su género y la edad en años.

**A.** Realice el modelo de clases. Implemente las clases con sus atributos y métodos para obtener/modificar el valor de los mismos, además de los constructores necesarios.

**B.** Implemente la posibilidad de poder agregar obras al teatro y actores a una obra.

**C.** Implemente los métodos necesarios (en las clases que correspondan) para poder conocer:

I- Dado el nombre de una obra, el nombre del director.

II- La obra con menor cantidad de actores.

III- El nombre de la obra en donde actúa un determinado actor (conociendo su nombre y apellido).

IV- El nombre y apellido del actor con más edad entre todos los actores de todas las obras.

**Notas**:

- Los metodos NO deben imprimir nada, solo devolver lo solicitado.

- Piense que todo lo pedido debe poder ser accedido a partir de un teatro, es decir: si en la función main instancia un teatro, a partir de él se debe poder conocer/hacer todo lo solicitado.

**Parcial 7.**

1- Representar los sigs. tipos de finales: teórico y teórico-practico. Los finales teóricos se caracterizan por mantener: nombre del alumno, fecha y una estructura que almacena el resultado (true: "bien", false: "mal") para cada una de sus 5 preguntas (numeradas de 1 a 5). Los finales teórico-práctico son finales teóricos que tienen además un puntaje practico (nro. entre 0 y 5).

**A).** Realice el modelo de clases. Implemente las clases con sus atributos y métodos para obtener/modificar el valor de aquellos que considere adecuado.

**B).** Implemente un constructor para los finales teóricos que reciba datos para iniciar nombre y fecha; y un constructor para los finales teórico-práctico que reciba datos para iniciar nombre, fecha y puntaje practico.

**C).** Ambos tipos de finales deben responde a los mensajes:

**#cargarResultadoPregunta:** recibe un nro. de pregunta valido y su resultado, y carga el resultado en la estructura.

**#verResultadoPregunta:** recibe un nro. de pregunta valido y retorna su resultado.

**D).** Ambos tipos de finales deben responder al mensaje **#calcularNotaFinal** que retorna un nro. según se indica:

- En los finales teóricos, la nota final se obtiene de sumar 2 puntos por cada pregunta con resultado "bien".

- En los finales teoricos-practicos, la nota final se obtiene de la fórmula: **(nota final del teórico/2) + puntaje practico.**

**E).** Ambos tipos de finales deben responder al mensaje **#estaAprobado** que retorna un boolean según se indica:

- Los finales teóricos se aprueban cuando su nota final es superior a 4.

- Los finales teórico-práctico se aprueban cuando su nota final es superior a 6.

**F).** Ambos tipos de finales deben responder al mensaje toString que retorna un String con el formato:

"Alumno: N - Nota Final: X - ¿Aprobó?: Y", donde N es el nombre, X es la nota final e Y indica si aprobó o no.

2- Realice un programa que instancie un final teórico de "Norma García" con fecha "22/10/2018". Leer desde teclado el resultado para las 5 preguntas y cargarlas a este final. Luego instancie un final teórico-practico de "Pablo Pérez" con fecha "22/10/2018" y puntaje practico leído desde teclado. Cargue de manera similar el resultado para las 5 preguntas. Al finalizar imprima la representación de cada final.

**Parcial 8.**

1. Representar los sigs. tickets emitidos por un supermercado: normal y con descuento. Los tickets normales se caracterizan por mantener: núm. de ticket, fecha y los productos comprados (a lo sumo 100). Los productos se caracterizan por tener código, descripción y precio. Los tickets con descuento son tickets normales que además tienen un porcentaje de descuento a aplicar.

a)- Realice el modelo de clases. Implemente las clases con sus atributos y métodos getters/setters que se adecuen al problema. Provea constructores para iniciar; los tickets normales a partir de un numero de ticket y fecha; los tickets con descuento a partir de un numero de ticket, fecha y porcentaje de descuento; los productos a partir de un código, descripción y precio.

b)- Ambos tickets deben responder a los mensajes:

> **#cargarProducto**: recibe un producto y lo carga al ticket.
>
> **#estaCompleto**: devuelve un boolean que indica si ya se cargaron los 100 productos.
>
> **#verCantDeProds:** devuelve la cantidad de productos cargados.

c)- Ambos tickets deben responder al mensaje **#calcularPrecioFinal** que calcula y retorna el precio final:

- En los tickets normales, el precio final es la suma de los precios de los productos cargados.

- En los tickets con descuento, el precio final es la suma de los precios de los productos cargados al que se le descuenta un monto (determinado por el porcentaje de descuento).

d)- Los tickets con descuento deben responder al mensaje **#calcularAhorroFinal** que calcula y retorna el monto descontado en el precio final (determinado por el porcentaje de descuento).

e)- Ambos tickets deben responder al mensaje **#mostrarResumen** que imprime el número de ticket, el código, descripción y precio de cada producto cargado, y el precio final; y para los tickets con descuento muestra además el ahorro final.

f)- Ambos tickets deben responder al mensaje **#abonableConDebito** que devuelve un boolean que indica si el ticket puede abonarse con debito, esto es posible cuando su precio final supera los $200.

2. Realice un programa que instancie un ticket normal, con numero 1 y fecha "23/05/2019". Leer desde teclado la información de productos y cargarlos al ticket, hasta ingresar código de producto 0 o completar el ticket. Luego, instancie un ticket con descuento con número 2, fecha "23/05/2019" y porcentaje de descuento leído de teclado. Cargue de manera similar productos a este ticket. Al finalizar, muestre el resumen de cada ticket e imprima si son abonables con débito.

**Parcial 9.**

Un supermercado realizará un sondeo de opinión dirigido a los distintos tipos de clientes de sus sucursales.

1)- Genere una clase para representar urnas de opinión, que se caracterizan por contabilizar las opiniones positivas y las opiniones negativas recibidas de clientes. Incorpore un constructor que inicie la urna vacía (sin opiniones). Provea getters/setters adecuados para el problema; y métodos que permitan:

- Registrar una opinión positiva.

- Registrar una opinión negativa.

- Devolver una representación en formato String del estilo: “X|Y” siendo X la cantidad de opiniones positivas y la cantidad de opiniones negativas.

2)- Genere una clase Sondeo de Opinión. Un sondeo se caracteriza por conocer la pregunta a realizar y por mantener una urna para cada tipo de cliente (0..2) y sucursal (0..3). Incorpore un constructor que inicie el sonde con una pregunta recibida. Inicialmente todas las urnas deben estar vacías. Provea getters/setters adecuados para el problema; y métodos que permitan:

- Validar un tipo de cliente recibido.

- Validar un número de sucursal recibido.

- Devolver la urna correspondiente a un tipo de cliente y número de sucursal validos recibidos.

- Mostrar el resultado del sondeo, esto es imprimir en consola la representación de cada urna.

- Calcular y devolver el tipo de cliente más exigente, esto es aquel que tuvo mayor cantidad total de opiniones negativas (teniendo en cuenta todas las sucursales).

3)- Realice un programa que instancie un sondeo para la pregunta “Es adecuada la limpieza”. Simule la encuesta a clientes del siguiente modo: lea tipo de cliente, numero de sucursal y opinión (entero) hasta ingresar opinión 0. Valide el tipo de cliente y número de sucursal y luego registre una opinión en la urna correspondiente (si opinión es positivo, registrar una opinión positiva; si es negativo, registrar una opinión negativa). Al finalizar, mostrar el resultado del sondeo, el número de sucursal ganadora y el tipo de cliente más exigente.

**Parcial 10.**

**1.** La UNLP desea un sistema que le permita administrar a sus proyectos, investigadores y subsidios. Un proyecto tiene un nombre, un código, un director y los investigadores que participan en el proyecto (hasta 50 como máximo). De cada investigador se desea saber su nombre, apellido, categoría (1 a 5) y su especialidad. Cualquier investigador puede pedir hasta un máximo de 5 subsidios. De cada subsidio se desea saber el monto pedido, el motivo y se fue otorgado o no.

**2.** Implemente el modelo de clases teniendo en cuenta:

a) Un proyecto solo debería poder construirse con el nombre y el código

b) Un investigador solo debería poder construirse con nombre y apellido, categoría y especialidad

c) Un subsidio solo debería poder construirse con el monto solicitado y el motivo. Un subsidio siempre se crea en estado no-otorgado.

**3.** Implemente métodos (en las clases donde corresponda) que permitan:

d) void agregarInvestigador(unInvestigador); // agrega un investigador a un proyecto.

e) void agregarSubsidio(unSubsidio); // agrega un subsidio a un investigador.

f) double dineroTotalOtorgado(); // devuelve la cantidad de dinero de todos los subsidios otorgados a todos los // investigadores de un proyecto.

g) int cantidadDeSubsidios(String nombre_y_apellido); // devuelve la cantidad de subsidios (otorgados o no) solicitados por el //investigador llamado "nombre_y_apellido".

h) void otorgarTodos(String nombre_y_apellido); // otorga todos los subsidios pendientes que tiene el investigador // llamado nombre_y_apellido

i) String toString(); // Devuelve un string que tiene el nombre del proyecto, su código, el nombre y apellido del // director, el total de dinero otorgado y el nombre y apellido de cada investigador. Para cada // investigador, además, se debe agregar la categoría del mismo y el dinero de sus // subsidios otorgados.

**NOTA**: puede crear todos los métodos auxiliares que considere necesario.

Escriba un programa principal que instancie un proyecto con un director y dos investigadores. Instancie dos subsidios y asígneselos al primero de ellos, luego imprima todos los datos del toString pedido en el punto 3.f).

**Parcial 11.**

**1.** Queremos representar un equipo de futbol 7:

- Un equipo tiene nombre, un entrenador y 7 jugadores (que juegan en posiciones numeradas de 1 a 7).
- De cualquier entrenador y jugador, se tiene nombre, DNI y sueldo básico. Además, del entrenador se tiene la cantidad de campeonatos dirigidos, y del jugador, la edad y descripción de sus habilidades.

**A)-** Realice el modelo de clases. Implemente las clases con sus atributos y métodos getters/setters que se adecuen al problema. Provea constructores para iniciar: al entrenador y al jugador a partir de toda su información recibida, al equipo a partir de un nombre y un entrenador (y sin jugadores).

**B)-** El entrenador y jugador deben permitir:

- Aumentar su sueldo básico en un porcentaje recibido.
- Calcular su sueldo a cobrar. El entrenador cobra el sueldo básico al que se adiciona $5000 por campeonatos dirigidos. El jugador cobra el sueldo básico al que se descuenta el 10% si excede los 30 años.
- Retornar su representación String según el ejemplo: “Nombre: … DNI: … Sueldo a cobrar: …”.

**C)-** El equipo debe permitir:

- Validar una posición de juego recibida.
- Asignar a un jugador en una posición de juego recibida.
- Retornar al jugador que se encuentra en una posición de juego recibida.
- Retornar un String con las posiciones de juego sin jugador asignado.
- Retornar el sueldo a cobrar por el plantel asignado (entrenador y jugadores).
- Retornar un String con la representación del equipo, compuesta por: el sueldo a cobrar por el plantel, la representación del entrenador y de sus jugadores asignados (incluyendo su posición de juego).

**2.** Realice un programa que instancie un equipo de futbol 7, con nombre “Papi Futbol” y un entrenador cuyos datos se leen de teclado. Cargue 4 jugadores en distintas posiciones de juego (la información se lee de teclado). Luego, imprima la representación del equipo, las posiciones de juego no asignadas, y la descripción de habilidades del jugador en posición de juego 2.

**Parcial 12.**

**A.** Generar una clase para representar canciones, que se caracterizan por título, interprete y duración en segundos. Implemente un constructor que inicie el objeto con título, interprete y duración recibidos y los métodos getters/setters y toString (ej. “Bohemian Rhapsody-Queen-360 segundos”).

**B.** Generar una clase Reproductor MP3. Un reproductor MP3 almacena a lo sumo 100 canciones y mantiene el numero de canción en reproducción (esto es, la posición dentro del almacenamiento de la canción que se está reproduciendo).

Implemente un constructor que inicie el reproductor MP3 sin canciones, y los siguientes métodos:

- quedaEspacio: retorna un boolean que indica si hay espacio para almacenar una cancion más.
- cargarCancion: recibe una canción y la almacena en el reproductor MP3.
- iniciarReproduccion: inicia el numero de canción en reproducción en 0.
- reproducir: imprime en consola, titulo, interprete y duración de la canción en reproducción.
- avanzar: actualiza el numero de canción en reproducción para pasar a la siguiente o volviendo a 0 en caso de haberse reproducido todas las canciones almacenadas.
- calcularDuracionTotal: calcula y devuelve la duración en segundos total entre todas las canciones almacenadas.

**C.** Realice un programa que instancie un reproductor MP3. Cárguele canciones, con datos leídos de teclado, hasta quedarse sin espacio o hasta leer el titulo de canción “ZZZ”. Luego, lea la cantidad de canciones que quiera escuchar (N), inicie la reproducción y reproduzca N canciones, avanzando a la próxima luego de cada reproducción. Al finalizar muestre la duración total entre todas las canciones almacenadas en el reproductor.