Práctica 02

Modelado y Programación

Objetivo: *El objetivo de esta práctica es implementar adecuadamente los patrones State, Template e Iterator en la resolución del problema propuesto. La solución propuesta debe evitar caer en problemas de rigidez, fragilidad, inmovilidad, y viscosidad.*

1. Menciona los principios de diseño esenciales del patrón State, Template e Iterator. Menciona una desventaja de cada patrón.

2. Se te ha encargado implementar el siguiente problema:

➔ El restaurante McBurguesas te ha pedido programar un robot que servirá como mesero y cocinero. Si el prototipo cumple con los requerimientos y las pruebas, los producirán en masa para toda su franquicia.
- El robot debe estar encendido todo el tiempo esperando a que un cliente lo active, pero si no está trabajando puede estar suspendido.
- Al activarse, el robot empieza a caminar hasta llegar a la mesa del cliente.
- Una vez que se encuentra frente al cliente debe leerle todo el menú del restaurante.
- El cliente elige lo que quiere comer y el robot se pone a cocinar el platillo.
- Al terminar de cocinar, entrega la comida y se vuelve a suspender hasta que otro cliente lo activa.
- Si el robot está **suspendido** no puede hacer otra cosa más que activarse. Si el robot está **caminando** puede **suspenderse** de nuevo, pero no puede **cocinar**, pues sería peligroso para los clientes. Si el robot está **atendiendo** a un cliente no puede **suspenderse** ni ponerse a **caminar**, solamente empezará a **cocinar** hasta tener la orden del cliente. Si está **cocinando**, ya no podrá tomar otra orden hasta terminar el platillo y no se pondrá a **camina**r mientras **cocina**.

➔ El menú de McBurguesas se compone de la siguiente manera:
- Todos los platillos que están en el menú comparten la misma información: un id, el nombre del platillo, su descripción, el precio, un booleano que indica si tiene queso y un booleano que indica si es vegetariano.
- Existe un menú general con hamburguesas que siempre pueden servirse en cualquier día del año. Este menú es fijo y siempre contendrá el mismo número de hamburguesas hasta que McBurguesas deje de existir. Las hamburguesas de este menú se guardan en un arreglo.
- Existe un menú que cambia cada día. Dependiendo del gusto del gerente, este menú puede cambiar cada día y tener más o menos platillos de un día para el otro. Las hamburguesas de este menú se guardan en un ArrayList.

- Existe un menú especial con hamburguesas de lujo. Estas hamburguesas son más caras. Las hamburguesas de este menú se guardan en un Hashtable.
- Cada menú debe tener al menos 3 elementos de tu elección.
- El robot podrá acceder a todos los elementos de los menús de manera general. El robot debe tomar los menús de alguna estructura de datos ajena. Esto significa que el robot NO debe guardar los menús dentro de si mismo. Sólo podrá leerlos de alguna entidad ajena a su clase Robot. Además, debe poder mostrarlos de corrido sin problemas.

➔ El robot muestra en pantalla el proceso de cocción para que el comensal no se impaciente esperando. Para este prototipo se asume que todas las hamburguesas tienen una preparación análoga:

- poner pan
- poner mayonesa
- poner mostaza
- preparar carne
- poner carne
- poner queso
- poner vegetales
- poner catsup
- poner pan

No todas las hamburguesas llevan queso y no todas las hamburguesas preparan la carne igual ya que algunas son vegetarianas.

Agrega una clase principal con un main para probar el programa. En él deben crear un objeto de la clase Robot y un menú para el usuario. El menú debe permitir interacción con el robot. Si alguna acción no puede realizarse, se debe imprimir un mensaje para indicarle al usuario (por ejemplo, si el robot está suspendido y se le pide cocinar, la pantalla debe mostrar un mensaje que diga “no puedo cocinar mientras esté suspendido”).
El programa debe saludar al usuario, mostrarle el estado actual del robot y las acciones disponibles del mismo. Si el usuario ingresa una opción, la pantalla debe imprimir la respuesta del robot (por ejemplo, el robot inicia suspendido, el usuario elige la opción de “encender” y se debe mostrar en pantalla “robot activado”). Cuando el robot lea el menú, el usuario puede elegir un solo platillo ingresando su id. Después se debe mostrar en pantalla el proceso de cocción. Al terminar se debe indicar que el platillo se entrega y se regresa al menú inicial donde el robot está suspendido. Por cada pedido sólo se puede pedir una hamburguesa.

Notas:
Este menú servirá para probar la funcionalidad de su sistema. Es posible que el flujo sea automático (es decir, que sólo haya una secuencia de pasos para que el robot pruebe todas sus funcionalidades sin detenerse), y el menú servirá para ver este flujo con pausas en cada cambio de estado. Entonces, con cada acción deberá imprimir el menú nuevamente, el estado del robot y todas las acciones disponibles.

Para esta práctica no hace falta simular que pasan los días. Tampoco es necesario que cambien los menús en ejecución. La forma en que se describe cada menú es una excusa para usar las estructuras de datos señaladas.

Queremos probar que las transiciones sean válidas o rechazadas por el mismo sistema. Por eso necesitamos que el usuario pueda interactuar para probar todas las opciones posibles y el robot diga "no puedo hacer esto" o "pasa al siguiente estado". Así que es preferible que tenga la interacción del menú durante la ejecución del programa.

Realiza los diagramas de clase y de estados, y agregalos como imagen en la carpeta de entrega.

Sigan los lineamientos para la entrega de prácticas. El documento estará en el classroom.

Evaluación

20% Implementación de cada patrón (en total 60%)
10% Parte teórica
10% Funcionamiento correcto
20% Diagramas (se califica que concuerden con la implementación y sean correctos)
NOTA: Si el código no está comentado en formato Javadoc, como se indica en los lineamientos, se descontarán 10 puntos de 100.