Top 13 Multithreading questions

Q1. Suppose, you have thread T1, T2, and T3. How will you ensure that thread T2 will run after T1 and thread T3 after T2?

Q2. Why do we call the start() method first, which in turn calls the run() method, why not directly call the run() method in our programs?

Q3: Explain the differences between User-level and Kernel level thread?

- **User-level threads are faster than kernel-level threads from the creation and managing perspective.**
- **User-level threads are generic, whereas the kernel-level threads are more specific to the concerned operating system.**
- **In the case of the user level, the multithreading process can't be implemented on multiprocessing, whereas kernel level can themselves be multithreaded.**

Q4: How will you awake a blocked thread in Java?

Q5: Which one is better to implement thread in Java? extending Thread class or implementing Runnable?

Q6: What's the difference between class lock and object lock?

Q7: Difference between t.start() and t.run() methods.

Q8: What happened if we are not overriding run() method:

Q9: What is the difference between wait and sleep in Java?

Q10: Which method will release lock?

Q11: What is a race condition? How will you find and solve race condition?

Q12: What are some common problems you have faced in multi-threading environment? How did you resolve it?

- **race conditions,**
- **deadlock**
- **Livelock-: When all the threads are in a blocked state and execution is stopped due to resource unavailability, then that situation is termed as livelock.**
- **Starvation**
-

Q13: Print sequence using 3 threads in java

```
T1 1
T2 2
T3 3
T1 4
T2 5
T3 6
T1 7
T2 8
T3 9
T1 10
```