UT3 – TA1

Ejercicio 1

1. ¿Cuál es el costo de memoria en cada caso? (usando arrays o listas encadenadas)
   - Arrays: Depende del tamaño que se cree el array. Solo se necesita un espacio de memoria para referenciar en el array y el largo.
   - Listas Encadenadas: Depende de la cantidad de productos que se creen en la lista. Por cada nodo se una referencia a el elemento y otra al siguiente.

2. ¿Cuáles son las consideraciones que tu Equipo haría referentes a la cantidad de productos que soporta cada tipo de estructura?
   - Arrays: la cantidad de productos es fijo. Y tarde o temprano va a hacer overflow y vas a necesitar redimensionar el array.
   - Lista Encadenada: es más flexible a la hora de agregar productos ya que no tiene un límite de nodos

3. ¿Cuáles son las consideraciones que tu Equipo haría referentes a la eficiencia de las operaciones solicitadas, dependiendo de cada tipo de implementación?
   - Arrays: Largo fijo, costo para redimensionar, pero eficiente a la hora de buscar un producto. Costoso en espacio de memoria
     - Agregar Producto: En el peor de los casos hay que redimensionar el array para agregar el producto.
     - Agregar Stock: Se necesita recorrer el array para encontrar el producto y agregar el stock.
     - Eliminar Stock: Se necesita recorrer el array para encontrar el producto y eliminar el stock.
     - Listar: Se necesita recorrer todo el array para ordenarlos y listarlos.
   - Lista Encadenada: difícil de recorrer, pero fácil de agregar y/o eliminar productos. Menos costos que los arrays en el espacio de memoria.
     - Agregar Producto: Se crea otro nodo y se enlaza a la lista.
     - Agregar Stock: Se necesita recorrer la lista para encontrar el producto y agregar el stock.
     - Eliminar Stock: Se necesita recorrer la lista para encontrar el producto y eliminar el stock.
     - Listar: Se necesita recorrer toda la lista para ordenarlos y listarlos. Mejora Podría ser agregarlos ordenados con un método de ordenamiento al método agregarProducto