**1. What is meant by a thread in runnable state?**

When a thread is started, it is made runnable. Resources are allocated to the thread.

**2. Which method invokes the run() method?**

The start method.

**3. What is the time unit of sleep method?**

The time unit of the sleep method is milliseconds.

**4. What are the advantages / reasons of using threads instead of processes?**

* Better utilization of resources (threads share address space)
* Easier to control and manipulate threads than process
* Creating a thread is faster than creating another process.

**Note: Disadvantage –** stronger synchronization required

**5. Which are the two ways to implement threads?**

Sub-classing the thread class and overriding the run( ) method (implements Runnable or extends Thread).

**6. Which scheduling algorithm Java uses?**

Java uses a preemptive priority CPU scheduling algorithm.

**7. What is NORM\_PRIORITY (default priority) in Java?**

The range of Thread priority in [java](http://www.javaranch.com/) is 1-10; min is 1, max is 10. The default priority of any thread is 5.

**8. Explain Try Catch block inside sleep method.**

For the *wait( )* and *sleep( )* methods, if the thread that is interrupted and then blocked, the method that blocked the thread throws an InterruptException object.

**9. Thread Methods**

**(deprecated = not recommend to use)**

* suspend( ) suspends execution of the currently running thread. (the method is deprecated, deadlock for monitors)
* join( ) waits for this thread to die, causes the current thread to pause execution until this thread that called join() method terminates
* wait( ) on an object.
* sleep( ) puts the currently running thread to sleep for a specified amount of time (milliseconds)

**10. Advantages / Disadvantages****of SA’s (S.A. = Scheduling Algorithm)**

*Non-preemptive Scheduling:* Processes can remain on CPU until they terminate

*Shortest Job First:*

* The processes in the ready list, at the time when the CPU becomes available, are sorted in increasing order by the service time (next CPU burst) and scheduled for the CPU in that order.
* Advantages- optimal average waiting time (among all non-preemptive S.A.)
* Disadvantages- Not suitable for CPU scheduling (hard to predict length of next CPU burst),
* SJF may penalize processes with high service time requests - starvation

*FCFS:*

* The ready queue contains the PCB of the process and it is treated like a FIFO queue.
* Advantages- Easy to implement because there are no computations
* Disadvantages- Large processes in front of the queue can create convoy effect🡪 creates poor average waiting/ turnaround time

*Preemptive Scheduling:* The schedule can forcibly remove a process from the CPU/context switching

*Shortest Remaining Time:*

* Process with the shortest CPU burst is scheduled. If a new process (P2) arrives in the ready queue with a shorter service time, P2 is scheduled and P1 is preempted.
* Advantages- Optimal average waiting time (among all preemptive SA’s)
* Disadvantages- Not suitable for CPU scheduling (hard to predict length of next CPU burst)
* SJF may penalize processes with high service time requests - starvation

*Round Robin:*

* Used for Time-sharing environments. Processes are dispatched in a FIFO sequence but each process can only run for a limited amount of time, known as a **time-slice** or **quantum**.
* Advantages- fair in using CPU time
* Disadvantages- poor average wait and turnaround times
* Special case at Arrival time zero, where multiple processes can arrive at this time.

Threads:

* A traditional process (also named a heavyweight process) is a single task with one single thread.
* A thread is also called a lightweight process (LWP), and may consist of a program counter, register set and a stack space. All threads in a process share the same address space.
* A task consists of a collection of resources like: main memory (code section, data section), I/O devices, files.
* Multithreading refers to the ability of an operating system to support multiple threads within a single process.

Advantages:

* Better utilization of resources (threads share address space)
* Easier to control and manipulate threads than process
* Creating a thread is faster than creating another process.

Disadvantages:

* Operating System needs to support multithreaded environment. Synchronization is difficult and stricter because threads are sharing resources.
* Process Thread is the main thread and executes in main memory. If main thread terminates all threads terminate.

Thread types:

* Kernel level thread 🡪 communicate directly with kernel, system calls.
* User level thread 🡪 communicate above kernel, library calls.

Thread Process States:

**New:** an object for the thread is created. No system resources have been allocated yet.

start()Make a thread runnable. Resources are allocated to the thread; the thread goes into the Runnable state.

**Runnable**: yield()

**Blocked:** wait(), sleep(), join()

**Terminate**: exit(), abnormal exit by user, completion

**Dead state**: the thread exits (terminates).

If thread is in wait(); notify() or notifyall() must be called.

isAlive() – returns a Boolean value that determines if a Thread is in the Dead state or not.

### Windows

Provides time slicing for threads with same priority

**Show Round Robin diagram and Shortest Job First Diagrams:**