Trabajo Práctico 1. Programación Orientada a Objetos

Puntuación

Puntaje Total: 100 puntos

Aprobación: 60 puntos

Fecha de entrega: **24/04/2024 - 23:55 Hs.**

Condiciones de entrega

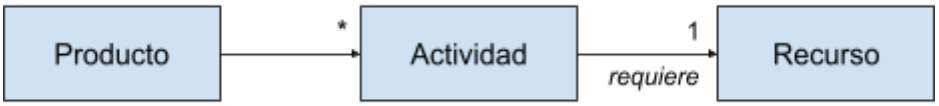
1. El presente trabajo práctico deberá resolverse en grupo de hasta 3 (tres) personas.
2. Entrega: Se realizará por medio del Campus Virtual de la UTN, en la tarea correspondiente al TP 1. La extensión del archivo será .zip o .rar, de acuerdo al programa de compresión usado. El nombre del archivo se consigue concatenando un prefijo del número del TP con los apellidos de los integrantes separados por guiones (Ej: Pérez y García, el nombre será `tp1-garcia-perez.zip`). Note que no hay espacios en blanco ni acentos en el nombre de archivo. Dentro del archivo de entrega, deben constar los siguientes:
   - Diagrama de clases conceptual, indicando la jerarquía de clases y relaciones entre las mismas, incluyendo atributos y métodos. Puede utilizar Violet, UMLet, Dia, etc. Entregar en formato pdf. Nombre: `diagrama.pdf`
   - Fuentes Pharo Smalltalk: Se debe crear un paquete para todas las clases del TP, llamado TP1. Realizar un **#fileOut** de este paquete y guardarlo en un archivo `TP1.st`
   - Los casos de prueba se entregarán en archivos de texto, no deben ser capturas de pantalla. Deberán cubrir diferentes resultados que puedan obtenerse según las funcionalidades solicitadas. Se enfatiza que se adjunten casos de prueba que sean claros, válidos y suficientes para poder probar el trabajo. Como sugerencia puede ir acumulando cada prueba en un Playground y luego grabarlo a disco con el nombre `caso-de-prueba-`***`N`***`.txt` , con ***N*** = `1`, `2`, etc.
   - Archivo de texto (`integrantes.txt`) con una línea para cada integrante en la cual figure el nombre del alumno/a y su dirección de email.
3. Penalización por entrega fuera de término: Si el trabajo práctico se entrega después de la fecha indicada, y hasta una semana tarde, tendrá una quita de 15 puntos. No serán recibidos trabajos luego de una semana de la fecha de entrega. Los trabajos que se deban rehacer/corregir fuera de la fecha de entrega tienen una quita de 30 puntos.

Introducción

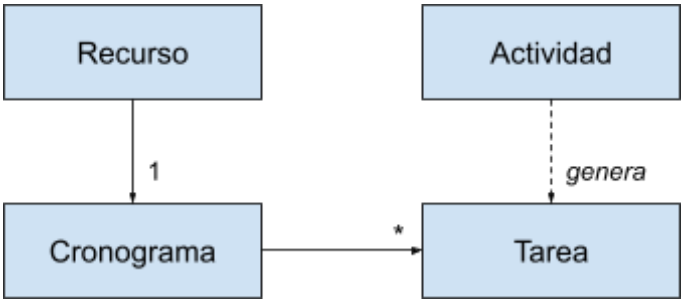
"TU MUEBLES" es una empresa de pequeña escala especializada en la fabricación de muebles a medida. Ante el crecimiento de la demanda durante el último periodo la empresa ha experimentado serios problemas en la planificación de las tareas de producción. Ante esta situación, la empresa ha decidido abordar el desarrollo de un sistema de planificación de la producción. Provisto de la información de las actividades y recursos que se necesitan para fabricar cada producto, el sistema debe generar un cronograma de las tareas a realizar por unidad de tiempo (para cada hora).

Diseño del Sistema

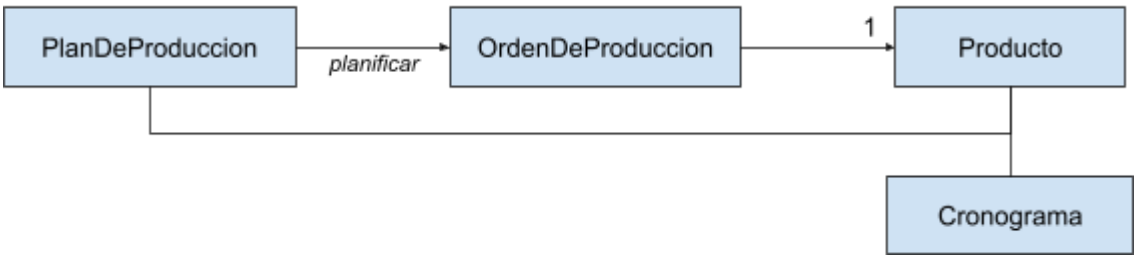
Mediante las clases **Producto**, **Actividad** y **Recurso** es posible definir el proceso de producción para cada producto. Para fabricar un producto deben realizarse un conjunto de actividades, las cuales requieren un recurso cada una. Por simplicidad las actividades se consideran secuenciales, es decir que deben realizarse una a continuación de la otra. Cada actividad especifica por cuánto tiempo (duración en horas) el recurso está asignado a la realización de la actividad.



Se representará una producción concreta mediante las clases **Cronograma** y **Tarea**. Para lograr esto, cada recurso mantiene en su estado interno un cronograma de producción, instancia de la clase **Cronograma**, que es básicamente un listado de las tareas productivas (instancias de **Tarea**) asignadas a dicho recurso por unidad de tiempo (horas). Luego, para producir un dado producto, cada actividad es responsable de generar la tarea respectiva para dicha actividad. Las tareas mantienen la información de la hora de inicio y finalización asignadas.



Ante el ingreso de una orden de producción el sistema debe generar el cronograma de tareas requeridas para satisfacer la orden. También debe ser capaz de recibir una lista de órdenes de producción.

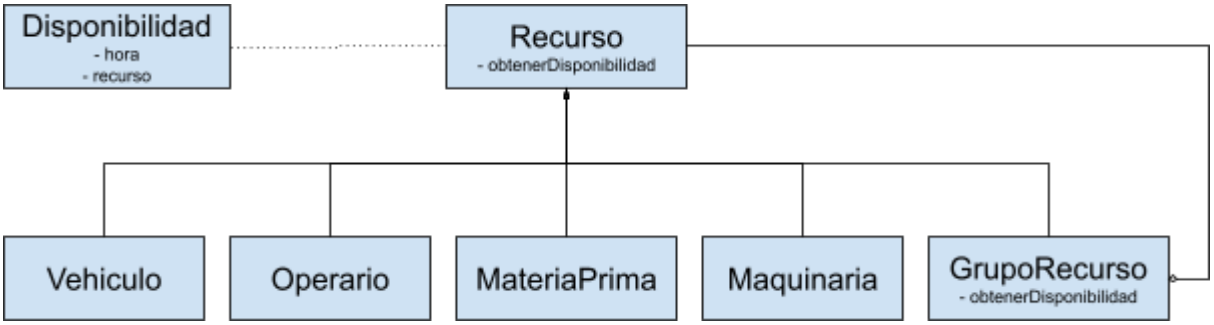


Para esto haremos uso de una instancia de la clase **PlanDeProduccion**, la cual recibe una/s orden/es de producción e invoca los métodos necesarios para generar e imprimir el cronograma de producción.

Implementación

1- Desarrollar la jerarquía de clases para los recursos:

Para el diseño de la clase **Recurso** se utiliza el patrón composite, donde un recurso puede ser un ítem individual (por ejemplo una sierra de cortar o un operario) o un grupo de recursos, por ejemplo el grupo de operarios que se dedican al armado de muebles.



- Todos los recursos responderán al mensaje **obtenerDisponibilidad** (método de instancia), el cual retornará una instancia de la clase **Disponibilidad** que representa la situación del recurso en ese momento. En el caso de un recurso individual, por ejemplo un vehículo, su disponibilidad contiene el mismo recurso y la hora en la que está disponible. **Pero, en el caso de un grupo de recursos, su disponibilidad incluye al recurso específico del grupo de recursos que está disponible primero.** Por ejemplo, cuando se le pregunta a un grupo de armadores por su disponibilidad, la instancia retornada contiene una instancia de Operario específica y la hora en que ese operario está disponible.

- Crear en el Playground las siguientes instancias de recurso (defina todas las variables como temporales):

```
tablasBlancas := MateriaPrima new: 'Tablas Blancas'.
tablasColor  := MateriaPrima new: 'Tablas Color'.
sierraChica  := Maquinaria new: 'Sierra Chica'.
sierraGrande := Maquinaria new: 'Sierra Grande'.
juan  := Operario new: 'Juan'.
```

```
pedro := Operario new: 'Pedro'.
seba  := Operario new: 'Seba'.
ford  := Vehiculo new: 'Ford-MTE845'.
mercedes  := Vehiculo new: 'Mercedes-MPA123'.
armadores := GrupoRecurso new: 'Armadores'.
armadores agregarRecurso: juan.
armadores agregarRecurso: pedro.
armadores agregarRecurso: seba.
```

2- Implementar la clase Producto y generar las siguientes instancias de producto:

```
placard200    := Producto new: 'Placard Estandar 200'.
escritorio100 := Producto new: 'Escritorio Estandar 100'.
placardABC    := Producto new: 'Placard ABC'.
escritorioDEF := Producto new: 'Escritorio DEF'.
```

3- Implementar la clase Actividad, con un constructor tal que permita configurar las actividades de los productos como se muestra a continuación:

```
placard200
 agregarActividad: (Actividad new: 'Preparacion' recurso: tablasBlancas duracionHoras: 2);
 agregarActividad: (Actividad new: 'Corte' recurso: sierraChica duracionHoras: 5);
 agregarActividad: (Actividad new: 'Armado' recurso: armadores duracionHoras: 6);
 agregarActividad: (Actividad new: 'Envio' recurso: mercedes duracionHoras: 2).

escritorio100
 agregarActividad: (Actividad new: 'Preparación' recurso: tablasBlancas duracionHoras: 2);
 agregarActividad: (Actividad new: 'Corte' recurso: sierraGrande duracionHoras: 5);
 agregarActividad: (Actividad new: 'Armado' recurso: armadores duracionHoras: 5);
 agregarActividad: (Actividad new: 'Envio' recurso: mercedes duracionHoras: 2).

placardABC
 agregarActividad: (Actividad new: 'Preparacion' recurso: tablasColor duracionHoras: 3);
 agregarActividad: (Actividad new: 'Corte' recurso: sierraChica duracionHoras: 4);
 agregarActividad: (Actividad new: 'Armado' recurso: armadores duracionHoras: 6);
 agregarActividad: (Actividad new: 'Envio' recurso: ford duracionHoras: 2).

escritorioDEF
 agregarActividad: (Actividad new: 'Preparacion' recurso: tablasColor duracionHoras: 2);
 agregarActividad: (Actividad new: 'Corte' recurso: sierraGrande duracionHoras: 3);
 agregarActividad: (Actividad new: 'Armado' recurso: armadores duracionHoras: 5);
 agregarActividad: (Actividad new: 'Envio' recurso: ford duracionHoras: 2).
```

4- Crear la clase OrdenDeProducción e instanciar las siguientes órdenes (notar que ademas del producto, la orden de producción tiene un código que se usará en la impresión del cronograma).

```
ordenes := OrderedCollection new.
ordenes add: (OrdenDeProduccion new: placard200    codigo: $A).
ordenes add: (OrdenDeProduccion new: escritorio100 codigo: $B).
ordenes add: (OrdenDeProduccion new: placardABC    codigo: $C).
ordenes add: (OrdenDeProduccion new: escritorioDEF codigo: $D).
```

5- Generar la clase PlanDeProduccion y el método *planificar:* que recibe una lista de órdenes.

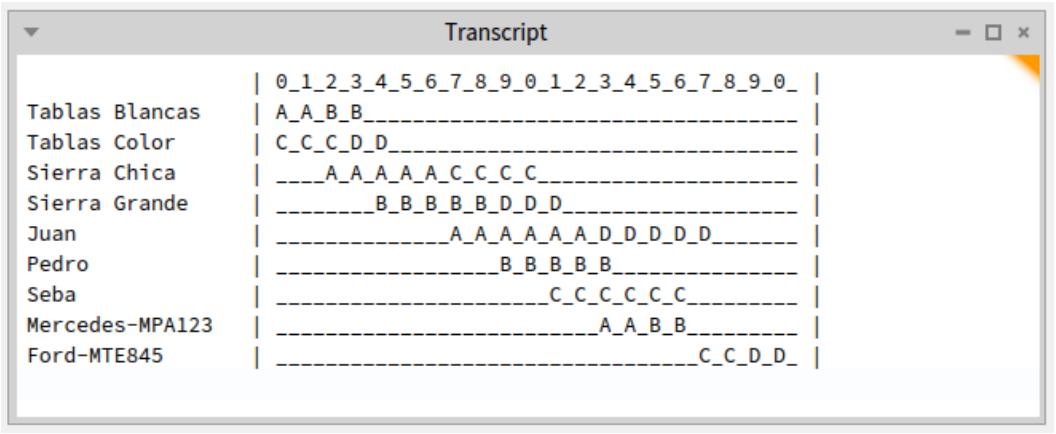
- El método ***planificar:*** debe recorrer el listados de órdenes y para cada producto asociado:
  - Recorrer en orden las actividades y planificar cada una.
  - La planificación de cada actividad deberá:
    - Determinar cuando está disponible el recurso que la actividad requiere.
    - Agregar una tarea al cronograma del recurso, desde la primera hora disponible y por cada hora de duración de la actividad.
    - Recordad que la Clase Actividad es la encargada de crear las tareas. Se sugiere retornar la última tarea planificada, o su finalización, de modo que sea posible programar la siguiente actividad.

```
plan := PlanDeProduccion new.
plan planificar: ordenes.
```

6- Implementar el método *imprimir* en PlanDeProduccion el cual deberá generar la salida en la ventana de Transcript.

- Será necesario que el plan de producción lleve una lista de los recursos utilizados al realizar la planificación.
- Notar que en el cronograma de cada recurso se muestra el código de la orden de producción en la que se debe estar trabajando a determinada hora.
- En la salida ordenar las recursos ascendentemente por la hora más temprana de utilización, en primer lugar, y por nombre, en segundo lugar.

```
plan imprimir.
```



7- Propuesta de casos de prueba adicionales:

Además de implementar las clases y métodos explícitamente solicitados así como todos aquellos métodos auxiliares que considere necesario, de manera que las definiciones del ejemplo cumplan la funcionalidad solicitada, **proponga al menos 2 casos de prueba adicionales**, definiendo otros productos, actividades, recursos, órdenes de producción y plan de producción.

- `caso-de-prueba-1.txt` - El ejemplo planteado a continuación.
- `caso-de-prueba-2.txt` - **Proponer**
- `caso-de-prueba-3.txt` - **Proponer**
- …

Playground completo

```
| tablasBlancas tablasColor sierraChica sierraGrande
  juan pedro seba ford mercedes armadores
  placard200 escritorio100 placardABC escritorioDEF
  ordenes plan |

"1- Crear los siguientes recursos"
tablasBlancas := MateriaPrima new: 'Tablas Blancas'.
tablasColor  := MateriaPrima new: 'Tablas Color'.
sierraChica  := Maquinaria new: 'Sierra Chica'.
sierraGrande := Maquinaria new: 'Sierra Grande'.
juan  := Operario new: 'Juan'.
pedro := Operario new: 'Pedro'.
seba  := Operario new: 'Seba'.
ford  := Vehiculo new: 'Ford-MTE845'.
mercedes  := Vehiculo new: 'Mercedes-MPA123'.
armadores := GrupoRecurso new: 'Armadores'.
armadores agregarRecurso: juan.
armadores agregarRecurso: pedro.
armadores agregarRecurso: seba.

"2- Definir los productos"
placard200     := Producto new: 'Placard Estandar 200'.
escritorio100 := Producto new: 'Escritorio Estandar 100'.
placardABC    := Producto new: 'Placard ABC'.
escritorioDEF := Producto new: 'Escritorio DEF'.

"3- Configurar las actividades necesarias para la fabricación de cada producto"
placard200
 agregarActividad: (Actividad new: 'Preparación' recurso: tablasBlancas duracionHoras: 2);
 agregarActividad: (Actividad new: 'Corte' recurso: sierraChica duracionHoras: 5);
 agregarActividad: (Actividad new: 'Armado' recurso: armadores duracionHoras: 6);
 agregarActividad: (Actividad new: 'Envío' recurso: mercedes duracionHoras: 2).

escritorio100
 agregarActividad: (Actividad new: 'Preparación' recurso: tablasBlancas duracionHoras: 2);
 agregarActividad: (Actividad new: 'Corte' recurso: sierraGrande duracionHoras: 5);
 agregarActividad: (Actividad new: 'Armado' recurso: armadores duracionHoras: 5);
 agregarActividad: (Actividad new: 'Envío' recurso: mercedes duracionHoras: 2).

placardABC
 agregarActividad: (Actividad new: 'Preparación' recurso: tablasColor duracionHoras: 3);
```

```
 agregarActividad: (Actividad new: 'Corte' recurso: sierraChica duracionHoras: 4);
 agregarActividad: (Actividad new: 'Armado' recurso: armadores duracionHoras: 6);
 agregarActividad: (Actividad new: 'Envío' recurso: ford duracionHoras: 2).

escritorioDEF
 agregarActividad: (Actividad new: 'Preparación' recurso: tablasColor duracionHoras: 2);
 agregarActividad: (Actividad new: 'Corte' recurso: sierraGrande duracionHoras: 3);
 agregarActividad: (Actividad new: 'Armado' recurso: armadores duracionHoras: 5);
 agregarActividad: (Actividad new: 'Envío' recurso: ford duracionHoras: 2).

"4- Crear los órdenes de producción"
ordenes := OrderedCollection new.
ordenes add: (OrdenDeProduccion new: placard200    codigo: $A).
ordenes add: (OrdenDeProduccion new: escritorio100 codigo: $B).
ordenes add: (OrdenDeProduccion new: placardABC    codigo: $C).
ordenes add: (OrdenDeProduccion new: escritorioDEF codigo: $D).

"5 y 6- Planificar e imprimir"
plan := PlanDeProduccion new.
plan planificar: ordenes.
plan imprimir.
```