Trabajo Práctico Integrador Nro 1.

Unidad 3: Paradigma orientado a objetos.

Objetivos:

- Integrar los contenidos prácticos correspondientes a la programación orientada a objetos.
- Reforzar las destrezas necesarias para resolver una situación problemática utilizando la programación orientada a objetos en Smalltalk, utilizando Pharo 8.0 como implementación.
- Modelar una solución de un problema mediante el paradigma orientado a objetos.

Los temas para evaluar en este trabajo práctico son los que se detallan a continuación: Objetos y mensajes en Smalltalk. Uso de ventanas en Smalltalk. Mensajes unarios, binarios y de palabras claves. Bloques. Composición de clases. Abstracción. Encapsulamiento. Herencia. Polimorfismo. Colecciones.

Caso de Estudio:

Un teatro del interior de la provincia tiene planes de disponer para sus clientes, un sistema informático para la gestión de funciones, habilitación de entradas y gestionar las ventas de entradas.
En principio se requiere implementar un prototipo de un programa orientado a objetos en Smalltalk, aplicado a la gestión de funciones, habilitación y ventas de entradas para una sola obra de teatro.

Durante las etapas de análisis, se identificaron las siguientes clases:
ObraTeatro, representa la obra de teatro para la cual se habilitarán posteriormente algunas funciones. Tiene los siguientes atributos:

- ***título*** que es una cadena de caracteres, que representa el nombre completo de la obra de teatro.
- ***Dirección*** que representa quien está a cargo de la dirección de la obra de teatro.
- ***funciones*** representa a todas las funciones que se hayan habilitado. Las funciones se deberán mantener ordenadas en la obra de teatro, en forma cronológica ascendente de acuerdo a su fecha (desde la más antigua a la más reciente).

Dirección, representa a quien está a cargo de dirigir la obra de teatro. Tiene como atributos, los siguientes:

- ***nombre***, es una cadena de caracteres que representa el nombre de la persona a cargo de la dirección.
- ***apellido***, es una cadena de caracteres que representa el apellido de la persona a cargo de la dirección.
- ***porcentajeVentas***, es un valor tipo flotante entre 0 y 100, que representa el porcentaje del importe de las ventas de las entradas a las diferentes funciones de la obra, que se estableció para quien está a cargo de la dirección de esta.

Función, está asociada a una determinada obra y corresponde a la programación de la representación pública del espectáculo en sí mismo en el escenario del teatro. Tiene los siguientes atributos:

- ***código***, es un valor numérico único (no se deberá repetir para diferentes funciones de la misma obra).
- ***fecha***, representa la fecha en la que se ha programado la función para llevarse a cabo.
- ❖ ***entradas***, representa todas las entradas que se habiliten a la venta para dicha función. Las entradas se deberán mantener ordenadas en la función asociada, de acuerdo con el orden en el que vayan siendo habilitadas (de acuerdo al orden de inserción de las mismas en la colección).
-

Entrada, representa la habilitación para una persona o grupo de personas a ser ubicadas dentro del teatro, según el tipo de entrada, durante una función en particular. Las entradas pueden ser de dos clases: vip y platea, dependiendo de la zona del teatro en la que estén los asientos para que disfruten de la obra los espectadores, y de la posibilidad de brindárseles algún servicio extra, por ejemplo, de catering. Una entrada tiene los siguientes atributos:

- ***importeBásico***, representa un importe básico en pesos de la entrada. Cabe aclarar que el importe final de la entrada dependerá del tipo de entrada del cual se trate.
- ***vendida***, representa si la entrada fue vendida o no (true: significa que dicha entrada se vendió, mientras que false significa que aún no se vendió). Por defecto, cuando recién se habilita una nueva entrada para una función que se está programando, la misma tiene el atributo ***vendida*** en false. Cuando la entrada es vendida, el atributo ***vendida*** recién pasará a true.

EntradaVip es una clase derivada de **Entrada**, y representa a una clase de entrada con ciertos privilegios, por ejemplo, la de ocupar un palco exclusivo del teatro, y de poder asistir en forma individual o en grupo, tal que la cantidad total de personas del grupo sea menor o igual a la cantidad de asientos disponibles del palco. También, se ofrece, a decisión de quien compre la entrada, de un servicio de catering para la persona o el grupo de personas que ocupe el palco correspondiente. Cabe aclarar que el importe de la entrada vip es independiente de la cantidad de personas que como grupo asistan realmente al momento de la función correspondiente. El importe final de la entrada vip sólo dependerá del importe básico de la entrada, de la cantidad de asientos habilitados que tenga el palco y de si se contrata o no el servicio de catering. Por ejemplo, si alguien compra una entrada vip asociada a un palco con 4 asientos, y no contrata servicio de catering, el importe final será el mismo si alguien asiste en forma individual, o acompañado por una, dos o hasta tres personas. Es decir, si el palco asociado a una entrada vip tiene habilitados 4 asientos, y sólo asisten 2 personas, en precio de la entrada es único y el mismo siempre para todo el grupo. Los atributos que agrega entonces una EntradaVip son los siguientes:

- ***servicioCatering***, que representa si se contrata o no, un servicio de catering. En caso de contratarse este servicio, el valor del atributo será true, y en caso contrario, será false. Por defecto será false.
- ***palco***, representa el palco del teatro asignado para la entrada vip correspondiente.

EntradaPlatea es una clase derivada de **Entrada**, y representa a una clase de entrada sin privilegios. Sólo se le brinda el derecho a ocupar una butaca individual en la platea del teatro durante la función correspondiente. El único atributo que agrega una EntradaPlatea es:

- ***nroButaca***, que representa el número de la butaca asignada para dicha entrada en la platea. Si bien el teatro dispone de 99 butacas en platea, numeradas del 1 al 99, inclusive, pero por motivos del

distanciamiento social, sólo se podrán asignar a las entradas de platea, las butacas del teatro con número impar. Cabe aclarar también que el teatro dispone en total de diez filas de butacas en platea: desde la 1 a la 9, ocupan la fila 1, desde la 10 a la 19, la fila 2, desde la 20 a la 29, la fila 3, y así sucesivamente hasta las butacas 90 a 99 que ocupan la fila 10.

Palco, representa a uno de los balcones disponibles en el teatro, que podrá ser ocupado por quien o quienes adquieran la entrada vip correspondiente para determinada función. Para cada palco, según ciertas políticas internas al teatro, se habilita una cantidad determinada de asientos para los espectadores. Los atributos de un palco son:

- ***código***, el cual es una letra desde la 'A' a la 'F', que representa unívocamente a cada uno de los 6 palcos del teatro.
- ***cantidadAsientos***, representa la cantidad de asientos que en forma predeterminada ya tiene el palco para los espectadores que asistan como parte del grupo que adquiera la entrada vip. Sólo se adquiere una única entrada vip para todas las personas del grupo. Cabe aclarar que cuando se vende una entrada vip, se está permitiendo que se use el palco asociado, con una cantidad real de personas igual o menor a la cantidad de asientos habilitados del palco. Es decir, si el palco tiene una cantidad de asientos habilitados igual a 4, sólo podrán ocupar el palco al momento de la función, una cantidad de personas menor o igual a 4 (4 o menos sí, pero no más de 4).

Requerimientos:

- Agregar una **función** a una obra de teatro. Para poder agregar la función, se deberá verificar que no exista ninguna otra función ya agregada que tenga el mismo código de la función que se intenta agregar. En caso de no existir ninguna función con ese código, se procederá a agregar la misma, y el comportamiento deberá retornar un true. En caso contrario, no se deberá agregar, y se deberá retornar un false.
  El método deberá recibir como colaborador la función que se desea habilitar para la obra de teatro.
  Método sugerido en Smalltalk: **addFuncion:**
  Se podrán agregar también todos los métodos que considere necesario para la implementación de este comportamiento.

- Habilitar una **entrada** a una función. Se deberá poder habilitar una entrada de cualquier clase (sin distinción), a una función en particular cuyo código se especifique. En caso de no encontrarse ninguna función habilitada con el código especificado, no se podrá habilitar dicha entrada, y se deberá retornar un false. En caso de que se haya encontrado una función con el código especificado, se podrá insertar la entrada dentro de las que tiene asociada dicha función, y en tal caso se deberá retornar un true. Toda nueva entrada que se habilite deberá tener false asignado por defecto en el atributo ***vendida***.

El método deberá recibir como colaboradores: el código de la función y la entrada que se requiere habilitar para dicha función. Métodos sugeridos en Smalltalk pueden ser: **addEntrada: aFuncion:** o **habilitarEntrada: aFuncion:**

Se podrán agregar también todos los métodos que considere necesario para la implementación de este comportamiento.

- **Consideración importante:** una vez validado que exista una función con el código especificado, no será requerido que se valide, por ejemplo, para el caso de una EntradaPlatea, que no exista el mismo valor de nroButaca, para otra entrada de platea ya habilitada para dicha función. De la misma forma, tampoco se requiere que se valide en el caso de una EntradaVip, que no exista el mismo valor de código de palco para otra entrada vip ya habilitada para dicha función. **Estas validaciones, sí serán requeridas en la instancia de recuperación de este TP.**

- Vender una entrada de platea para una función cuyo código se especifique. Para llevar adelante este comportamiento también se deberá encontrar dentro de las funciones habilitadas, alguna cuyo código sea igual al especificado como colaborador. En caso de no encontrarse ninguna función con dicho código, directamente se deberá retornar la siguiente cadena de caracteres: '*Función no encontrada*'. En caso de haberse detectado una función cuyo código sea igual al código de función especificado como colaborador, se procederá a detectar la primera entrada de platea no vendida, disponible para dicha función. En caso de encontrarse alguna entrada de platea habilitada se procederá a modificar a false el valor del atributo ***vendida*** de dicha entrada; y se deberá retornar una cadena de caracteres con los siguientes datos: '*código de la función, fecha de la función, número de butaca asignado, e importe final de dicha entrada*'. En caso de no haberse encontrado ninguna entrada de platea disponible a la venta, para dicha función se deberá retornar la siguiente cadena de caracteres: '*No se encontró ninguna entrada de platea disponible para la venta*'.

  El método deberá recibir como colaborador: el código de la función. Método sugerido en Smalltalk: **venderEntradaPlatea:**

  Se podrán agregar también todos los métodos que considere necesario para la implementación de este comportamiento.

- Vender una entrada vip para una función cuyo código se especifique. Para llevar adelante este comportamiento también se deberá encontrar dentro de las funciones habilitadas, alguna cuyo código sea igual al especificado como colaborador. En caso de no encontrarse ninguna función con dicho código, directamente se deberá retornar la siguiente cadena de caracteres: '*Función no encontrada*'. En caso de haberse detectado una función cuyo código sea igual al código de función especificado como colaborador, se procederá a detectar la primera entrada vip no vendida, disponible para dicha función. En caso de encontrarse alguna entrada vip habilitada se procederá a modificar a false el valor del atributo ***vendida*** de dicha entrada, y también se deberá modificar el valor del atributo ***servicioCatering*** de acuerdo con el segundo colaborador recibido; y se deberá retornar una cadena de

caracteres con los siguientes datos: *'código de la función, fecha de la función, acepta servicio de catering (si/no), código de palco asignado, e importe final de dicha entrada'*. En caso de no haberse encontrado ninguna entrada vip disponible a la venta, para dicha función se deberá retornar la siguiente cadena de caracteres: *'No se encontró ninguna entrada vip disponible para la venta'*.
El método deberá recibir como colaboradores: el código de la función, y un valor booleano (true o false) que represente si se acepta o no el servicio de catering. Método sugerido en Smalltalk: **venderEntradaVip: aceptoServicioCatering:**
Se podrán agregar también todos los métodos que considere necesario para la implementación de este comportamiento.

- Conocer el importe total de todas las **entradas vendidas** (sin distinción de clase de entradas) correspondiente a aquellas funciones cuyas fechas se encuentran dentro del rango de fechas especificados como colaboradores: fechaDesde y fechaHasta.
  El método deberá recibir como colaboradores: la fechaDesde, y la fechaHasta. Método sugerido en Smalltalk: **importeTotalVentasFechaDesde: fechaHasta:**
  Se podrán agregar también todos los métodos que considere necesario para la implementación de este comportamiento.

- Generar un listado de aquellos números de butacas de la platea para las que aún no se vendieron entradas para una función cuyo código se especifica. El listado se deberá mostrar los números de las butacas ordenados en forma ascendente por número de butaca, y cada número se deberá separar del otro mediante un guion medio. Por ejemplo: '1 - 9 – 15 – 23 – 29 – 37 – 43 – 55 – 91 - 99'
  El método deberá recibir como colaborador: el código de la función. Método sugerido en Smalltalk: **listadoOrdenadoNroButacasDisponiblesParaFuncion:**
  Se podrán agregar también todos los métodos que considere necesario para la implementación de este comportamiento.

- Definir los comportamientos polimórficos importeFinal de las diferentes clases de entradas.
  El método no deberá recibir ningún colaborador. Método sugerido en Smalltalk para todas las clases involucradas en la jerarquía: **importeFinal**
  A continuación, se especifica el comportamiento para cada una de las clases derivadas:
  - El importe final de una EntradaPlatea se calcula de la siguiente manera: si la butaca se encuentra ubicada hasta la quinta fila inclusive (es decir, butacas que van desde el número 1 al número 49) el importe final corresponde al importe básico más un 25 % del mismo. En caso contrario (desde la sexta fila hasta la última fila, inclusive), el importe final corresponde al importe básico más un 10 % del mismo.

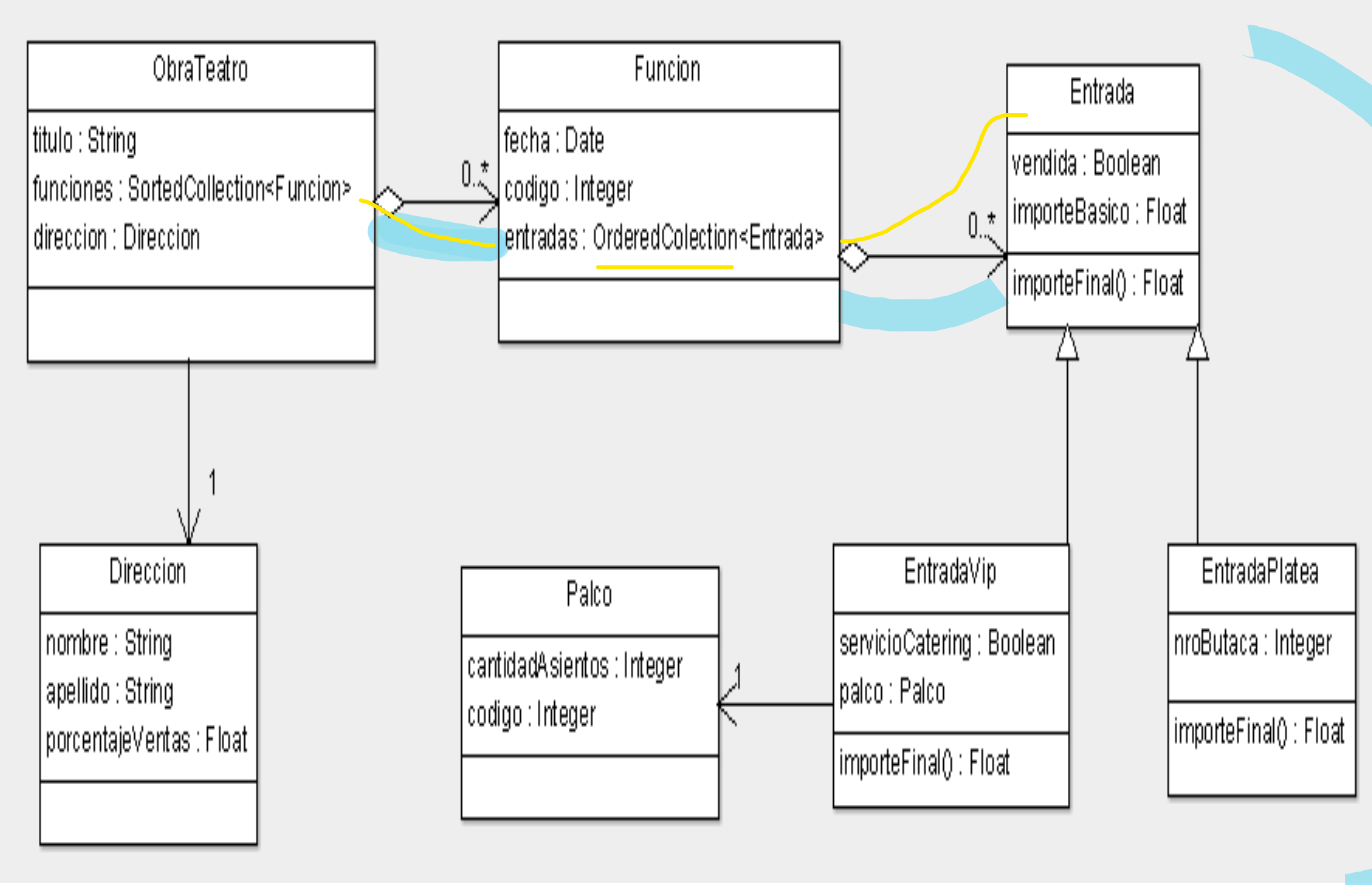
- El importe final de una EntradaVip se calcula de la siguiente manera: se multiplica el importe básico por la cantidad de asientos que tenga habilitados el palco. Y en caso de tener contratado el servicio de catering, al anterior importe se le incrementa un 30 % más (independientemente de la cantidad de personas que realmente asistan al momento de la función).

Se podrán agregar también todos los métodos que considere necesario para la implementación de estos comportamientos.

No obstante, se podrán implementar, de acuerdo con los requerimientos anteriormente especificados, otros métodos polimórficos, además del método importeFinal.

❖ En todas las clases se deberán implementar los métodos: initialize, de acceso, de modificación, y asString. Además, se deberán también implementar todos los métodos que se consideren necesarios para llevar adelante todos los comportamientos requeridos. Por ejemplo: los comportamientos para buscar una función por código, etc.

A continuación, se muestra un diagrama de clases, incompleto que se podrá utilizar como modelo para iniciar el desarrollo.



Se sugiere dividir en el trabajo en etapas, de manera que pueda ir resolviendo los requerimientos solicitados a medida que avanzan las clases teórico-prácticas de la materia, tal como se indica a continuación:

- **Etapa 1:** desarrollar la clase Direccion.
- **Etapa 2:** desarrollar la clase Entrada, EntradaPlatea y EntradaVip.
- **Etapa 3:** desarrollar el resto de las clases.

Únicamente para instancia de recuperación

En caso de requerir el acceso a la instancia de recuperación, además de corregir todo lo que corresponda de la instancia original, se deberán agregar los siguientes requerimientos adicionales:

1) Agregar las siguientes validaciones adicionales al comportamiento **habilitarEntrada: aFuncion:**, que permite habilitar una entrada determinada a una función cuyo código se especifica. Para implementar estas validaciones adicionales, ahora sí se deberá diferenciar entre las diferentes clases de entradas que se esté intentando habilitar.
   Una vez validado que exista una función con el código especificado, se deberá validar, por ejemplo, para el caso de una EntradaPlatea, que no exista el mismo valor de nroButaca, para otra entrada de platea ya habilitada para dicha función. En caso de ya existir, dicha entrada no se deberá insertar a la colección, y se deberá retornar un false.
   De la misma forma, se deberá validar en el caso de una EntradaVip, que no exista el mismo valor de código de palco para otra entrada vip ya habilitada para dicha función. En caso de ya existir, dicha entrada no se deberá insertar a la colección, y se deberá retornar un false.

2) Implementar un comportamiento que calcule el importe que deberá recibir la dirección de la obra de teatro por las ventas de entradas correspondientes a funciones comprendidas dentro de dos fechas que se especifican como colaboradores. Se deberá reutilizar el comportamiento ya implementado: **importeTotalVentasFechaDesde: fechaHasta:**

En cada etapa debe ir generando un Playground con la creación de los objetos y el paso de mensajes correspondiente para probar el modelo desarrollado enviando todas las salidas al objeto Transcript.

Consignas generales:

- Defina adecuadamente la jerarquía de clases, y asigne en cada clase los atributos y métodos que correspondan según el criterio del grupo.
- Realice la codificación completa en Smalltalk de todas las clases involucradas en el diseño de la solución problemática. Debe haber coherencia total entre el diagrama de clases propuesto y la implementación en

Smalltalk. Usted puede reformular el diagrama propuesto, en tal caso deberá presentarlo como parte del trabajo. Siéntase libre de agregar todo el comportamiento extra en las clases que considere necesario.

- Escriba las líneas de código necesarias en la ventana Playground para hacer funcionar el programa y verifique en la ventana Transcript todas las salidas de información generadas.
- Reutilice adecuadamente los comportamientos que sean necesarios.
- Tenga presente la delegación de responsabilidades para cada uno de requerimientos según corresponda.

Criterios de evaluación:

- Identificación de todas las clases involucradas, y asignación correcta de atributos y responsabilidades de cada una.
- Adecuado diseño del diagrama de clases: jerarquía de clases y demás relaciones bien especificadas y representadas, etc.
- Claridad y completitud del diagrama de clases.
- Prolijidad en la codificación en Smalltalk: identificadores, comentarios, envío de mensajes, etc.
- Manejo adecuado de **TODAS** las propiedades esenciales de la programación orientada a objetos en la resolución propuesta.
- Delegación apropiada de responsabilidades en las clases involucradas.
- Reutilización conveniente de los comportamientos implementados.
- Validaciones.
- Correcta utilización de métodos:
  - ✓ unarios;
  - ✓ binarios;
  - ✓ de palabras claves.

  Se deberán utilizar todos estos tipos de mensajes en la resolución propuesta en forma adecuada.
- Elección y uso apropiado de las colecciones en cada caso.
- Variedad de mensajes utilizados de las diferentes colecciones utilizadas, y de otros objetos.
- Uso de mensajes específicos tanto de colecciones como de otros objetos, para cada caso.
- Correcta identificación y usos de métodos polimórficos.
- Generación y visualización correcta de todas las salidas de información.
- Cumplimiento de todas las responsabilidades solicitadas.
- Realización de las pruebas solicitadas en el Playground, instanciando adecuadamente los objetos, y realizando las colaboraciones necesarias. Recuerde que los resultados del envío de los mensajes a los objetos se deberán visualizar en la ventana Transcript.

Tabla de valoración de los ítems evaluados

Nro. de ítem	Ítems o requerimiento a evaluar	Puntaje	Observaciones	Obtenido
1	Implementación de clases, con métodos comunes: initialize, métodos de acceso, modificación, y asString.	20		
2	Definición de métodos polimórficos en la jerarquía Entrada.	15		
3	Comportamientos para agregar una nueva función.	10		
4	Comportamientos para habilitar entradas a una determinada función.	10		
5	Comportamientos para vender una entrada de platea.	10		
6	Comportamientos para vender una entrada vip	10		
7	Comportamientos para conocer el importe total de venta de entradas de funciones comprendidas en cierto rango de fechas.	10		
8	Comportamientos para generar un listado ordenado de menor a mayor de los números de butacas disponibles (sin vender) para una determinada función.	15		
	Total	**100**		

Condiciones de entrega:

- Este trabajo práctico integrador **se deberá realizar en forma grupal**.
- Deberán nombrar el **archivo comprimido** con el siguiente formato: TP1_NroLeg1erIntegranteApellido1erIntegrante_NroLeg2doIntegranteApellido2doIntegrente_NroLeg3erIntegranteApellido3erIntegrante. El orden en el cual deberán colocar los legajos y apellidos cada integrante en el nombre del archivo comprimido será de acuerdo con el orden alfabético de los apellidos de los integrantes, desde la A hasta la Z.
- Se deberá subir una carpeta comprimida que contenga en su interior el archivo .ST correspondiente a la codificación en Smalltalk; un archivo de texto con el código necesario a colocar en la ventana Playground para hacer funcionar el programa.