Thread Concepts and Thread Pools

Multithreading in Java allows concurrent execution of two or more threads, which are lightweight
processes within a program. It is a powerful feature for improving performance, particularly in applications that require high efficiency and responsiveness.

Threads

A thread is a small part of a program that allows it to do several tasks at the same time. Threads share the same memory space but run independently, which can significantly improve the efficiency of a program.

Key Thread Concepts:

1. Thread Creation: Threads in Java can be created by either implementing the Runnable interface
or extending the Thread class.

2. Thread Lifecycle: A thread goes through several states during its lifetime: New, Runnable, Blocked, Waiting, Timed Waiting, and Terminated.

3. Thread Synchronization: Synchronization ensures that only one thread can access a shared resource at a time, preventing data inconsistency and corruption.

A thread pool manages a pool of worker threads, which are reused to execute tasks. The main advantage of using thread pools is to limit the number of threads that can be created, reducing resource consumption and improving performance.

Key Concepts of Thread Pools:

1. Executor Framework: Java provides the Executor framework, which includes the Executor, ExecutorService, and ScheduledExecutorService interfaces for managing and controlling thread execution.

2. Fixed Thread Pool: A fixed thread pool contains a fixed number of threads. If all threads are busy,
new tasks are queued until a thread becomes available.

3. Cached Thread Pool: A cached thread pool creates new threads as needed but reuses previously
created threads when they are available.

4. Single Thread Executor: This creates a single worker thread to execute tasks sequentially, useful
for scenarios where tasks must be performed in order.
5. Scheduled Thread Pool: This thread pool is used for scheduling tasks to run after a given delay or
periodically.
Using Thread Pools: Thread pools in Java are created using the Executors class, which provides factory methods for different types of thread pools.
\