Segunda Parte ejercicio 25 Bag

1) Liste los métodos que debe contener una clase que implementa la interface Bag.

Métodos de Interface Bag:
1) boolean add (T key) // agrega clave e incrementa el contador.
2) int ocurrencesOf (T key) // retorna valor del contador de la clave.
3) void removeOccurrence (T key) // disminuye en 1 contador de la clave.
4) void removeAll (T key) // elimina clave-valor de bag
5) int size() // retorna la suma total de todo los contadores de bag.
6) Iterator <T> iterator()
// necesario para iterar sobre elementos (ya está en AbstractCollection).

Además, por extender Collection<T>, también podrían implementar los métodos opcionales (pero AbstractCollection ya provee comportamientos por defecto)
por ejemplo: contains, isEmpty, remove, etc.

2) Explique cómo implementaría un Bag<T> usando composición con un Map<K,V>? ¿De qué tipo tendrían que ser las claves y valores del Map?

Usaría un Map < T, Integer >
- Clave (T) → el objeto almacenado.
- Valor (Integer) → cuántas veces fue agregado a la bolsa (su *cardinalidad*).

3) Implemente la interfaz Bag, utilizando AbstractCollection como superclase, y componga con un Map.
Para simplificar la implementación utilice la clase BagImpl que se encuentra en este link. ✓

4) Discuta con un ayudante:
a) ¿Cuáles son los beneficios de utilizar AbstractCollection como superclase para implementar el Bag?

- Proporciona implementaciones por defecto de varios métodos de Collection, como contains, isEmpty, remove, etc.
- Solo obliga a implementar size() e iterator(), simplificando enormemente la clase.
- Ahorra código y evita errores.

b) ¿Qué ventajas tiene componer con un Map para implementar el Bag?

- Permite almacenar directamente la cardinalidad de los elementos.
- Optimiza el método size(), ya que se evita recorrer todo el Bag repetidamente.
- Facilita la implementación de occurrencesOf, removeOccurrence y removeAll.

c) En lugar de componer con un Map,
¿es posible extenderlo para poder implementar el Bag?
¿Qué diferencias tendría esa solución con respecto a la planteada en este ejercicio?

Sí, es posible, pero no es recomendable**.**
La composición implica que la clase Bag tenga un atributo interno de tipo Map, por ejemplo:
private Map<T, Integer> bag;
Esta opción es la recomendada, porque permite controlar qué métodos se exponen y ocultar aquellos propios del map que no corresponden a la lógica de Bag.
En cambio, si se utilizara extensión, por ejemplo:
public class Bag extends HashMap<T, Integer>
Bag se comportaría como un Map completo, heredando todos sus métodos públicos, incluso aquellos que no son apropiados para una bolsa (Bag).
Esto podría romper la lógica del objeto y ofrecer operaciones que no deberían estar disponibles.
Por esa razón, **se prefiere la composición antes que la extensión**, ya que permite encapsular mejor el comportamiento y definir exactamente qué funcionalidades tendrá el Bag.

d) ¿Para qué cree que podría ser útil un objeto Bag?

- Para contar elementos repetidos (frecuencia).
- En análisis estadístico (histogramas).
- Al analizar texto (contar palabras o caracteres).
- Inventarios (cantidad de cada producto).
- Sistemas de recomendación (preferencias repetidas).
- Juegos (cantidad de recursos/objetos acumulables).