ESCUELA DE CIENCIAS EXACTAS E INGENIERÍA
CIENCIAS DE LA COMPUTACIÓN E INTELIGENCIA ARTIFICIAL
ASIGNATURA ESTRUCTURA DE DATOS LINEALES
Actividad # 3 Implementación Tablas Hash

Enunciado del Ejercicio: Gestión de Inventario de una Tienda de Electrónica

Una popular tienda de electrónica necesita mejorar la eficiencia de su sistema de gestión de inventario. Actualmente, cada producto se identifica mediante un código SKU (Stock Keeping Unit) único, que es una cadena alfanumérica. La tienda maneja una gran cantidad de productos y realiza operaciones frecuentes como:

1. **Agregar un nuevo producto:** Se debe registrar un nuevo producto con su SKU, nombre, descripción y cantidad disponible.
2. **Buscar un producto:** Dado un SKU, se necesita recuperar rápidamente toda la información asociada a ese producto.
3. **Actualizar la cantidad de un producto:** Para un SKU dado, se debe poder modificar la cantidad de existencias.
4. **Eliminar un producto:** Si un producto ya no se vende, se debe poder eliminarlo del inventario.

El problema actual es que el sistema existente, basado en búsquedas lineales o estructuras de datos ineficientes, se vuelve muy lento a medida que el número de productos aumenta, afectando la experiencia del cliente y la eficiencia operativa.

Objetivo:

Diseñar e implementar un sistema de gestión de inventario que utilice una **tabla hash** para almacenar y gestionar la información de los productos. La tabla hash debe permitir un acceso, inserción, actualización y eliminación de productos de forma eficiente (idealmente en tiempo promedio O(1)).

Requisitos:

1. **Estructura de Datos:** Define una estructura o clase Producto que contenga al menos los siguientes atributos:
   - sku (cadena de caracteres, clave única)
   - nombre (cadena de caracteres)
   - descripcion (cadena de caracteres)
   - cantidad (entero)
2. **Tabla Hash:** Implementa una clase TablaHashInventario que encapsule la lógica de la tabla hash. Esta clase debe incluir:
   - **Función Hash:** Elige e implementa una función hash adecuada que tome un sku (cadena) y devuelva un índice numérico para la tabla. Considera la distribución de las claves y la minimización de colisiones.
   - **Manejo de Colisiones:** Implementa una estrategia de manejo de colisiones. Se recomienda utilizar **encadenamiento separado (separate chaining)**, donde cada celda de la tabla hash apunta a una lista enlazada (o similar) de productos que han colisionado en ese índice.
   - **Operaciones:** La clase TablaHashInventario debe proporcionar los siguientes métodos:
     - agregar_producto(producto): Agrega un nuevo producto a la tabla hash. Si el SKU ya existe, se debe manejar el caso (e.g., lanzar una excepción o actualizar el producto existente).

- buscar_producto(sku): Busca y devuelve el objeto Producto asociado al SKU dado. Si no se encuentra, debe retornar None o lanzar una excepción.
- actualizar_cantidad(sku, nueva_cantidad): Actualiza la cantidad de un producto. Si el SKU no existe, debe manejar el caso.
- eliminar_producto(sku): Elimina el producto asociado al SKU de la tabla. Si el SKU no existe, debe manejar el caso.
- mostrar_inventario(): (Opcional pero recomendable) Un método para imprimir el contenido actual de la tabla hash (SKU y nombre de los productos) de manera legible, para propósitos de depuración.

3. **Interfaz de Usuario (Opcional pero recomendable):** Implementa un pequeño programa principal que permita al usuario interactuar con el sistema de inventario a través de un menú de opciones (agregar, buscar, actualizar, eliminar, salir).

Consideraciones Adicionales:

- **Tamaño Inicial de la Tabla:** ¿Cómo elegir el tamaño inicial de la tabla hash? ¿Debería ser un número primo? Justifica tu decisión.
- **Redimensionamiento (Opcional Avanzado):** Si la tabla se llena demasiado y las colisiones se vuelven excesivas, ¿cómo manejarías el redimensionamiento de la tabla hash para mantener un rendimiento óptimo? (No es un requisito obligatorio, pero una buena consideración para soluciones más robustas).
- **Pruebas:** Diseña algunos casos de prueba para verificar que todas las operaciones funcionan correctamente, incluyendo casos límite (SKUs no existentes, añadir múltiples productos con el mismo SKU, etc.).

Entrega:

Se debe entregar el código fuente de la implementación en el lenguaje de programación de tu elección (Python, Java, C++, etc.), incluyendo las clases Producto y TablaHashInventario, y el programa principal si se implementa. También se debe incluir una breve descripción de las decisiones de diseño (función hash, manejo de colisiones) y una justificación de la complejidad de las operaciones principales.