import java.io.IOException;

import java.nio.ByteBuffer;

import java.nio.channels.FileChannel;

import java.nio.file.Path;

import java.nio.file.Paths;

import java.nio.file.StandardOpenOption;

public class ReadLargeFile {

public static void main(String[] args) {

// Replace "path/to/large\_file.txt" with the actual path to your large file

String filePath = "path/to/large\_file.txt";

try {

Path path = Paths.get(filePath);

FileChannel fileChannel = FileChannel.open(path, StandardOpenOption.READ);

// Set the buffer size according to your requirements.

int bufferSize = 8 \* 1024; // 8KB buffer

long fileSize = fileChannel.size();

long bytesRead = 0;

while (bytesRead < fileSize) {

long remainingBytes = fileSize - bytesRead;

int chunkSize = (int) Math.min(bufferSize, remainingBytes);

// Create a mapped byte buffer for the chunk

ByteBuffer buffer = fileChannel.map(FileChannel.MapMode.READ\_ONLY, bytesRead, chunkSize);

// Process the chunk (you can do whatever processing you need here)

// For example, you can convert bytes to string and print them:

String chunkData = new String(buffer.array(), 0, chunkSize);

System.out.print(chunkData);

bytesRead += chunkSize;

}

fileChannel.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}