7. Use the **Runnable** interface to code an application in which two threads print the odd and even numbers between 1 and 50, inclusive. One thread will print the odd numbers and its thread name; the other thread prints the even numbers and its thread name. Use the **yield**() method to alternate between the two threads. Assume each thread has the same priority. Write a driver to test your code. Save all files.

8. Based on Exercise 7 above, modify your code as follows:

a. Assign each thread a different priority. Run the code and observe the execution result.

Save all files.

b. Assign different sleeping times to each thread using the **sleep()** method to alternate execution between the two threads. Remove the existing **yield()** method. Run the code and observe the execution result. Save all files.

9. Why must threads be synchronized in application codes? List the ways to synchronize threads in coding.

Threads should be synchronized so we can call or modify objects from both threads. This way we can ensure one thread does not interrupt a task another thread is performing on that same object. This can be avoided through synchronizing the threads and making sure only one thread is operating on that object at one time.