Video33 (1)

Conceptos Principales

1. a
2. b
3. c

Notas **Resumen general**

El video explica cómo manejar **archivos binarios secuenciales**, comparándolos con los archivos de texto. Se introducen **archivos híbridos** (texto con información binaria) y luego se pasa a **archivos binarios puros**, mostrando cómo mejoran la eficiencia y reducen el espacio ocupado en disco.

De texto a binario

1. **Archivo híbrido (texto + binario)**

- Soluciona el problema de **pérdida de información** de los archivos de tamaño fijo.
- Cada campo guarda **un byte adicional** que indica su **longitud**.

  Ejemplo:

  ```
  [7]Francis[8]Bernard[18]San José, Costa Rica
  ```

- Así, el programa **sabe cuántos bytes leer** para cada campo.

Ventajas:

- No se recorta información.
- Se puede abrir en un editor de texto (aunque no conviene editarlo manualmente).

Desventajas:

- Cada campo implica **más lecturas al disco** (primero leo el tamaño, luego el valor).
- Las lecturas múltiples hacen que el archivo sea **más lento** de procesar.

Archivos binarios puros

- En estos archivos **no hay texto ni caracteres visibles**: solo **bytes**.
- La extensión del archivo no importa (puede ser `.dat`, `.bin`, etc.).
- Se usan en muchos programas, como **juegos** que almacenan datos del progreso o configuraciones.

Ejemplo:

Si escribes “128” en un archivo de texto, el disco guarda **4 bytes** (uno por cada carácter: `'1'`, `'2'`, `'8'`, y el *enter*).

En cambio, si lo escribes directamente en binario, ocupa **solo 1 byte**.

> Conclusión: el binario es mucho más eficiente, especialmente con grandes volúmenes de datos.

Ventajas del binario

- Ahorra espacio (no hay separadores, saltos de línea ni caracteres extra).
- Permite representar **información compleja y compacta**.
- Es más **rápido de escribir y leer** (lecturas directas de bloques).

Archivos binarios de acceso secuencial

- Los datos se guardan **en bloques consecutivos** (no se pueden saltar ni reordenar).
- Se usan, por ejemplo, para **guardar una lista enlazada** o **el estado de un juego**.

Ejemplo:

Una lista enlazada en memoria:

```
Nodo1 → Nodo2 → Nodo3 → Nodo4
```

Se convierte en un archivo:

```
[Registro1][Registro2][Registro3][Registro4]
```

Cada registro ocupa un **bloque fijo de bytes** (por ejemplo, 12 bytes).

> Los punteros de memoria no se guardan, porque cambian cada vez que el programa se ejecuta.

Cómo se lee y escribe

- Se escribe con funciones tipo `fwrite()` y se lee con `fread()`.
- Se leen bloques de tamaño fijo (por ejemplo, 12 bytes por registro).
- El proceso termina cuando `fread()` devuelve 0 bytes (fin del archivo).

Comparación: Tipos de acceso

Tipo de archivo	Características	Ventajas	Desventajas
Texto plano	Legible, usa separadores	Simple, editable	Ocupa más espacio
Texto con longitud (híbrido)	Guarda tamaño de campo	No pierde info	Más lecturas (lento)
Binario fijo	Bloques iguales	Rápido y compacto	Puede recortar info
Binario secuencial	Guarda todo en orden	Eficiente y rápido	No permite saltar registros

Conclusión

Los **archivos binarios secuenciales**:

- Guardan información en **bloques consecutivos**.
- No tienen separadores ni delimitadores.
- Son **eficientes y compactos**, pero no permiten acceso directo a un registro específico.

  (Para eso se usarán los **archivos binarios de acceso directo**, en el siguiente video.)