Descripción de Funciones por Clase

Clase BST<E extends Comparable<E>>

Esta clase implementa un árbol binario de búsqueda genérico que proporciona una estructura de datos eficiente para almacenar elementos ordenados. Se encarga de:

- Organizar elementos de forma jerárquica según su orden natural.
- Permitir búsquedas con complejidad O(log n) en el caso promedio.
- Facilitar recorridos ordenados (ascendente y descendente) de los elementos.
- Gestionar la inserción de nuevos elementos manteniendo la propiedad de ordenación del árbol.

Clase BSTNode (interna de BST)

Es una clase auxiliar que representa cada nodo dentro del árbol binario. Se encarga de:

- Almacenar el valor del elemento.
- Mantener referencias a sus hijos (subárboles izquierdo y derecho).
- Servir como unidad estructural básica para la construcción del árbol.

Clase Producto

Modela los productos del retail con sus atributos y comportamientos. Se encarga de:

- Encapsular toda la información relevante de un producto (SKU, precios, nombre, categoría).
- Implementar la comparación entre productos basada en el SKU para su ordenación.
- Proporcionar métodos de acceso a los datos de los productos.
- Facilitar la representación textual de los productos para su visualización.

Clase BuscadorProductos

Actúa como controlador principal del sistema. Se encarga de:

- Coordinar la interacción entre el usuario y las funcionalidades del sistema.
- Gestionar la carga de productos desde archivos CSV.
- Implementar la lógica de búsqueda y listado de productos.
- Proporcionar una interfaz de usuario a través de un menú de consola.
- Organizar la presentación de los resultados de manera amigable.

Clase CSVHandler

Especializada en el procesamiento de archivos CSV. Se encarga de:

- Leer y parsear archivos CSV con datos de productos.
- Identificar y mapear columnas del CSV a atributos de la clase Producto.
- Manejar errores de formato y validación de datos.
- Convertir los datos brutos del CSV en objetos Producto utilizables por el sistema.
- Resolver problemas relacionados con rutas de archivos y formatos.

Descripción de los atributos de cada clase

Clase BST<E extends Comparable<E>>

- **root**: Referencia al nodo raíz del árbol binario de búsqueda.
- **size**: Contador que mantiene el número de elementos en el árbol.

Clase BSTNode (clase interna de BST)

- **data**: Almacena el valor del nodo, de tipo genérico E.
- **left**: Referencia al hijo izquierdo del nodo.
- **right**: Referencia al hijo derecho del nodo.

Clase Producto

- **sku**: Identificador único del producto (String).
- **priceRetail**: Precio original del producto (double).
- **priceCurrent**: Precio actual/en oferta del producto (double).
- **productName**: Nombre del producto (String).
- **category**: Categoría a la que pertenece el producto (String).

Clase BuscadorProductos

- **productosTree**: Árbol binario de búsqueda que almacena los productos.

Clase CSVHandler

- No contiene atributos propios, solo métodos estáticos.

2. Descripción de los métodos de cada clase

Clase BST<E extends Comparable<E>>

- **BST()**: Constructor que inicializa un árbol vacío.
- **insert(E element)**: Inserta un nuevo elemento en el árbol.
- **insertRecursive(BSTNode current, E element)**: Método auxiliar para realizar la inserción recursivamente.
- **search(E element)**: Busca un elemento en el árbol.
- **searchRecursive(BSTNode current, E element)**: Método auxiliar para realizar la búsqueda recursivamente.
- **inOrderTraversal(Consumer<E> action)**: Recorre el árbol en orden (izquierda-raíz-derecha).
- **inOrderTraversal(BSTNode node, Consumer<E> action)**: Método auxiliar para el recorrido en orden.
- **reverseInOrderTraversal(Consumer<E> action)**: Recorre el árbol en orden inverso (derecha-raíz-izquierda).
- **reverseInOrderTraversal(BSTNode node, Consumer<E> action)**: Método auxiliar para el recorrido en orden inverso.
- **size()**: Devuelve el número de elementos en el árbol.
- **isEmpty()**: Verifica si el árbol está vacío.
- **clear()**: Elimina todos los elementos del árbol.

Clase Producto

- **Producto(String sku, double priceRetail, double priceCurrent, String productName, String category)**: Constructor completo.
- **Producto(String sku)**: Constructor para búsquedas que solo requiere el SKU.
- **getSku()**, **getPriceRetail()**, **getPriceCurrent()**, **getProductName()**, **getCategory()**: Métodos getter.
- **compareTo(Producto other)**: Compara productos por su SKU (implementación de Comparable).
- **equals(Object obj)**: Compara si dos productos son iguales basándose en su SKU.
- **hashCode()**: Genera un código hash basado en el SKU.
- **toString()**: Devuelve una representación en texto del producto.

Clase BuscadorProductos

- **BuscadorProductos()**: Constructor que inicializa el árbol de productos.
- **cargarProductos(String filePath)**: Carga productos desde un archivo CSV.
- **buscarProductoPorSKU(String sku)**: Busca un producto por su SKU.
- **listarProductosAscendente()**: Lista productos en orden ascendente por SKU.
- **listarProductosDescendente()**: Lista productos en orden descendente por SKU.
- **main(String[] args)**: Método principal que ejecuta el programa.
- **obtenerRutaArchivo(Scanner scanner)**: Solicita al usuario la ruta del archivo CSV.
- **mostrarMenuPrincipal(Scanner scanner, BuscadorProductos buscador)**: Muestra el menú principal.
- **buscarProducto(Scanner scanner, BuscadorProductos buscador)**: Gestiona la búsqueda de un producto.
- **listarProductos(Scanner scanner, BuscadorProductos buscador, boolean ascendente)**: Gestiona el listado de productos.
- **cargarNuevoArchivo(Scanner scanner, BuscadorProductos buscador)**: Carga un nuevo archivo CSV.

Clase CSVHandler

- **cargarProductosDesdeCSV(String filePath)**: Lee un archivo CSV y convierte sus datos en una lista de productos.
- **obtenerRutaValida(String filePath)**: Valida y normaliza la ruta de un archivo.
- **limpiarCampo(String campo)**: Limpia un campo de texto eliminando comillas y espacios.
- **parseDouble(String valor, double defaultValue)**: Convierte un string a double con manejo de errores.