**Exercise 1: Basic Thread Creation**

Question: Create a Java program that defines a class **MyThread** extending **Thread**. Override the **run()** method to print "Thread is running". Create an instance of **MyThread** and start it.

**Exercise 2: Runnable Interface**

Question: Develop a Java program that implements the **Runnable** interface using a class named **MyRunnable**. In the **run()** method, print "Runnable is running". Create a thread using an instance of **MyRunnable** and start it.

.

**Exercise 3: Thread Synchronization**

Question: Create a Java program that simulates a counter using a class **Counter**. The **Counter** class should have an **increment()** method that increments a private **count** variable. Use synchronization to prevent race conditions when two threads increment the counter concurrently.

**Exercise 4: Thread Priority**

Question: Write a Java program that creates two threads, **thread1** and **thread2**. Make **thread1** print even numbers and **thread2** print odd numbers. Set the priority of **thread1** to maximum and **thread2** to minimum.

**Exercise 5: Thread Join**

Question: Develop a Java program that creates two threads, **thread1** and **thread2**. Ensure that **thread2** doesn't start until **thread1** has finished executing.

**Exercise 6: Thread Deadlock**

Question: Create a Java program that causes a deadlock scenario using two threads and two resources. Illustrate how a cyclic dependency of locks can lead to a deadlock.

**Exercise 7: Thread Interruption**

Question: Write a Java program that starts a thread and interrupts it after a short delay. Inside the thread, continuously print "Thread is running..." until it's interrupted, then print "Thread interrupted."

**Exercise 8: Thread Local**

Question: Develop a Java program that uses the **ThreadLocal** class to store and access an integer value separately for each thread. Print the thread-specific value in each thread's context.

**Exercise 9: Volatile Keyword**

Question: Write a Java program that declares a **volatile** boolean flag. Create two threads: one that continuously checks the flag and waits until it becomes **true**, and another that changes the flag to **true** after a short delay.

**Exercise 10: Thread Pool**

Question: Develop a Java program that utilizes a thread pool to execute ten tasks concurrently. Each task should print a message indicating its task number and the thread that executed it.