EJERCICIOS DE APAREO

1. Apareo Básico de Números Enteros

Problema: Se tienen dos listas de números de documentos de identidad recolectados en dos mesas electorales distintas. Ambos vectores, mesaA y mesaB, están ordenados de forma ascendente.

- **Tarea**: Realizar un procedimiento que reciba ambos vectores y genere un tercer vector padronFinal que contenga todos los documentos de ambas mesas ordenados ascendentemente.
- **Concepto clave**: Implementar el ciclo while (i < n && j < m) para comparar elementos y los ciclos posteriores para vaciar los elementos restantes del vector que no se haya agotado.

2. Consolidación de Inventario (Apareo con Registros)

Problema: Una empresa tiene dos depósitos. Cada uno cuenta con un vector de productos cargado con la estructura struct Producto { int codigo; int stock; }. Ambos vectores están ordenados por código.

- **Tarea**: Generar un tercer vector que unifique el inventario. Si un código de producto aparece en ambos depósitos, el registro en el tercer vector debe ser la suma de ambos stocks (adaptando la lógica de comparación).
- **Concepto clave**: Apareo de vectores de registros utilizando el campo codigo como clave de comparación.

3. Fusión de Listas de Alumnos (Caso de Examen)

Problema: Utilizando la estructura struct Alumno { int legajo; string apellido; string nombres; int documento; } se reciben dos vectores de alumnos de distintas sedes, ambos ordenados por legajo de forma ascendente.

- **Tarea**: Diseñar un procedimiento de apareo que cargue un tercer vector manteniendo el orden por legajo.
- **Concepto clave**: Respetar la firma de la función ***void apareo(Alumno vecA[], int n, Alumno vecB[], int m, Alumno vecC[], int &k)*** y el pasaje por referencia del tamaño del vector resultante.

4. Apareo sin Duplicados (Validación Logística)

Problema: Dos sistemas de logística generan listas de códigos de rastreo diarios. Debido a errores de carga, un mismo código podría estar en ambos sistemas. Los vectores están ordenados por código.

- **Tarea**: Realizar el apareo de ambos vectores en un tercero, pero con la restricción de que no debe haber valores repetidos en el vector final.
- **Concepto clave:** Incorporar validaciones dentro del algoritmo de apareo original para omitir la carga si el elemento a insertar es igual al último ya insertado en el vector resultante.