Ejercicio Comercio Informáticos

Un comercio dedicado al rubro informático requiere visualizar estadísticas sobre el historial de sus operaciones de venta de los productos y/o servicios que ofrece. El lugar se dedica tanto a vender insumos informáticos como a ofrecer servicios pre y post venta.

El comercio desea saber el monto total facturado y la cantidad de servicios de reparación simple, son aquellos cuyo nivel de dificultad es menor a 2.

El comercio nos indica que todos los productos poseen:

* un nombre
* un precio de lista.
* un método que nos indica su precio de venta que depende de cada tipo de producto o servicio.

Estos productos están clasificados en los siguientes tipos:

* Insumos se conoce:
  + un nombre.
  + un precio de lista.
  + un tipo (periférico, hardware o limpieza).
  + un porcentaje de ganancia.
* Servicios, podemos encontrar dos tipos:
  + Servicio de armado:
    - un nombre.
    - un precio de lista.
    - cantidad de horas que ocupó el servicio.
    - Para calcular su precio, se multiplican la cantidad de horas del servicio por el precio de lista del valor de la hora, el cual es de $250.
  + Servicio de reparación:
    - un nombre.
    - un precio de lista.
    - cantidad de horas que ocupó el servicio
    - una dificultad (entero entre 1 y 5).
    - Para calcular su precio, se multiplican la cantidad de horas del servicio por el precio de lista del valor de la hora, el cual es de $180.
    - Si el nivel de dificultad es mayor a 3, se incrementa el precio en un 25%.

Todos los métodos de creación y seteo de atributos de los Productos deben ser protegidos. Solo se podrá interactuar con ellos a través de los métodos públicos de Comercio.

El comercio cuenta con un límite de 1000 insumos o servicios que puede ofrecer como máximo y por ahora solo se podrán agregar más productos a los existentes mediante los siguientes métodos públicos crearInsumo, crearServicioDeArmado y crearServicioDeReparacion.

Estos métodos crearan una instancia de cada Producto y lo agregarán a los productos disponibles para la venta, siempre y cuando no exista ya un Producto agregado con el mismo nombre, en ese caso se debe lanzar una excepción ProductoYaIngresadoException que indique el producto ya está ingresado.

Para ello hay que implementar el método equals por el nombre del Producto.

También debe existir un método buscarProductoPorNombre que se le pasa el nombre del producto y devuelve la instancia que se encuentra entre los productos disponibles. En caso de que no se encuentre debe retornar una excepción ProductoNoEncotradoException que indique que el producto no se ha encontrado.

Además un método modificarPrecioListaProducto que debe recibir el nombre del Producto y el nuevo precio de lista. Esté método debe buscar el producto en los disponibles y modificar su precio de lista, siguiendo las reglas de cada insumo o servicio.

El comercio guarda un historial de las operaciones realizadas que deben ser inmutables, con un límite de 10.000, que contiene tanto los insumos vendidos como los servicios prestados.

Para lograr la inmutabilidad **el Producto debe facilitar un método abstracto** clonar que debe ser implementado por todos los insumos y servicios.

Clonar un objeto es obtener otro exactamente igual pero con otra referencia.

Todo los Productos son facturables, es decir, deben ser capaces de devolver el monto de facturación, cuyas reglas son:

* Para los insumos,
  + se suma su porcentaje de ganancia al precio de lista.
  + Sobre ese valor se suma el IVA.
* Para los servicios,
  + se suma la mitad del IVA sobre su precio.
* En todos los casos,
  + el IVA es del 21%.

El comercio debe contener un método addOperacion que luego de clonar el **Producto lo agregue al historial de operaciones**. En caso que el producto que se quiera agregar a la operación sea nulo, se debe lanzar una excepción ProductoNoExistenteException que indique un mensaje por el cual no se puede agregar la operación.

También el comercio debe tener un par de métodos para ver el contenido de los productos disponibles para la venta y el historial de operaciones.

Para sumar un porcentaje sobre un precio, contás con el método estático sumarPorcentaje de la clase Matematica, al cual se le envía un valor monetario y un porcentaje y devuelve un valor double con el porcentaje sumado al precio.

Como ayuda aquí está esta clase y su método:

**public class** Matematica {

**public static** Double sumarPorcentaje(Double valor, Double porcentaje) { Double resultado = 0.0;

Double delta = (valor.doubleValue() \* porcentaje.doubleValue() / 100); resultado = valor + delta;

**return** resultado;

}

}

Basado en el enunciado descripto, se pide realizar:

1. El diagrama de clases que lo modelice, con sus relaciones, atributos y métodos. (**4 puntos**)
   1. Este diagrama lo pueden presentar en el formato de imagen.
   2. Para ello en el proyecto java crear un package doc y adjuntarlo ahí.
2. Implementar todas las clases en un paquete domain o dominio (**4 puntos**)
   1. Crear las clases, enumerados e interfaces que consideren necesarias para lograr resolver el problema.
3. Implementar una clase ComercioTest con el método main en el paquete test. (**2 puntos**)
   1. Se debe crear al menos un producto de cada tipo, agregarlo al inventario de venta y simular la operación de venta de cada uno de ellos.
   2. También se debe verificar el correcto lanzamiento de las excepciones indicadas, para los tres casos.
   3. Se debe modificar el precio de al menos uno de los productos para la venta y que esta operación no afecte el producto en el historial.
   4. Se debe mostrar al final de la ejecución el listado de los productos disponibles.
   5. Se debe mostrar al final de la ejecución el listado de los producto en el historial.
   6. Se debe mostrar la final de la ejecución el monto total facturado.
   7. Se debe mostrar la final de la ejecución la cantidad de servicios simples.
4. Para la entrega
   1. Crear un proyecto java en Eclipse (preferentemente), con el siguiente formato de nombre
      1. dv-pav-comercio-informatico-[apellido]-[nombre]
      2. donde:
         * [apellido] será reemplazado por cada uno de sus apellidos, en caso de tener más de uno, colocar solo el primero.
         * [nombre] será reemplazado por cada uno de sus nombres, en caso de tener más

de uno, colocar solo el primero.

* + - * IMPORTANTE: tanto el apellido y el nombre debe estar en **minúscula**.
    1. Entregar el proyecto java de eclipse en formato zip, ejemplo: dv-pav-comercio-informatico-montero-francisco.zip
  1. Enviar el archivo zip a la casilla de correo [francisco.montero@davinci.edu.ar](mailto:francisco.montero@davinci.edu.ar)