Video18 (1)

Conceptos Principales

1. a
2. b
3. c

Notas

📌 Diccionarios / Tablas de Hash

✅ Concepto general

- Un **diccionario** (o hash table) es una estructura que **relaciona llaves con valores**.
- Internamente es un **arreglo de estructuras** (`struct`) donde se guarda tanto la **llave** como el **valor**.
- Se usa una **función de hash** para convertir una llave en una **posición del arreglo**.

🧩 Funcionamiento básico

1. Se recibe una **llave**.
2. Se le aplica la **función hash** → devuelve un índice.
3. En ese índice del arreglo se guarda el par `{llave, valor}`.
4. Para buscar: se repite el cálculo y se verifica si la llave está en esa posición.

⚠️ Colisiones

Ocurren cuando **dos llaves diferentes producen el mismo índice**.

Formas de resolverlas:

1. **Encadenamiento (listas enlazadas)**
   - Cada posición del arreglo apunta a una lista simple.
   - Si varias llaves caen en la misma posición, se encadenan en esa lista.
   - Desventaja: si hay muchas colisiones, la búsqueda se vuelve lenta (búsqueda lineal).
2. **Encadenamiento con cabeza**
   - La posición almacena directamente el **primer nodo** de la lista.
   - A veces permite acceder más rápido si justo coincide la llave.
3. **Direcciones abiertas**
   - Si una posición está ocupada, se busca otra dentro del mismo arreglo.
   - Estrategias:
     - **Lineal**: probar la siguiente, y la siguiente, hasta encontrar un espacio.
     - **Cuadrática**: saltar 1, luego 2^2, luego 4^2, etc.
     - **Doble hash**: usar una **segunda función de hash** si la primera posición ya está ocupada.

📊 Factor de carga y redimensionamiento

- **Factor de carga** = (número de elementos) ÷ (tamaño del arreglo).
- Ejemplo: 6 elementos en arreglo de 12 → **50%**.
- Si el factor de carga pasa ~66%–75%, conviene **redimensionar**:
  - Crear un arreglo más grande.
  - Reinsertar todos los elementos aplicando la función de hash otra vez (mod nuevo tamaño).
- Mantener factor de carga bajo = menos colisiones, mejor rendimiento.

🩺 Puntos clave

- La **función hash** debe ser eficiente y distribuir bien los valores.
- Inserción y búsqueda siguen el **mismo proceso**: calcular hash y manejar colisiones.
- Python, por ejemplo, redimensiona sus tablas cuando pasan cierto límite (~66%).

✅ **Idea central**:

Un diccionario es un **arreglo indexado por llaves en vez de índices numéricos**, y funciona bien gracias a las **funciones hash** y a estrategias para manejar **colisiones** y **redimensionamiento**.