

Nearpod - Diccionarios

app.nearpod.com/presentation?pin=J9GB6

# Diccionario

Un **diccionario** es una estructura de datos que almacena información en **pares clave-valor**.

En Java, su equivalente directo es la interfaz `Map<K,V>`, donde:

- `K` = clave (Key)
- `V` = valor (Value)

Cada clave está asociada a un único valor, y **no pueden existir claves duplicadas**.

Ejemplo conceptual:

```
"ID" → 1204
"Nombre" → "María"
"Edad" → 22
```

Esto lo vuelve ideal para búsquedas rápidas, asociaciones, tablas, índices y catálogos.

3 de 14Abrir navegador ▲

Nearpod - Diccionarios

app.nearpod.com/presentation?pin=J9GB6

# Características

- ✓ Almacena pares *clave-valor*
- ✓ Las claves **no se pueden repetir**
- ✓ Los valores **pueden repetirse**
- ✓ La búsqueda se realiza por clave
- ✓ No forma parte de la jerarquía `Collection`, sino paralela
- ✓ Las implementaciones tienen distintos comportamientos internos

4 de 14Abrir navegador ▲

Nearpod - Diccionarios

app.nearpod.com/presentation?pin=J9GB6

# Componentes Principales

3.1 Clave (Key)

- No se puede duplicar
- Debe implementar correctamente `equals()` y `hashCode()` (para HashMap)
- Puede ser objeto, entero, string, etc.
- No es recomendable que sea mutable

3.2 Valor (Value)

- Puede repetirse
- Puede ser de cualquier tipo

3.3 Entrada (Entry)

Cada par clave-valor se encapsula en un `Map.Entry<K,V>`

Es usado para iterar:

```
for (Map.Entry<String, Integer> e : mapa.entrySet()) {  
    System.out.println(e.getKey() + " => " + e.getValue());  
}
```

5 de 14Abrir navegador ▲

Nearpod - Diccionarios

app.nearpod.com/presentation?pin=J9GB6

# HashMap

✓ La más rápida en promedio

✓ Búsqueda  $O(1)$

✓ No garantiza orden

✓ Permite **una clave null**

✓ Permite valores null

✓ No es sincronizada

Uso típico:

- índices
- cache
- acceso rápido

6 de 14Abrir navegador ▲

Nearpod - Diccionarios

app.nearpod.com/presentation?pin=J9GB6

# TreeMap

- ✓ Ordena automáticamente → orden natural o Comparador
- ✓ Implementa **NavigableMap**
- ✓ No permite claves null
- ✓ Operaciones  $O(\log n)$
- ✓ Internamente usa un árbol rojo-negro

Ideal para:

- diccionarios ordenados
- autocompletar
- estructuras jerárquicas

7 de 14

Abrir navegador ▲

Nearpod - Diccionarios

app.nearpod.com/presentation?pin=J9GB6

# LinkedHashMap

- ✓ Mantiene orden de inserción
- ✓ Similar a HashMap pero con LinkedList interna
- ✓ Útil para históricos o registros ordenados
- ✓ Se puede configurar para orden por "acceso" (LRU Cache)

8 de 14

Abrir navegador ▲

Nearpod - Diccionarios

app.nearpod.com/presentation?pin=J9GB6

# HashTable

- ✓ Sincronizada
- ✓ NO permite null (ni clave ni valor)
- ✓ Obsoleta para la mayoría de los usos modernos
- ✓ Su equivalente moderno es `ConcurrentHashMap`

9 de 14Abrir navegador ▲

Nearpod - Diccionarios

app.nearpod.com/presentation?pin=J9GB6

# Operaciones Diccionarios

Operación	Método en Map	Descripción
Insertar	<code>put(k,v)</code>	Agrega o reemplaza
Insertar sin reemplazar	<code>putIfAbsent(k,v)</code>	No reemplaza si ya existe
Obtener valor	<code>get(k)</code>	Devuelve valor o null
Eliminar	<code>remove(k)</code>	Elimina por clave
Tamaño	<code>size()</code>	Número de pares
Verificar clave	<code>containsKey(k)</code>	True/False
Verificar valor	<code>containsValue(v)</code>	True/False
Limpiar	<code>clear()</code>	Vacía el map
Iterar	<code>entrySet()</code>	Iteración completa

10 de 14Abrir navegador ▲



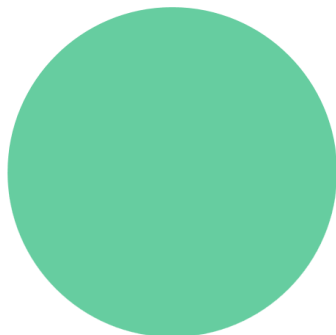
## Completar Frases Codigo Dicionario



## Diccionarios



ACERTASTE 7 DE 7



100 CORRECTA(S)  
0 INCORRECTA(S)  
0 SIN RESPUESTA

12 de 14

Abrir navegador ▲



### Diccionarios en Java: Memoria de Map Memory

Tavares

Comenzar



¡HAS GANADO!  
¡ENHORABUENA!



TOTAL PUNTOS **100.000**

🔄 Reintentar