Programación Básica 1 - Parcial 2 C01 Q2 TM 2023

Gestión de tienda

Se nos solicitó desarrollar un producto software que permita gestionar ciertas operaciones realizables en una tienda que vende productos o servicios.
Cada tienda puede disponer de productos o servicios para generar ventas, los cuales son llamados “Vendible”. Cada instancia de Vendible está categorizada como producto o servicio.
En esta entrega, nos centraremos en la generación de operaciones para ventas.

Clase de prueba GestionDeTienda

Será la clase que contenga al método main y deberemos desarrollar lo siguiente para garantizar el correcto funcionamiento:

- **private static MenuPrincipal obtenerOpcionDeEnumParaMenuPrincipal():** Obtener la opción seleccionada por el usuario del menú principal. Se debe validar que la opción ingresada por el usuario, se encuentre entre las disponibles.
- **private static void buscarVendiblesCuyoCodigoIniciaConTexto(Tienda tienda):** Solicitarle al usuario el ingreso del texto que se usará para buscar en el comienzo del código de cada vendible. Mostrar los vendibles que cumplan con la búsqueda.
- **private static void crearVentaDeProductosOServicios(Tienda tienda):** Solicitar al usuario el ingreso del cliente y el vendedor, para luego proceder a agregar los vendibles que serán parte de la venta. Se debe permitir al usuario ingresar la cantidad de vendibles que quiera, pero con un máximo de 1000. Se deberán mostrar los vendibles para que el usuario pueda elegir cual es el que desea incluir en la venta, ingresando su código. Si se quiere continuar agregando vendibles a la venta, se deberá indicar mediante el ingreso del caracter 's'. En caso de no querer continuar con el agregado de vendibles a la venta, se deberá ingresar el carácter 'n'. Con el vendedor, cliente y vendibles ingresados, se debe proceder a la creación de la venta (ver el método crearVentaDeProductosOServicios() de la Tienda). En caso de que la venta se concrete, mostrar un mensaje de éxito, caso contrario, mostrar un mensaje que indique que la venta fue cancelada.

Nombre y Apellido DNI FIrma

- **private static void mostrarMenuPrincipal():** Mostrar el menú principal de manera dinámica utilizando las opciones del enum. La presentación de las opciones debe ser una debajo de otra, comenzando con el número 1 como primera opción.
- **private static TipoDeVendible obtenerOpcionDeEnumParaTipoDeVendible():** Solicitar al usuario el ingreso de la opción requerida para el TipoDeVendible (PRODUCTO o SERVICIO) deseado y devolverla. Se debe validar que el usuario ingrese una opción válida, caso contrario se debe continuar mostrando las opciones y solicitando el ingreso.

Clase Tienda

Esta clase contiene la lógica principal de las operaciones que soportará la tienda. Completar el constructor, getter, setters y método necesarios para asegurar el correcto funcionamiento, además de los siguientes métodos:

- **public Vendible[] buscarVendiblesCuyoCodigoIniciaConTexto(String textoABuscar):** Se debera devolver un array de vendibles cuyo código comienza con el texto indicado indicado para buscar.
- **public Vendible obtenerVendiblePorCodigo(String codigo):** Se deberá encontrar y devolver un vendible en el array de vendibles que cumpla con el código suministrado. En caso de no encontrarlo, se deberá devolver null.
- **public boolean crearVentaDeProductosOServicios(String cliente, String vendedor, Vendible[] vendiblesParaVenta):** Se deberá instanciar una venta utilizando los parámetros y luego agregar la misma al array de ventas. El método debe devolver verdadero en caso de completar la operación o falso en caso de no lograrlo por alguna razón.
- **public int obtenerCantidadDeServiciosEnVentas():** Se deberá devolver la cantidad de servicios presentes en todas las ventas. Cada venta contiene los vendibles (PRODUCTO o SERVICIO) que se incluyeron en la misma. De estos, nos interesa saber cuantos vendibles son de tipo "SERVICIO". Es importante mirar los vendibles en todas las ventas de la tienda.
- **public Vendible[] obtenerProductosConStockMaximoOrdenadosPorPrecioDescendente():** Armar y devolver un array de vendibles de tipo producto, cuyo stock sea el maximo admitido por la tienda (Revisar la constante). El array a devolver deberá estar ordenado por precio de manera descendente (ver ordenarVendiblesPorPrecioDescendente()).

- **private void ordenarVendiblesPorPrecioDescendente(Vendible[] vendibles):** Ordenar el array de vendibles suministrado por precio del vendible, de manera descendente.
- **private void inicializarTienda():** Se deberán generar los vendibles que estarán disponibles para que la tienda pueda operar. Es necesario asignar cada posición del array de vendibles con una instancia de vendible. En las posiciones pares, se deberá colocar un vendible de tipo PRODUCTO, mientras que en las posiciones impares, se deberá colocar un vendible de tipo SERVICIO. El código de los vendibles debe ser la letra 'P' seguido de un número para los productos. Para los servicios, el código del vendible debe ser una letra 'S' seguido de un número. El precio de los productos se deberá generar considerando que no sean todos iguales (nos sirve para ordenar). El precio de los servicios puede ser fijo. Los servicios no tienen cantidad maxima, en este atributo se deberá ingresar un cero. Ejemplos: códigos de productos: "P0", "P1", etc. Ejemplos: códigos de servicios: "S0", "S1", etc.

Clase Vendible

Representa la abstracción de algo posible de ser vendido en la tienda. Para este producto software podrá ser de tipo PRODUCTO o SERVICIO.
Completar el constructor, getters, setters y otros métodos necesarios.

Lineamientos

- No es posible modificar la firma de los métodos existentes, pero es posible agregar todos los métodos que se consideren necesarios.
- Se pueden aplicar las mejoras que considere necesarias, siempre cumpliendo el punto anterior.
- Se deberán completar los métodos y código faltante para que el sistema funcione correctamente.
- Deberá ajustar el nombre del proyecto suministrado indicando sus apellidos y nombres en donde dice “ApellidosNombres”: PBI2023Q2C01TM-Parcial2-**ApellidosNombres.**
- La versión de java del proyecto deberá ser ajustada a la del laboratorio.

Condiciones de aprobación

Se considerará desaprobado todo proyecto que:

Nombre y Apellido DNI FIrma

a. no compile, o,
b. no se identifiquen los atributos necesarios para resolver lo solicitado, o,
c. los tipos de datos elegidos para los atributos no sean los adecuados, o,
d. no cumpla con las condiciones mínimas descritas a continuación.

Se considerará aprobado aquel examen que cumpla con el 70% de puntaje.

Clase de prueba GestionDeTienda	private static MenuPrincipal obtenerOpcionDeEnumParaMenuPrincipal()	5%
	private static void buscarVendiblesCuyoCodigoIniciaConTexto(Tienda tienda)	5%
	private static void crearVentaDeProductosOServicios(Tienda tienda)	15%
	private static void mostrarMenuPrincipal()	10%
	private static TipoDeVendible obtenerOpcionDeEnumParaTipoDeVendible()	5%
Clase Tienda	public Vendible[] buscarVendiblesCuyoCodigoIniciaConTexto(String textoABuscar)	5%
	public Vendible obtenerVendiblePorCodigo(String codigo)	5%
	public boolean crearVentaDeProductosOServicios(String cliente, String vendedor, Vendible[] vendiblesParaVenta)	5%
	public int obtenerCantidadDeServiciosEnVentas()	20%
	public Vendible[] obtenerProductosConStockMaximoOrdenadosPorPrecioDescendente()	10%
	private void ordenarVendiblesPorPrecioPrescendente(Vendible[] vendibles)	5%
	private void inicializarTienda()	5% ,

Nombre y Apellido DNI FIrma

Clase Vendible	Constructor, getters, setters y métodos necesarios	5%
Total		100%

Nombre y Apellido DNI FIrma