S No.	Mini-project description
1	**Hare and Tortoise Race** This is a simple project which involves creation of multiple threads and explores some thread control mechanisms and thread priorities. a. Create two threads named 'Hare' and 'Tortoise'. The threads created should run 100 meters and the thread which finishes first 'wins the race'. When one of the threads wins the race, stop the second thread. Start both the threads and observe which thread finishes first. b. Since the Hare is faster than Tortoise, set a high priority to Hare thread and observe the results. c. Now modify the program so that the 'Hare' thread 'sleeps' for 1000 milliseconds after running 60 meters. Observe which thread wins the race. **How the thread is created:** We can create threads in two ways: Either by extending Thread class or by implementing Runnable interface. *(Explore -> Thread class and Runnable Interface)* ***[Explore More]*** **How the Thread runs 100 meters:** We can use a 'for loop' for iterating from 0 to 100. A print statement can be included so that we can see how far each thread has run. *(Refer the previous modules to see the usage of loops)*

	We can declare a variable to store the distance covered. The same variable is used to iterate the loop. **How to set Priorities to Threads:** Since the Hare is faster than Tortoise, we can set a higher priority to Hare thread. Make use of the static variables in the Thread class to select proper priority for each thread: ***[Explore More]*** **How to make the Hare 'sleep':** We can make use of some thread controlling mechanism to 'pause' the Hare thread for a specific period of time. ***[Explore More]*** Get the name of the thread to check which thread has to be suspended for 1000 milliseconds.