JAVA concurrent:-

Concurrent:- Dealing with lot of thing at once

Parallelism:- Doing lot of thing at once

Out of Order Execution :- For improvement the performance compiler or JVM perform out of Order Execution with affecting the semantic

Example :- a=3 a=3

B=4 a=a+3

A=a+3 b=4

JVM Memory model:- JVM memory model specification in-force the visibility of variable

Happened before relationship:-

Happened before relationship state that if any volatile is updated then

All volatile or non-volatile variable will so their visibility after reading the volatile variable.

Happened before relationship can be applicable by

Lock

Synchronization

Semaphore:- Semaphore is locking mechanism of use of limited resource

Some important use case:- limited database connection

Mtd:- Try-Acquire, Try-Acquire(Timeout )

Acquire, Available permit ,AcquireUnIntreptiuly

Release, semaphore(count , fairness)

Fork-join-pool:-

Fork-join-pool use to contain a pool of thread which as potently a blocking-queue of task

Each of available thread will pick task in FIFO order and execute.

Before execution of actual task is will be divided in multiple sub task and pushed in duck(dequeue)

and after finishing the original task thread will return to thread pool

Work stealing(pre fork join ):- Another thread will take task from end side

Best practice :-

* Avoid synchronization
* Local variable
* I/0 blocking
* Wait or sleep

Example :- fibonacci series

Synchronous queue :- Blocking Queue with size 1.