Here are scenario-based questions for students to identify the appropriate ExecutorService thread pool type:

ExecutorService Thread Pool Quiz - Choose the Best Option

Instructions: For each scenario, choose the most appropriate thread pool type:

- A) Fixed Thread Pool
- B) Cached Thread Pool
- C) Single Thread Executor
- D) Scheduled Thread Pool

Question 1: Web Server Request Processing

You're building a web server that needs to handle HTTP requests. The server typically receives 50-100 concurrent requests, and each request takes 200-500ms to process. You want to limit resource usage and maintain consistent performance.

Which thread pool should you use?

Question 2: File Processing Pipeline

You have a system that processes uploaded files one by one. Each file must be processed in the exact order it was uploaded (FIFO). Processing involves validation, transformation, and database updates that must happen sequentially to maintain data consistency.

Which thread pool should you use?

Question 3: Microservice with Burst Traffic

Your microservice handles API calls that come in unpredictable bursts. Sometimes you get 5 requests per minute, other times 200 requests in 10 seconds. Each request is lightweight (50-100ms processing time). You want to minimize resource usage during quiet periods but handle bursts efficiently.

Which thread pool should you use?

Question 4: System Health Monitoring

You need to implement a monitoring system that:

- Checks database connectivity every 30 seconds
- Sends heartbeat signals every 5 minutes
- Cleans up temporary files every hour
- Generates daily reports at 2 AM

Which thread pool should you use?

Question 5: Image Processing Service

Your application processes high-resolution images (resize, apply filters, generate thumbnails). Each image takes 2-5 seconds to process. You have an 8-core server and want to maximize CPU utilization while preventing system overload.

Which thread pool should you use?

Question 6: Chat Application

You're building a chat application where messages from each user must be processed in order to maintain conversation flow. However, messages from different users can be processed concurrently. Each user's messages should be handled by the same thread to ensure ordering.

Which thread pool should you use?

Question 7: Email Notification System

Your system sends various types of notifications:

- Welcome emails (send immediately)
- Password reset emails (send immediately)
- Weekly newsletters (send every Sunday at 9 AM)
- Monthly reports (send on 1st of each month)
- Reminder emails (send 24 hours after user signup)

Which thread pool should you use?

Question 8: Log File Analyzer

You have a single log file that needs to be processed line by line. Each line must be parsed, analyzed, and results written to a database. The order of processing matters because later entries might reference earlier ones.

Which thread pool should you use?

Answer Key:

1. **A) Fixed Thread Pool** - Predictable load, need resource control
2. **C) Single Thread Executor** - Sequential processing required
3. **B) Cached Thread Pool** - Unpredictable burst traffic, short tasks
4. **D) Scheduled Thread Pool** - Time-based recurring tasks
5. **A) Fixed Thread Pool** - CPU-intensive, want to match core count
6. **C) Single Thread Executor** - Order preservation per user (or multiple single-thread executors)
7. **D) Scheduled Thread Pool** - Mix of immediate and scheduled tasks
8. **C) Single Thread Executor** - Sequential processing with dependencies

---

Bonus Challenge Questions:

Question 9: What happens if you use `scheduleAtFixedRate(task, 0, 1, SECONDS)` but each task takes 2 seconds to complete?

Question 10: Why might using a Cached Thread Pool for CPU-intensive tasks be problematic on a 4-core machine?

Would you like more details about any of these scenarios or additional questions?

Question 1: Download Files

You need to download 3 files from the internet. Each download takes about 5 seconds. You want to download all files at the same time and then print "All downloads complete!" only after ALL files are finished.

```
// Which approach is correct?

// Option A
Thread t1 = new Thread(() -> downloadFile("file1.txt"));
Thread t2 = new Thread(() -> downloadFile("file2.txt"));
Thread t3 = new Thread(() -> downloadFile("file3.txt"));
t1.start(); t2.start(); t3.start();
System.out.println("All downloads complete!");

// Option B
Thread t1 = new Thread(() -> downloadFile("file1.txt"));
Thread t2 = new Thread(() -> downloadFile("file2.txt"));
Thread t3 = new Thread(() -> downloadFile("file3.txt"));
t1.start(); t2.start(); t3.start();
t1.join(); t2.join(); t3.join();
System.out.println("All downloads complete!");
```

Question 2: Pizza Restaurant

A pizza restaurant has:

1 chef making pizzas (puts pizzas in a queue) 3 delivery workers taking pizzas from the queue and delivering them The chef makes 1 pizza every 2 seconds. Each delivery takes 5 seconds.

What type of thread pool should the delivery workers use?

A) Single Thread Executor (only 1 delivery person works) B) Fixed Thread Pool with 3 threads (all 3 delivery people work) C) Cached Thread Pool (unlimited delivery people)