ALGORITMOS DE CLASIFICACIÓN DE OBJETOS CONTENIDOS EN ARRAYLISTS

1. Comparación de métodos de ordenamiento en Java

- Investigar cómo funcionan Collections.sort(), List.sort() y Stream.sorted().
- Comparar sus ventajas, limitaciones y casos de uso.
- Analizar qué método es más eficiente según el tamaño de una lista y el tipo de datos.
- Agregar un ejemplo corto.

2. Uso de Comparable vs Comparator

- Explicar las diferencias conceptuales y prácticas.
- Investigar cuándo conviene implementar Comparable y cuándo usar Comparator.
- Ejemplificar con un caso real (ejemplo: lista de productos, empleados o estudiantes).
- Agregar un ejemplo corto.

3. Implementación de algoritmos clásicos en ArrayList

- Investigar y programar ordenamientos manuales en Java (Burbuja, Selección, Inserción, QuickSort, MergeSort).
- Comparar su eficiencia usando un ArrayList.
- Mostrar cuándo conviene usar cada uno en lugar de Collections.sort().

Presentar su trabajo en un documento en Word con: portada y desarrollo. Aplicar formato: fuente (arial), tamaño (11), interlineado (1.5) y aplicar negrita a temas y subtemas.

Criterio	Excelente (90-100%)	Bueno (70-89%)	Aceptable (50-69%)	Deficiente (<50%)	Ponderación
1. Comparación de métodos de ordenamiento en Java	Explica con claridad y profundidad cómo funcionan Collections.sort(), List.sort() y Stream.sorted(). Presenta comparación detallada de ventajas, limitaciones y casos de uso. Incluye análisis de eficiencia con ejemplos y pruebas.	Explica adecuadamente los tres métodos, destacando ventajas y limitaciones. Compara eficiencia con ejemplos simples.	Explica de forma superficial los métodos, sin mucha comparación ni análisis de eficiencia.	Menciona de forma incompleta o confusa los métodos, sin comparación ni ejemplos claros.	**30%**
2. Uso de Comparable vs Comparator	Explica claramente las diferencias conceptuales y prácticas. Argumenta bien cuándo conviene usar cada uno. Incluye ejemplo real bien implementado y explicado.	Explica diferencias básicas y cuándo usar cada uno. Presenta ejemplo práctico, aunque con algunas limitaciones.	Menciona diferencias sin mucha claridad y con un ejemplo poco relacionado o incompleto.	No distingue correctamente entre Comparable y Comparator. No presenta ejemplo funcional.	**30%**
3. Implementación de algoritmos clásicos en ArrayList	Investiga y programa al menos 3-4 algoritmos clásicos (Burbuja, Selección, Inserción, QuickSort, MergeSort) de forma correcta. Compara su eficiencia con pruebas en ArrayList. Explica cuándo conviene usarlos frente a Collections.sort().	Implementa al menos 2-3 algoritmos clásicos. Muestra alguna comparación de eficiencia y ejemplos de uso.	Implementa solo 1-2 algoritmos con errores o sin pruebas de eficiencia.	No implementa algoritmos o los presenta de forma incorrecta y sin análisis.	**40%**