Random access files

Random access files allow programs to read from or write to any position in a file without sequentially processing the entire file. This feature is useful for scenarios like databases, where access to specific parts of large files is needed.

In C, random file access is achieved using functions from the standard I/O library, specifically fseek(), ftell(), and fread()/fwrite().
Key Functions:

- fseek(FILE *stream, long offset, int whence): Moves the file pointer to a specific location.
- ftell(FILE *stream): Returns the current position of the file pointer.
- fread(): Reads binary data from a file.
- fwrite(): Writes binary data to a file.

Explanation:

- fseek(file, 0, SEEK_SET): Moves the file pointer to the beginning of the file.
- fwrite(buffer, size, count, file): Writes data starting at the current file pointer.
- fread(buffer, size, count, file): Reads data starting at the current file pointer.
- fopen("example.bin", "wb+"): Opens the file for read/write in binary mode (b), and the plus (+) allows both read and write operations.

C buffering is handled at a more manual level with standard I/O functions or custom logic.
Java abstracts buffering into specialized classes (like BufferedReader, BufferedWriter), making it easier to handle larger data transfers without manually managing memory.FeatureC (Standard I/O Buffering)Java (Buffered Streams)Buffered OutputBy default, C streams are buffered (line-buffered for stdout in the terminal).
Functions: fflush() to force flush the buffer.Java provides BufferedOutputStream to buffer output. Flushing is handled using flush() method.Buffered InputC can use fgets() to read a buffer of characters from an input stream (like a file or stdin).Java has BufferedInputStream and BufferedReader for reading large chunks of data efficiently.

In C, when you declare a buffer (like char buffer[BUFFER_SIZE]), it is typically stored in the stack (if small) or heap (if dynamically allocated).
In Java, objects like BufferedReader or manually defined buffers (e.g., char[]) are stored in the heap (as Java uses heap memory for object storage).