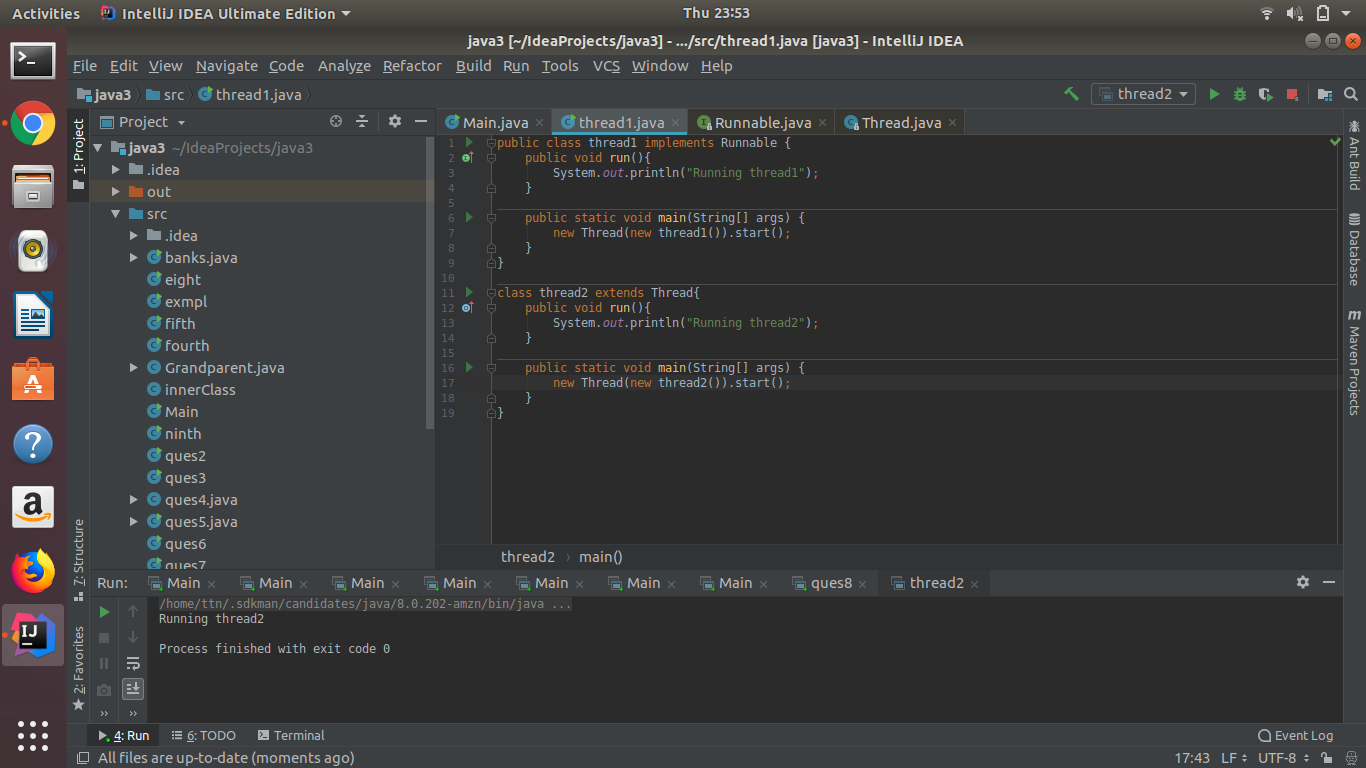
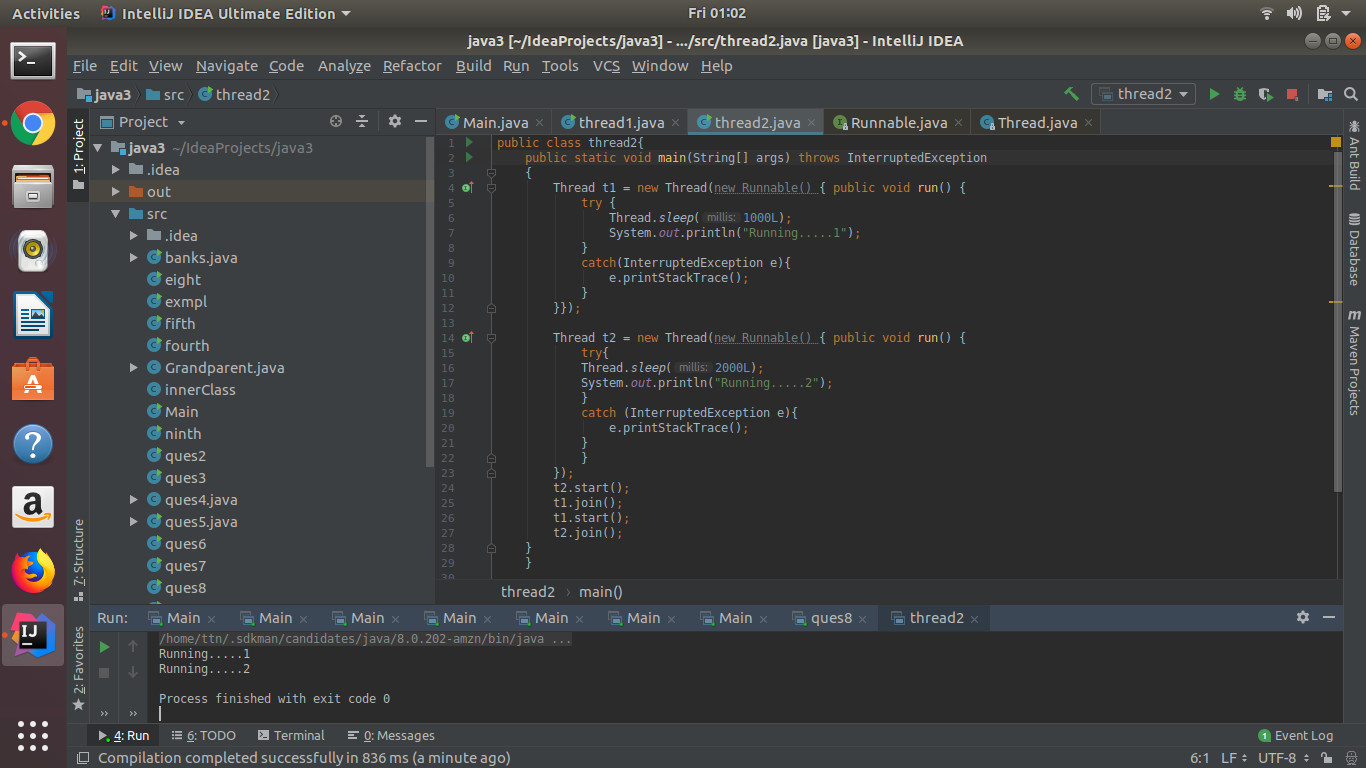
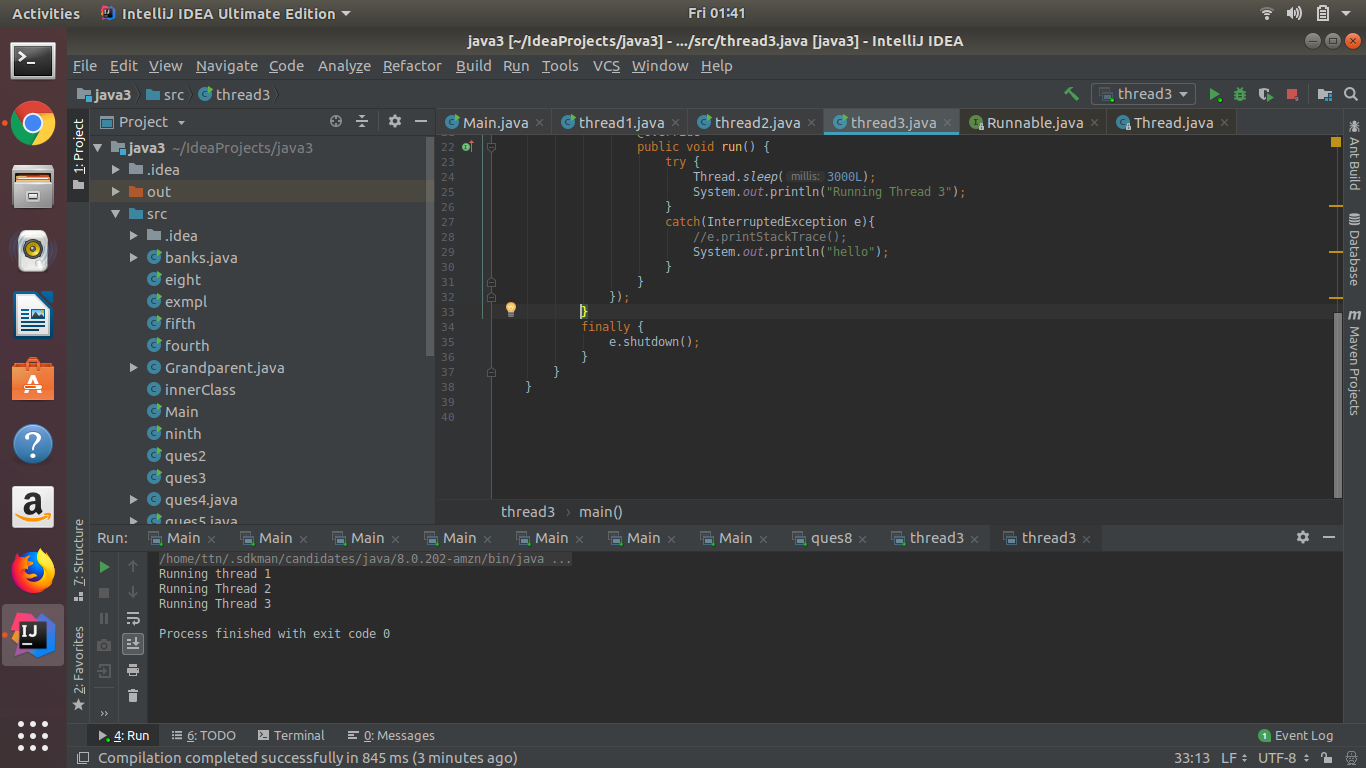
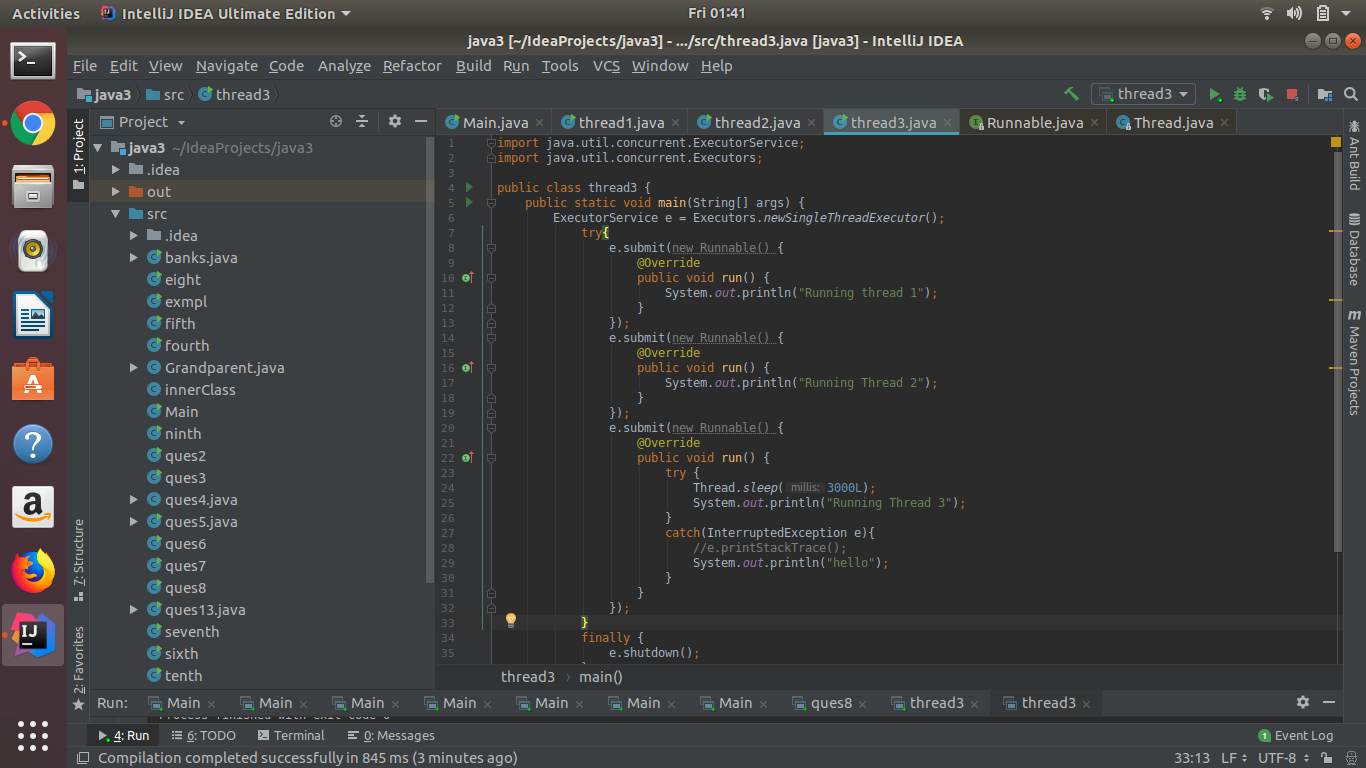
**1.Create and Run a Thread using Runnable Interface and Thread class.**

****

**2.Use sleep and join methods with thread.**

****

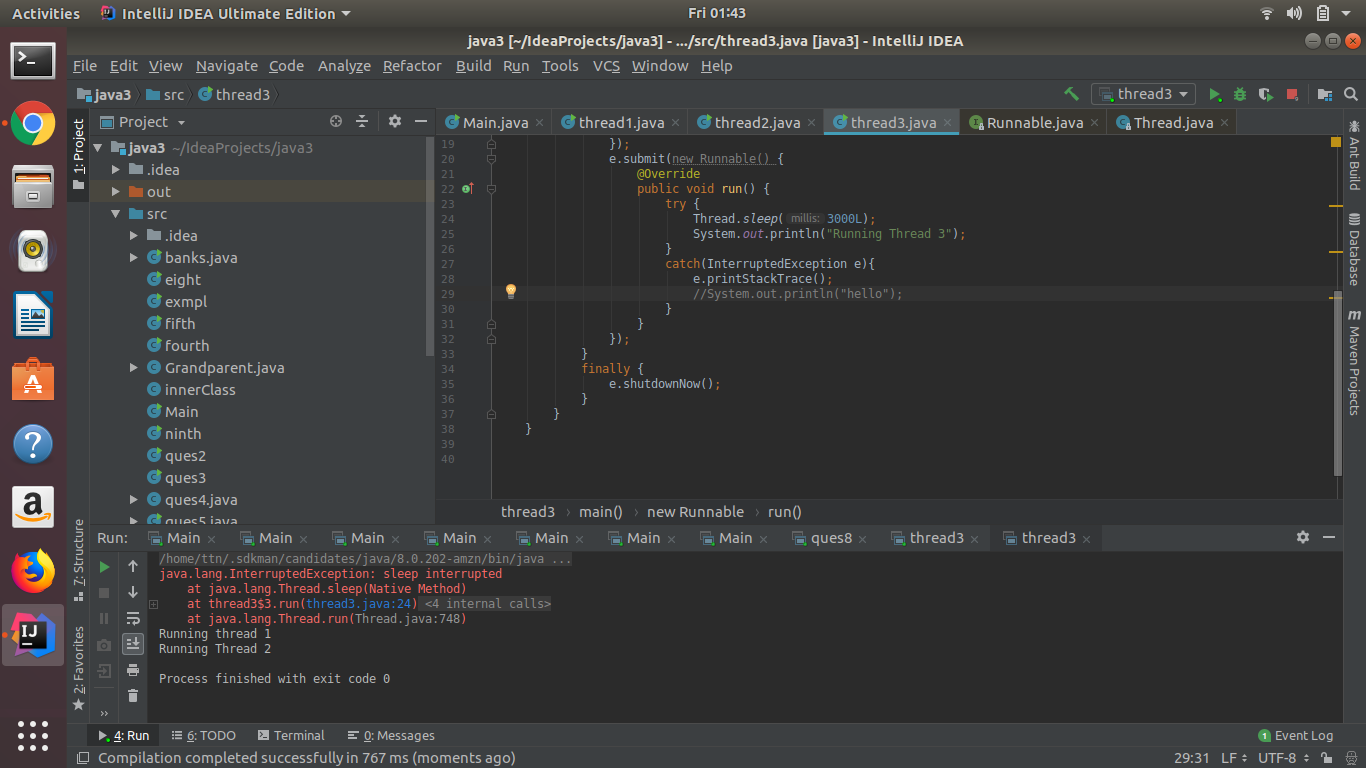
**3.Use a singleThreadExecutor to submit multiple threads.**

****

**4.Try shutdown() and shutdownNow() and observe the difference.**

**In above same code if we replace shutdown() by shutdownNow()**

**Then the currently waiting thread i.e. thread3 is been canceled without executed and output is given by e.printStackTrace().**

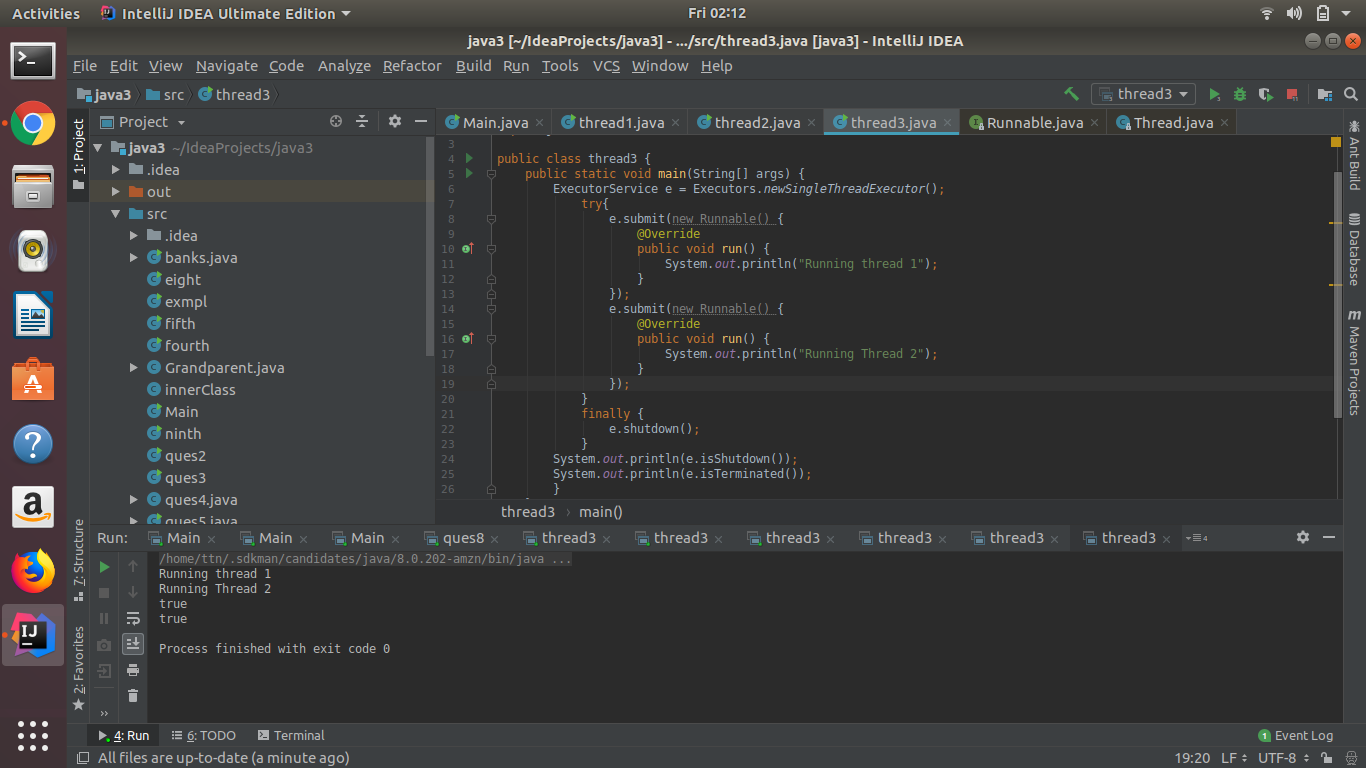
****

**5.Use isShutDown() and isTerminate() with ExecutorService.**

**If we do shutdown() and then do**

**: isShutDown() -> it gives true**

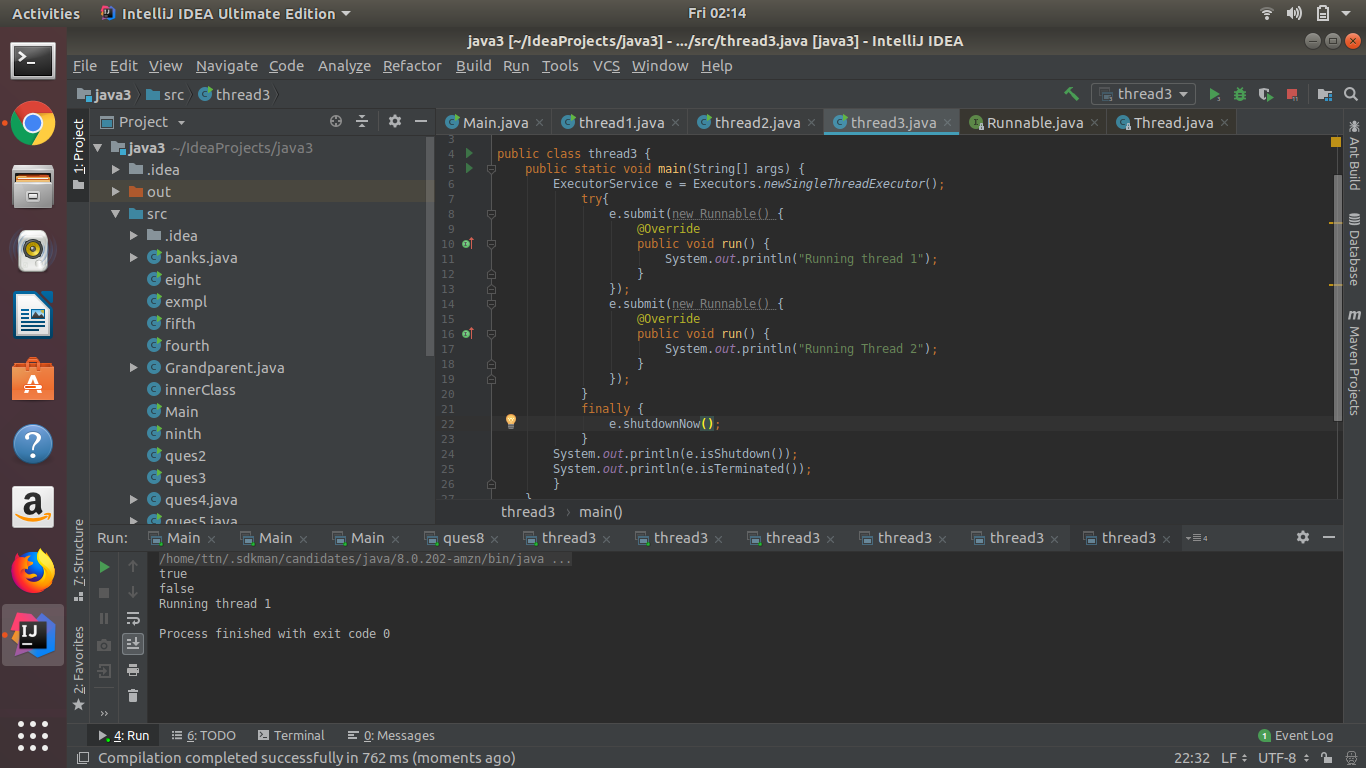
**: is Terminate() -> it gives true**

****

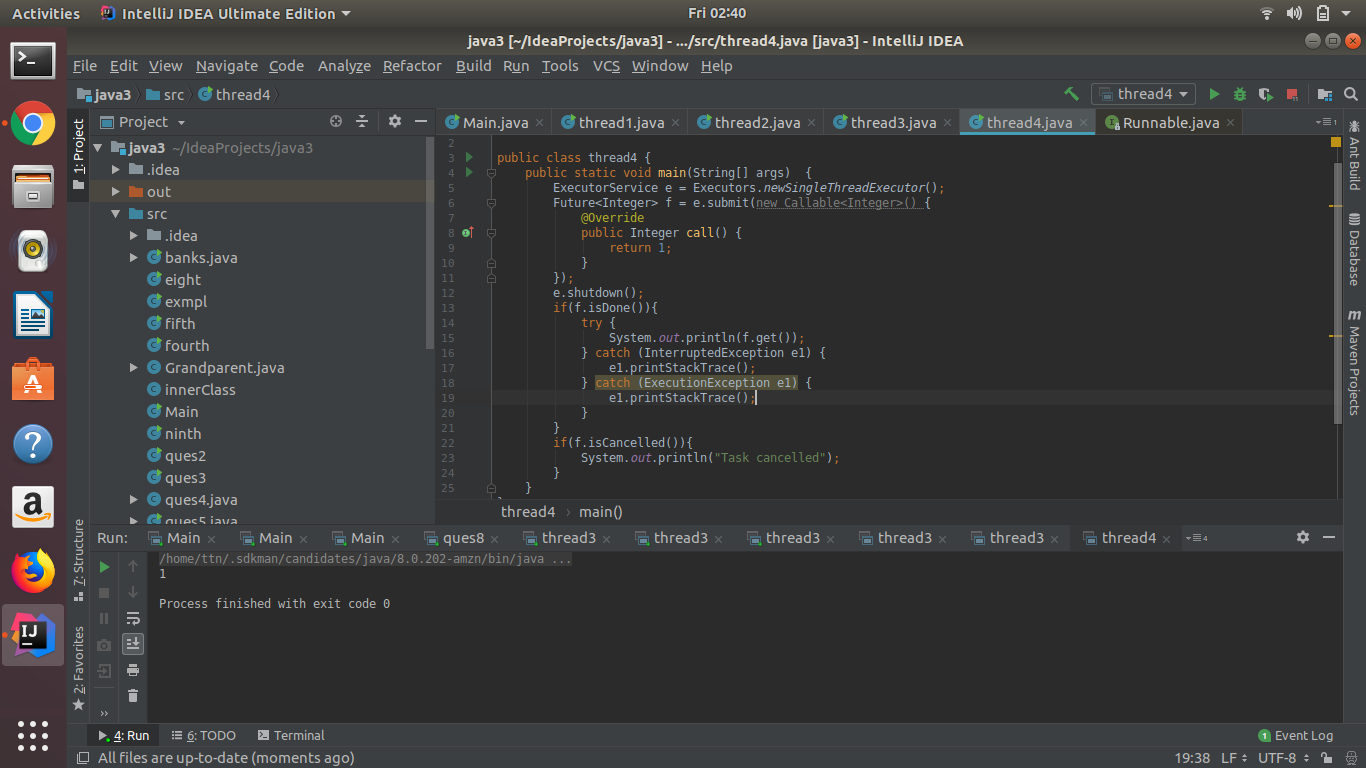
**If we do shutdownNow() and then do**

**: isShutDown() -> it gives true**

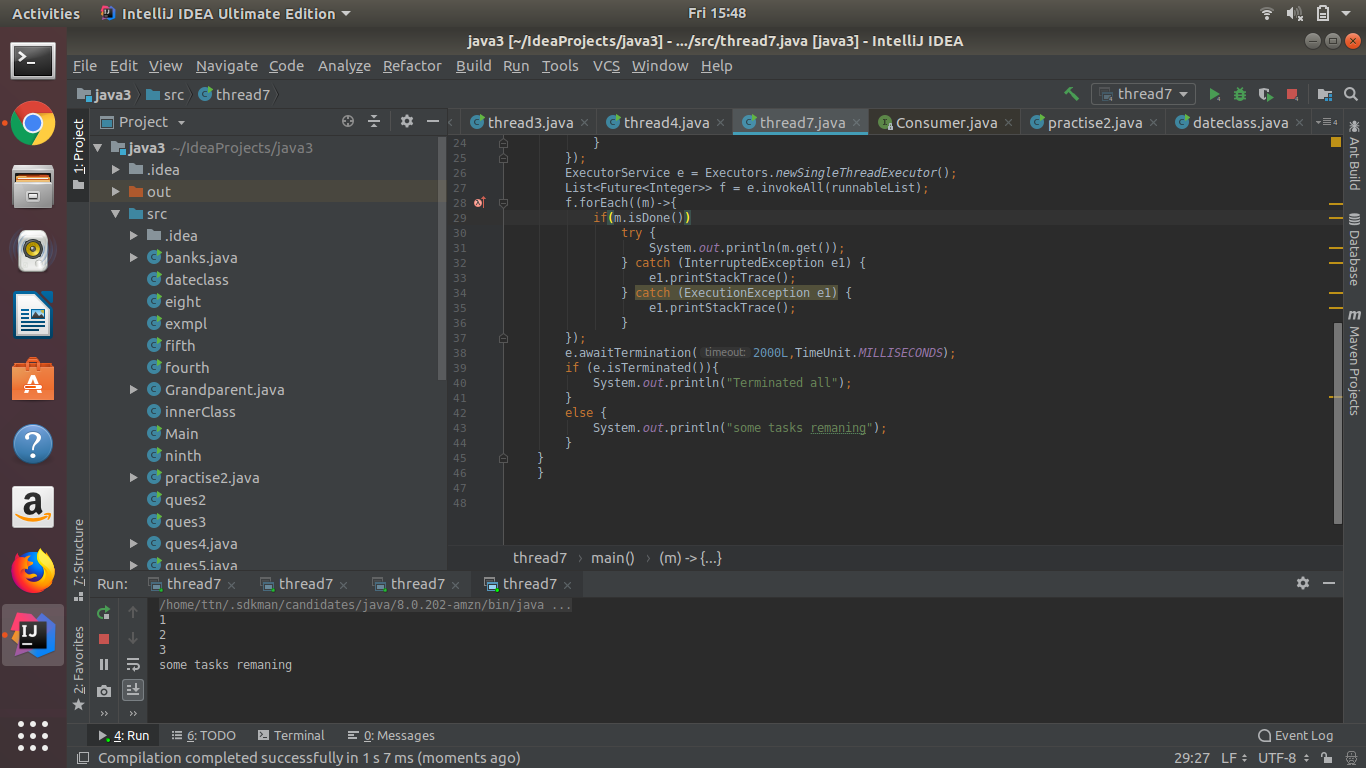
**:is Terminate() -> it gives false**

****

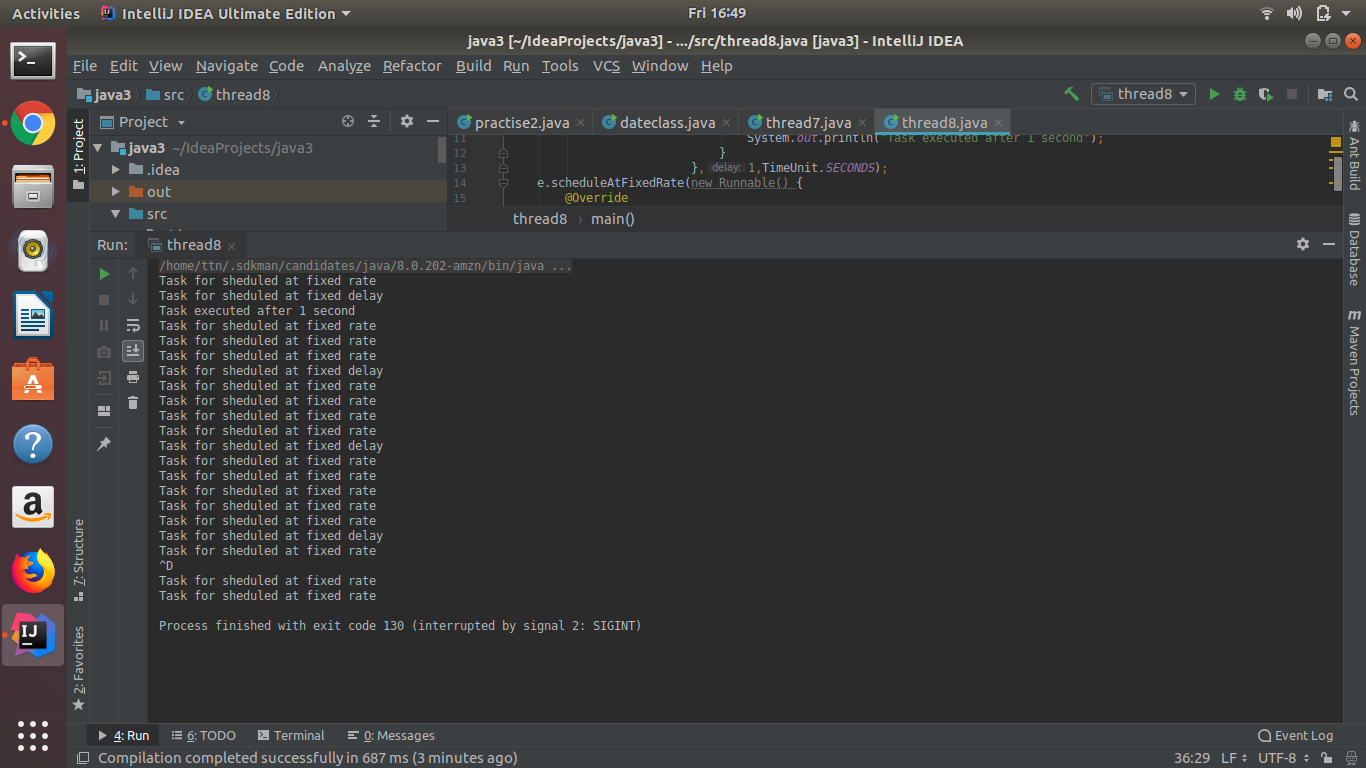
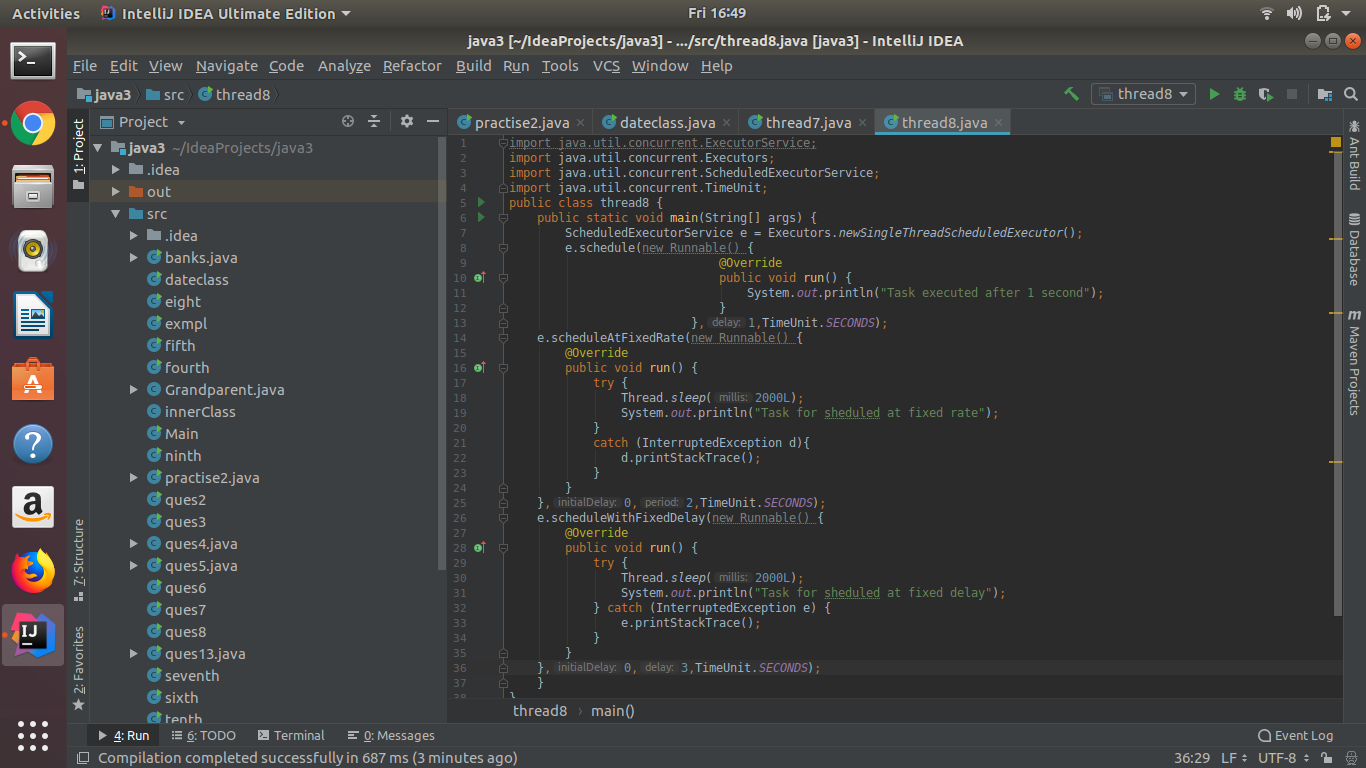
**6.Return a Future from ExecutorService by using callable and use get(), isDone(), isCancelled() with the Future object to know the status of task submitted.**

****

**7.Submit List of tasks to ExecutorService and wait for the completion of all the tasks.**

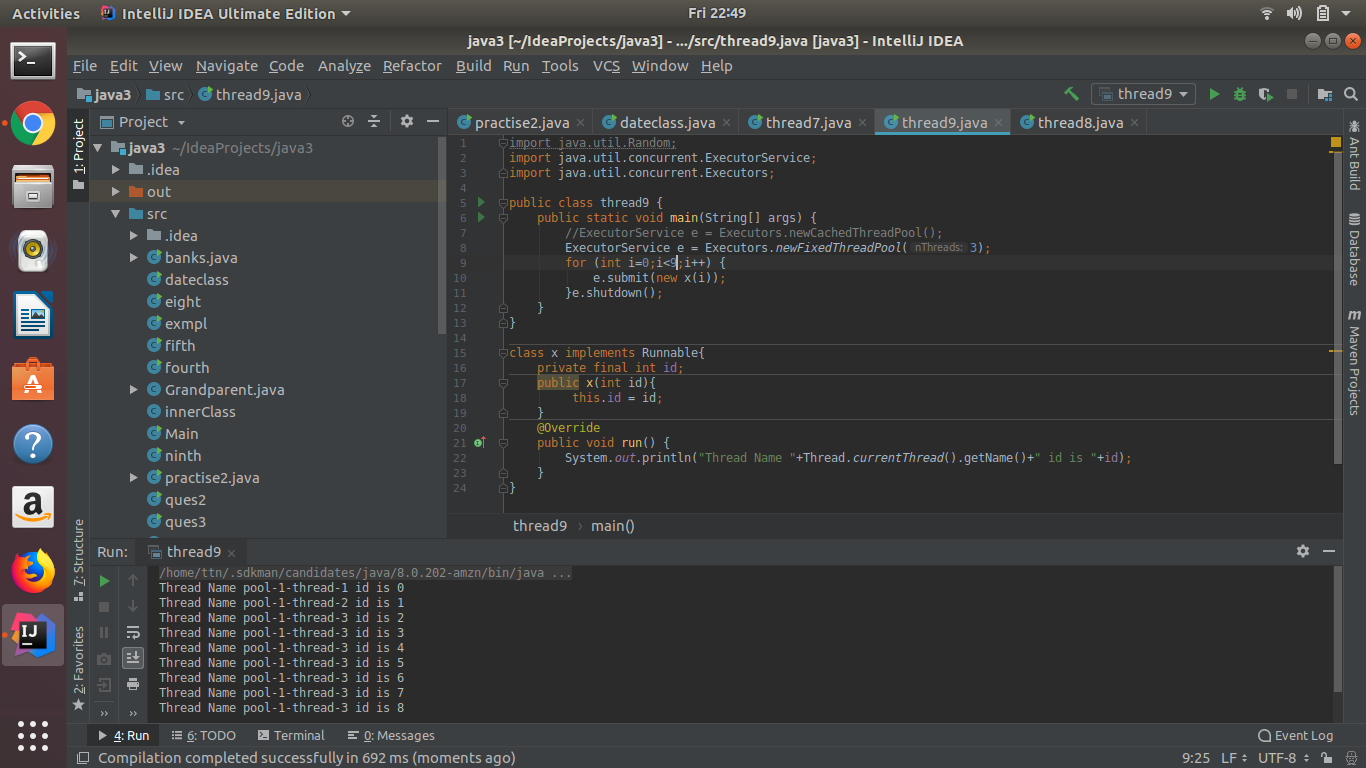
****

**8.Schedule task using schedule(), scheduleAtFixedRate() and scheduleAtFixedDelay()**

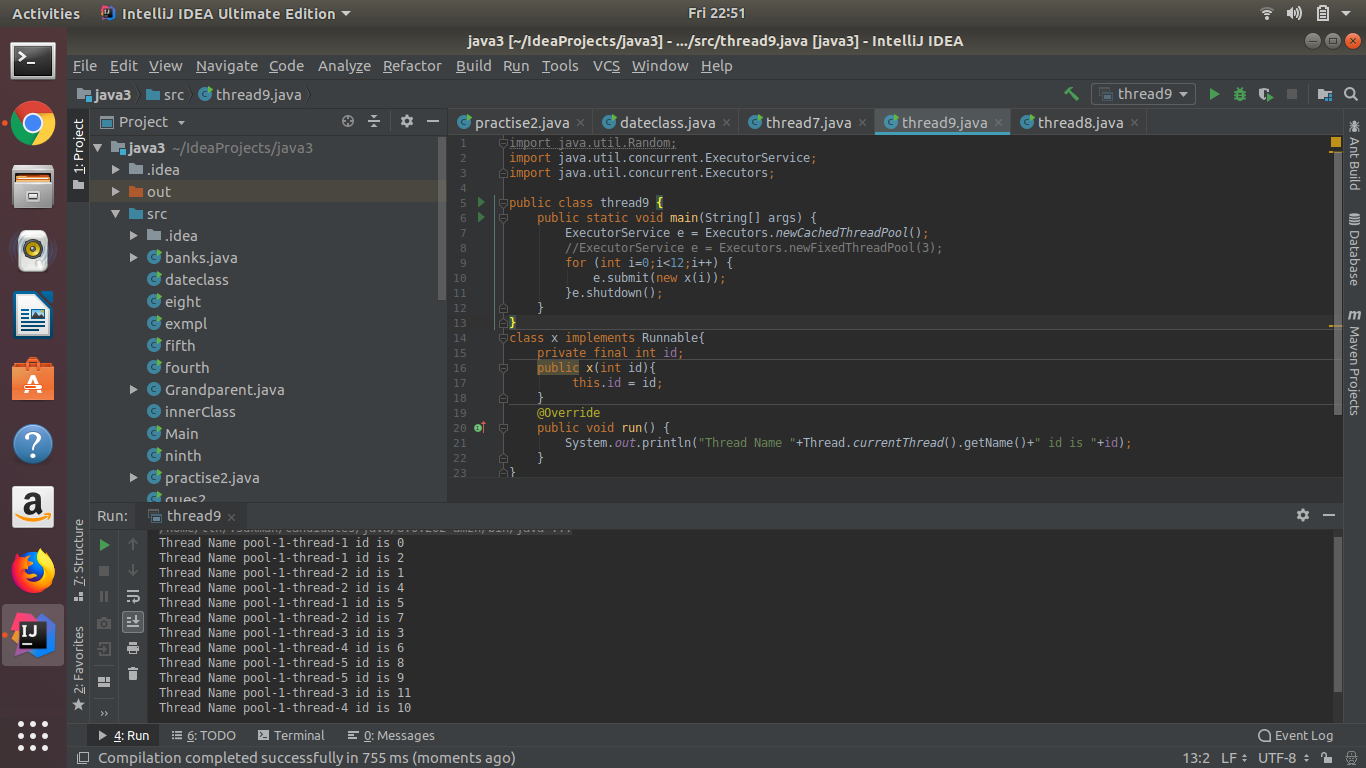
****

**9.Increase concurrency with Thread pools using newCachedThreadPool() and newFixedThreadPool().**

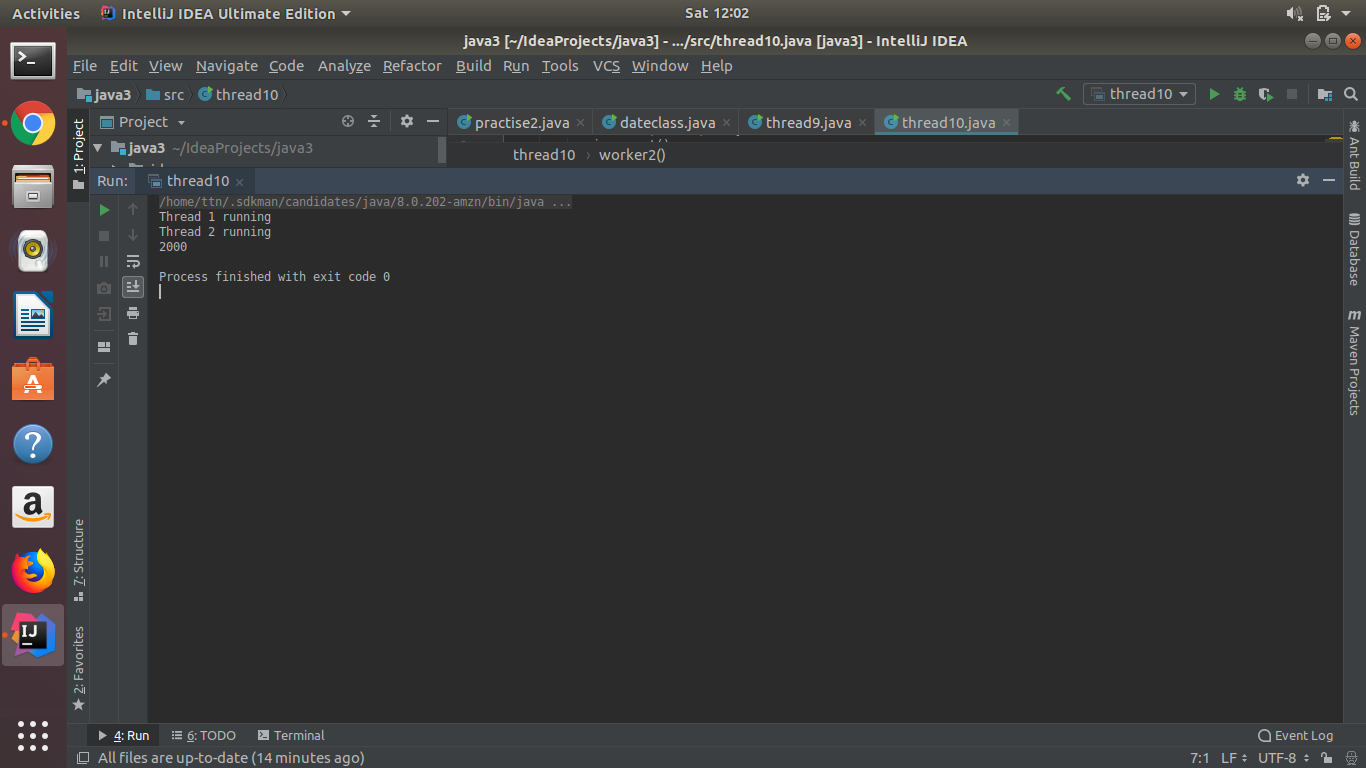
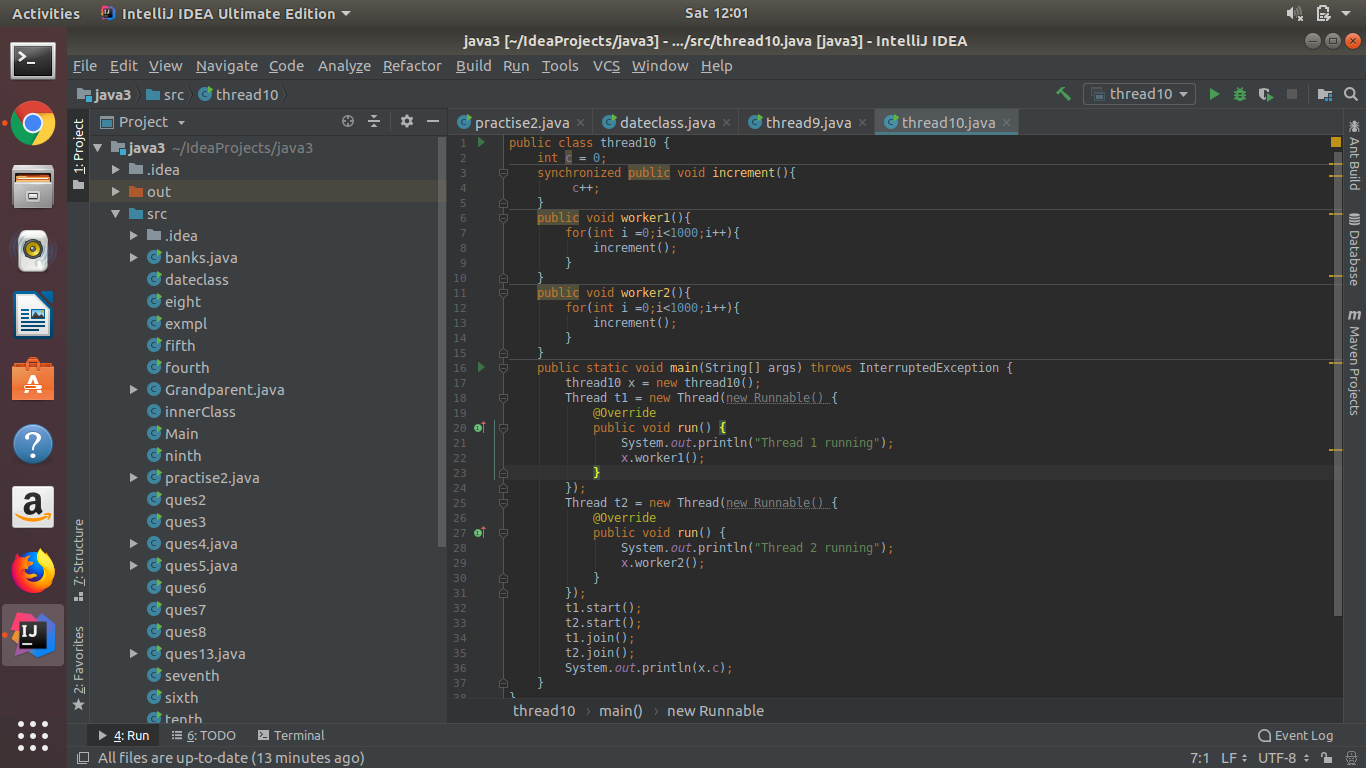
**newFixedThreadPool() ->**

****

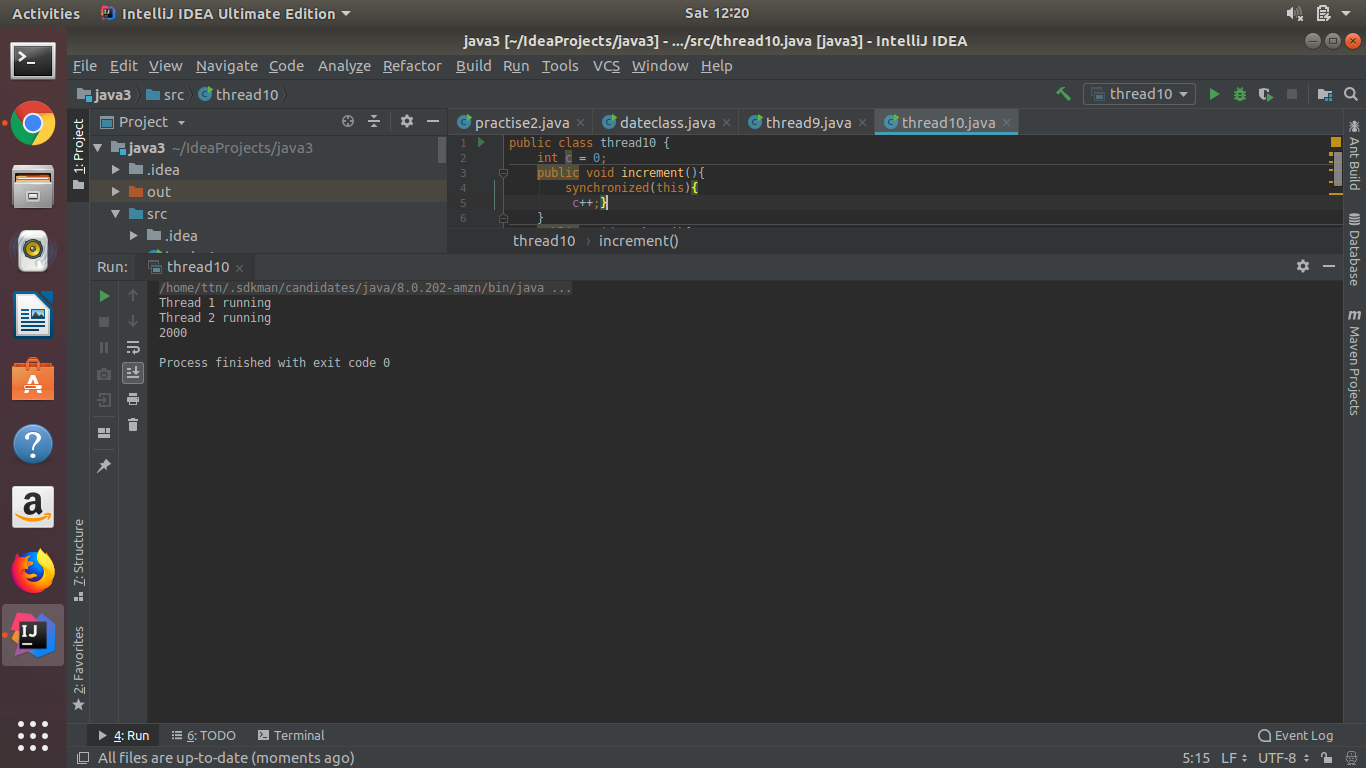
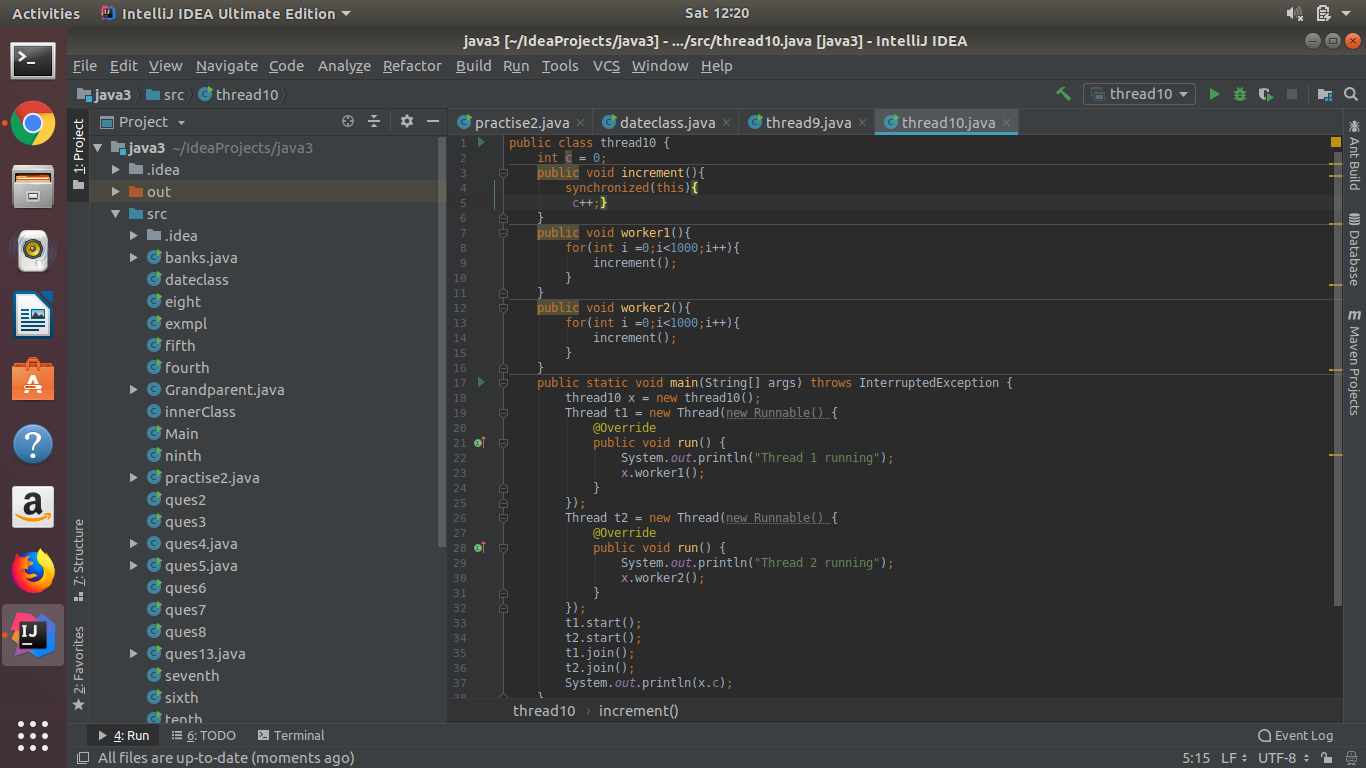
**And newCachedThreadPool() :->**

****

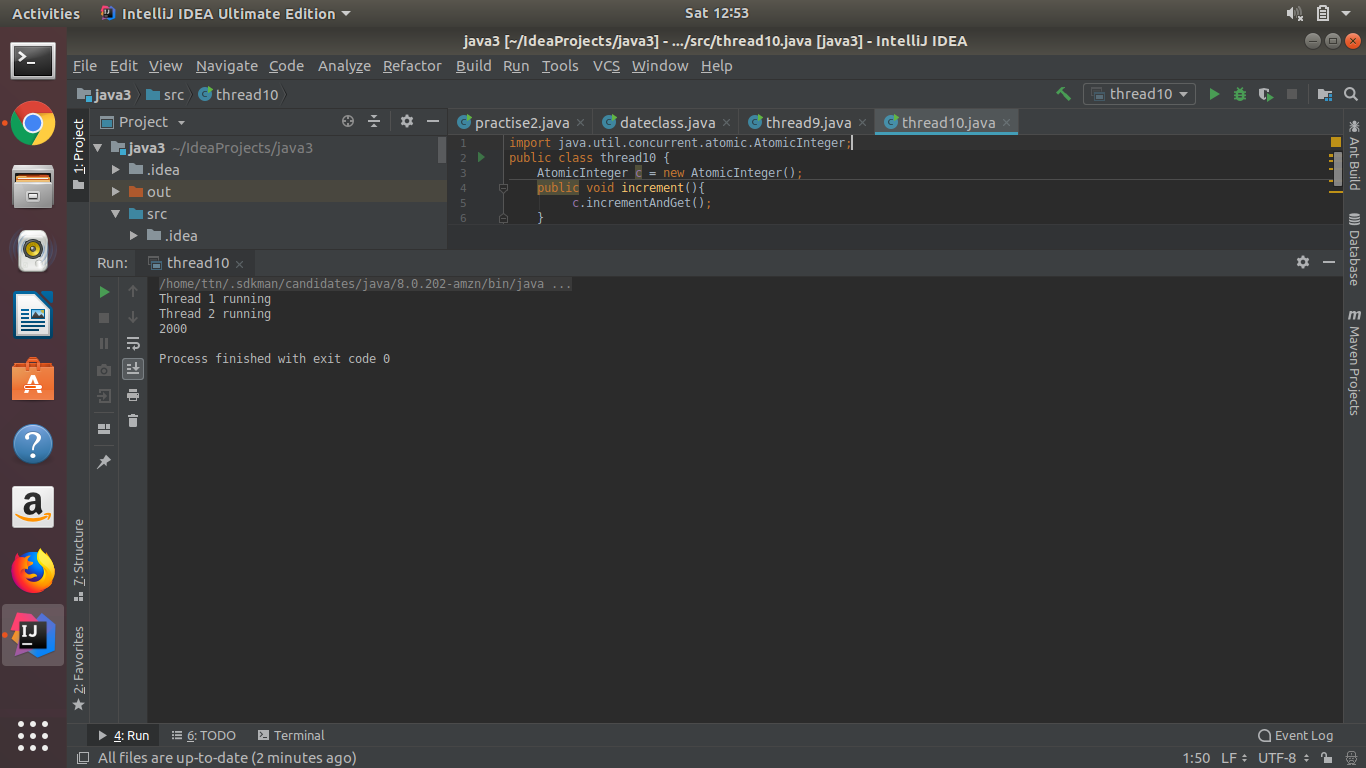
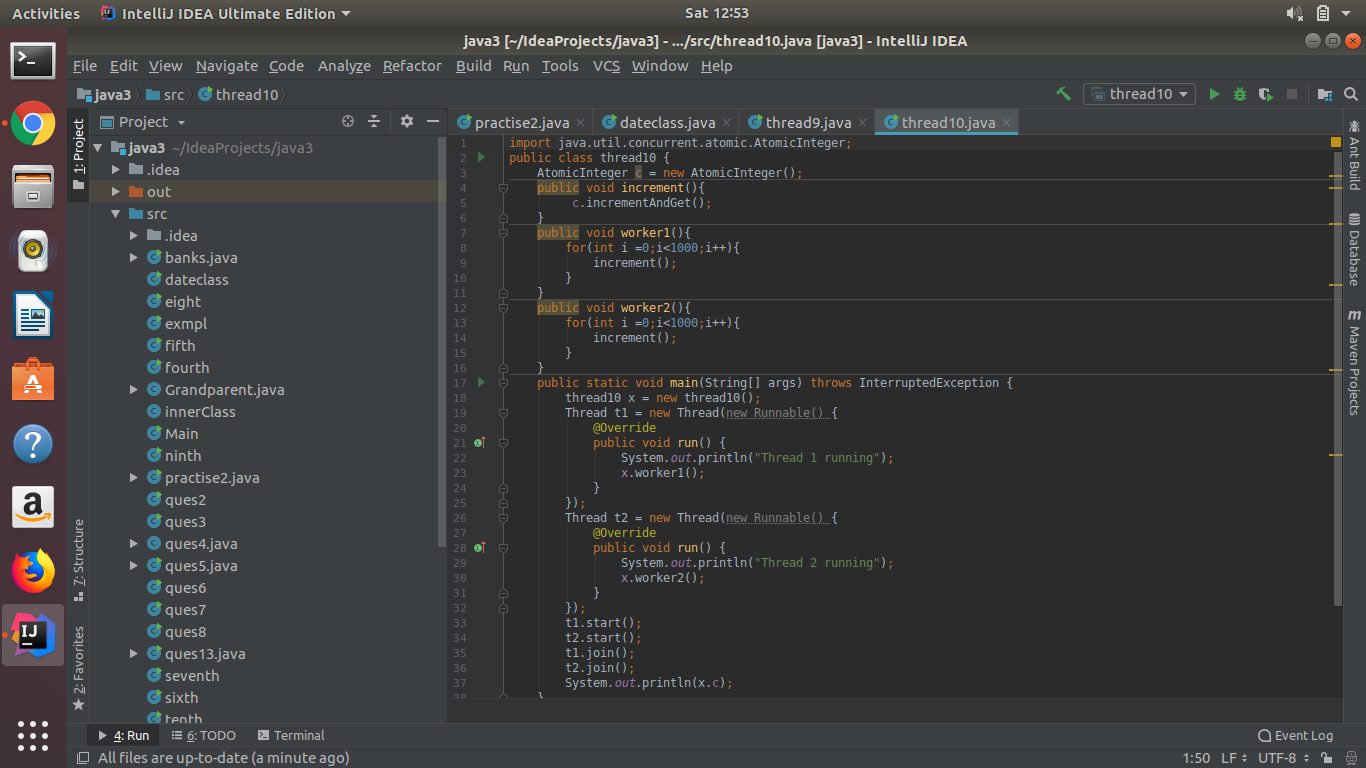
**10.Use Synchronize method to enable synchronization between multiple threads trying to access method at same time.**

****

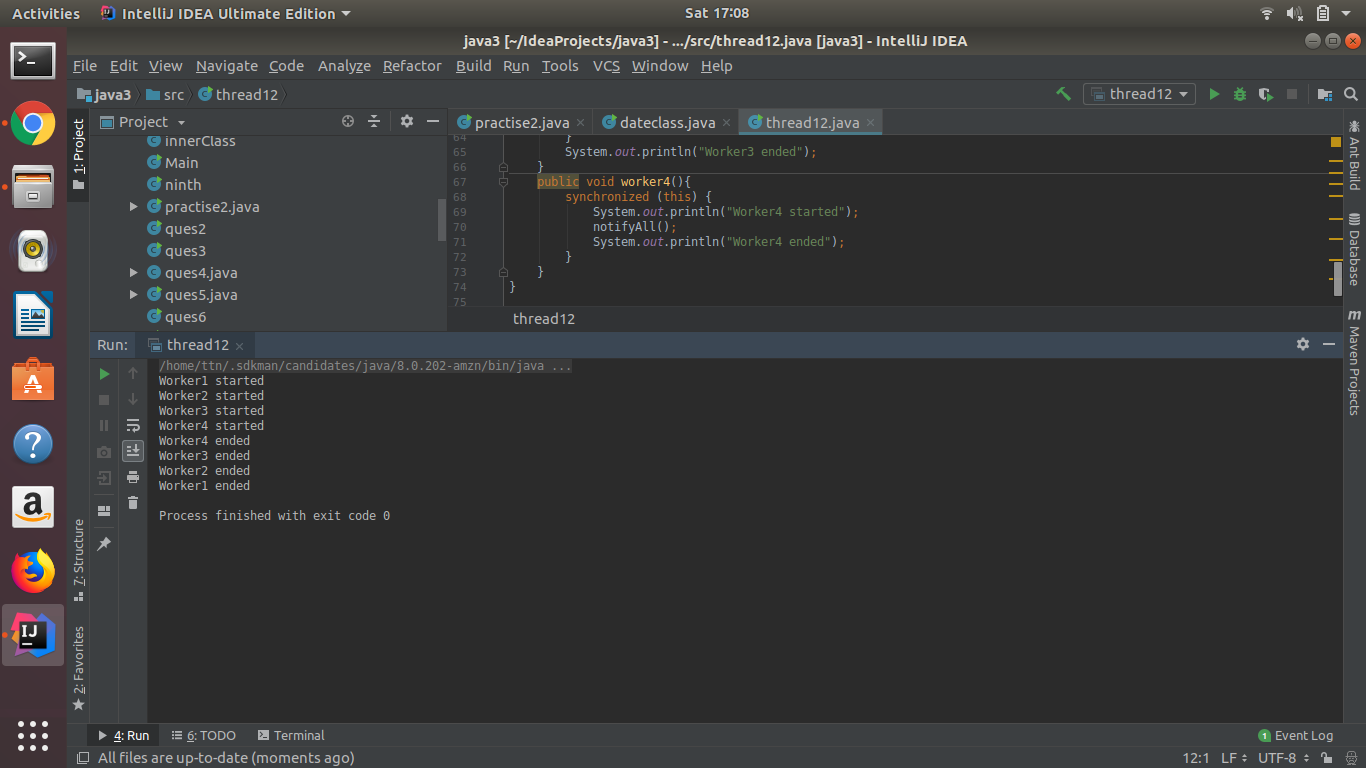
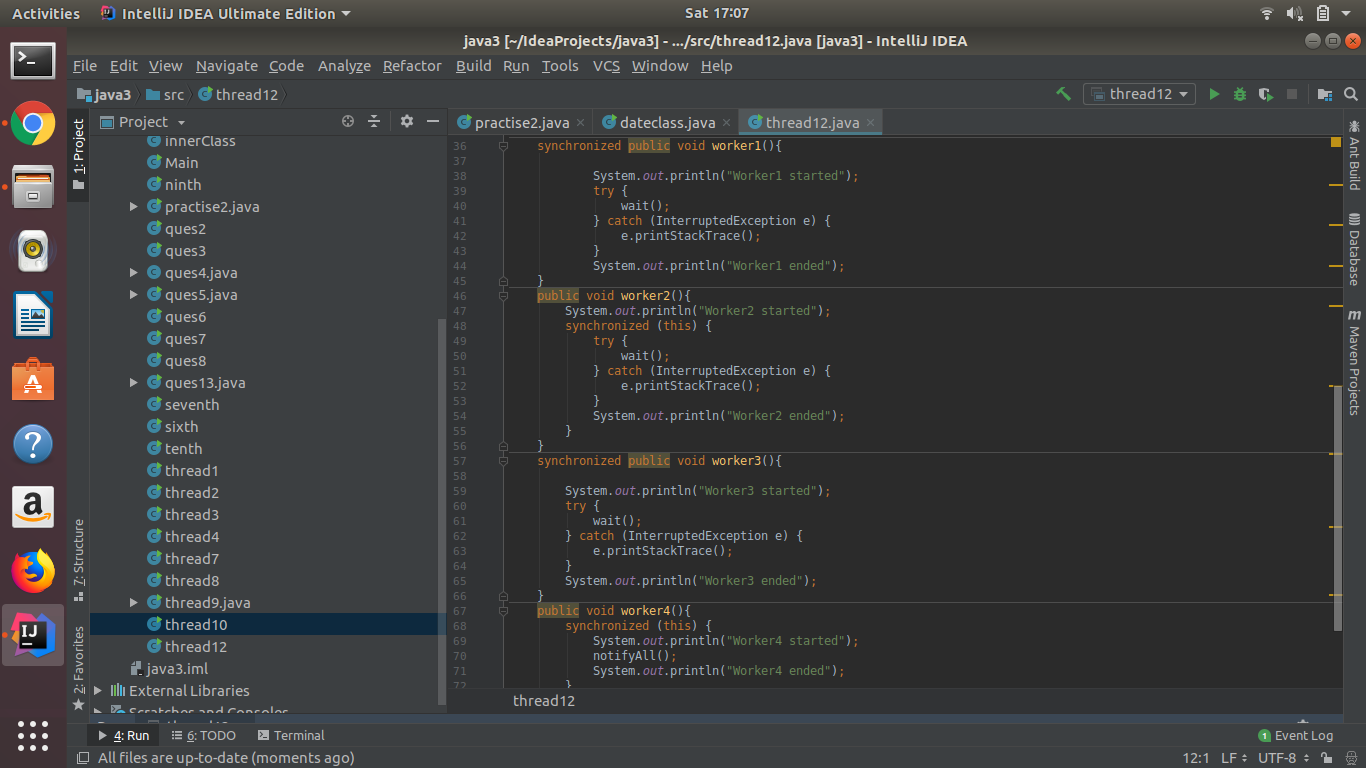
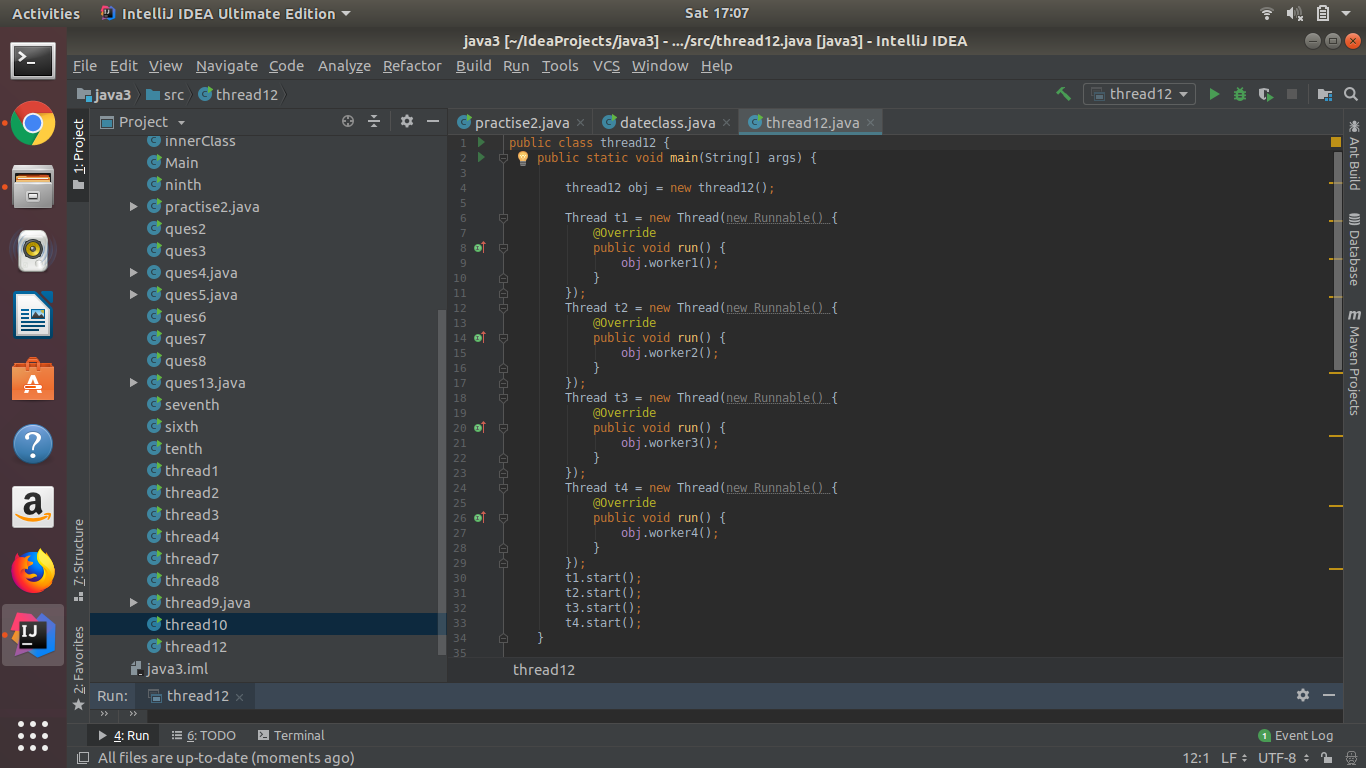
**11.Use Synchronize block to enable synchronization between multiple threads trying to access method at same time.**

****

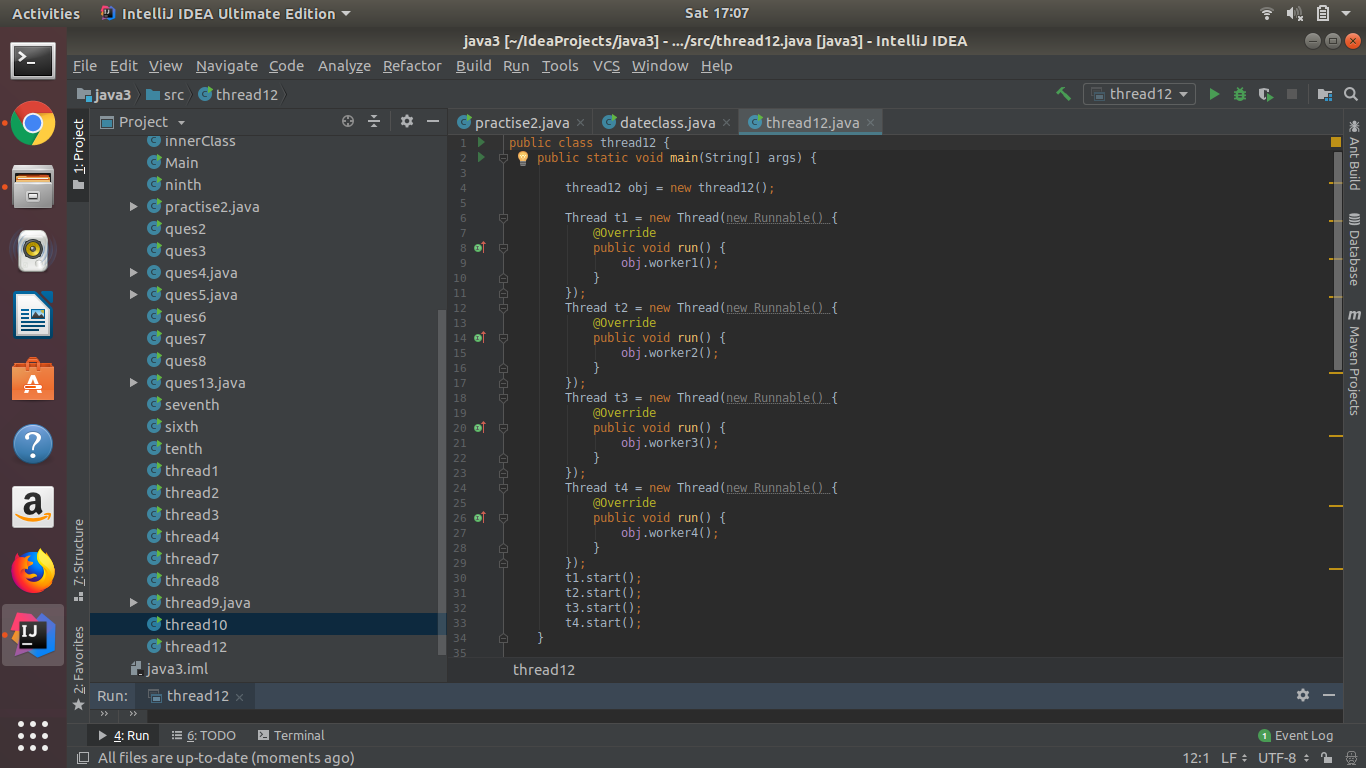
**12.Use Atomic Classes instead of Synchronize method and blocks.**

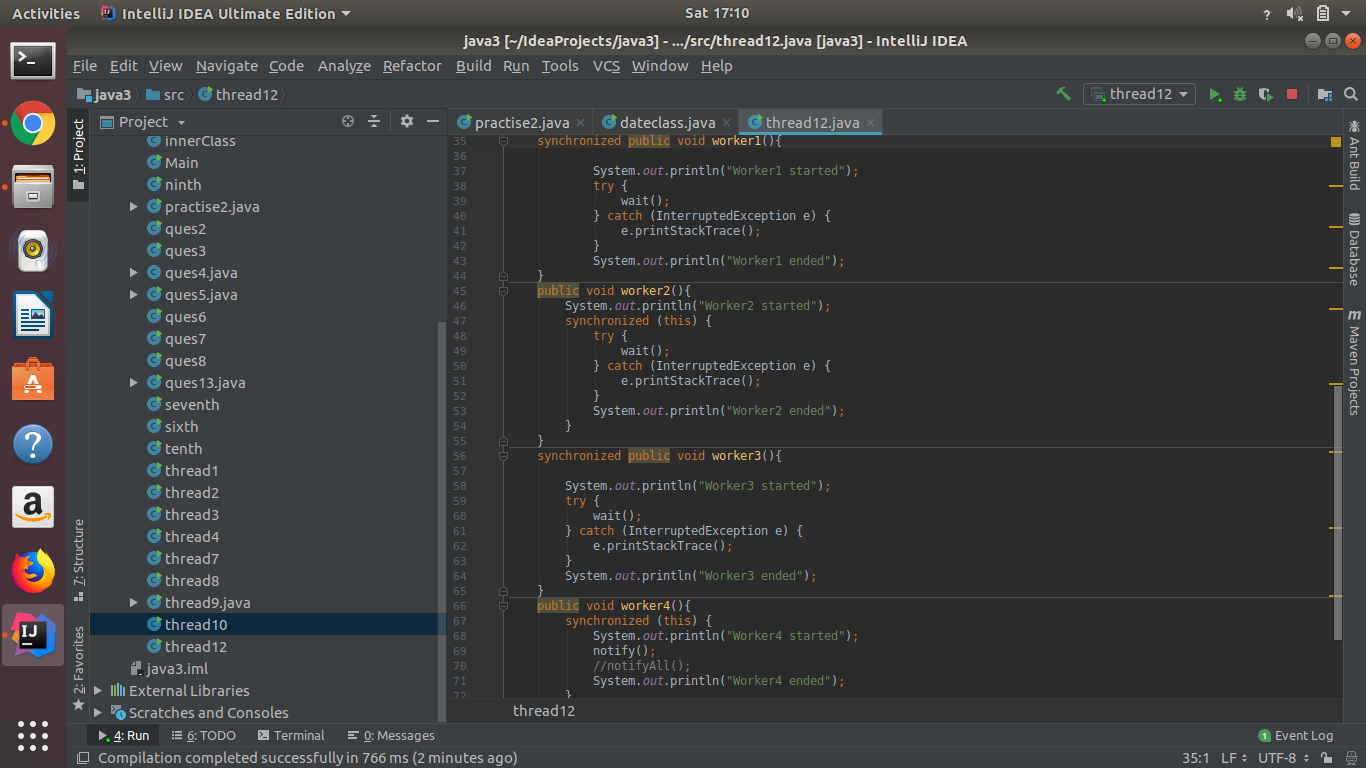
****

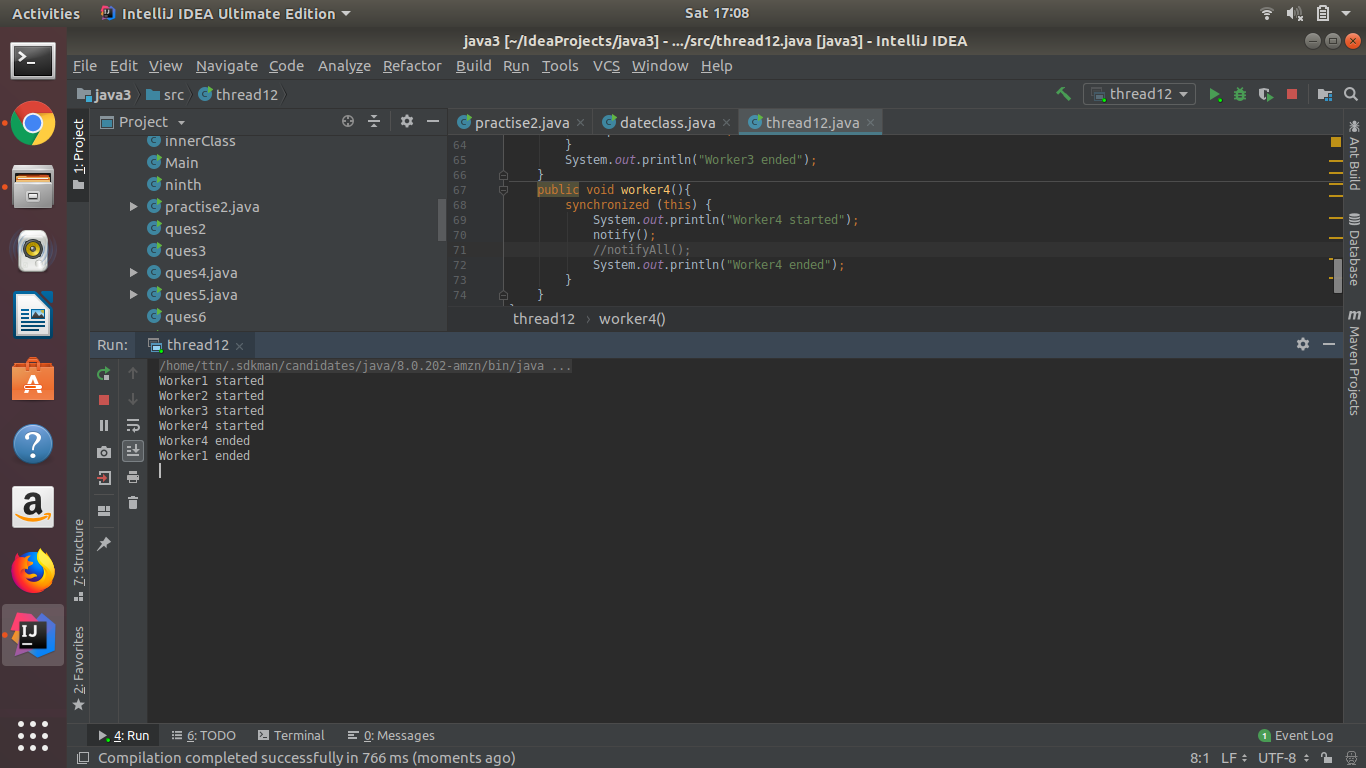
**13.Coordinate 2 threads using wait() and notifyAll().**

****

**14.Coordinate mulitple threads using wait() and notify()**

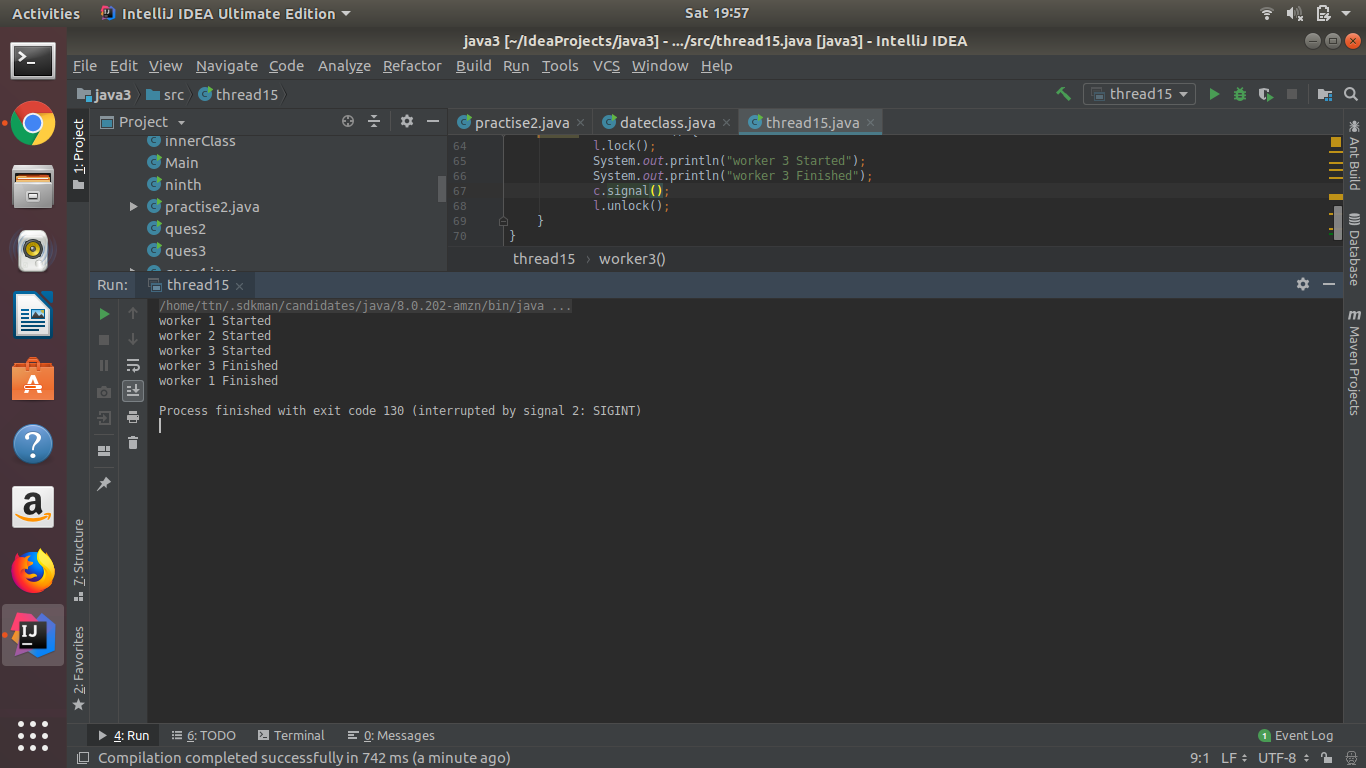
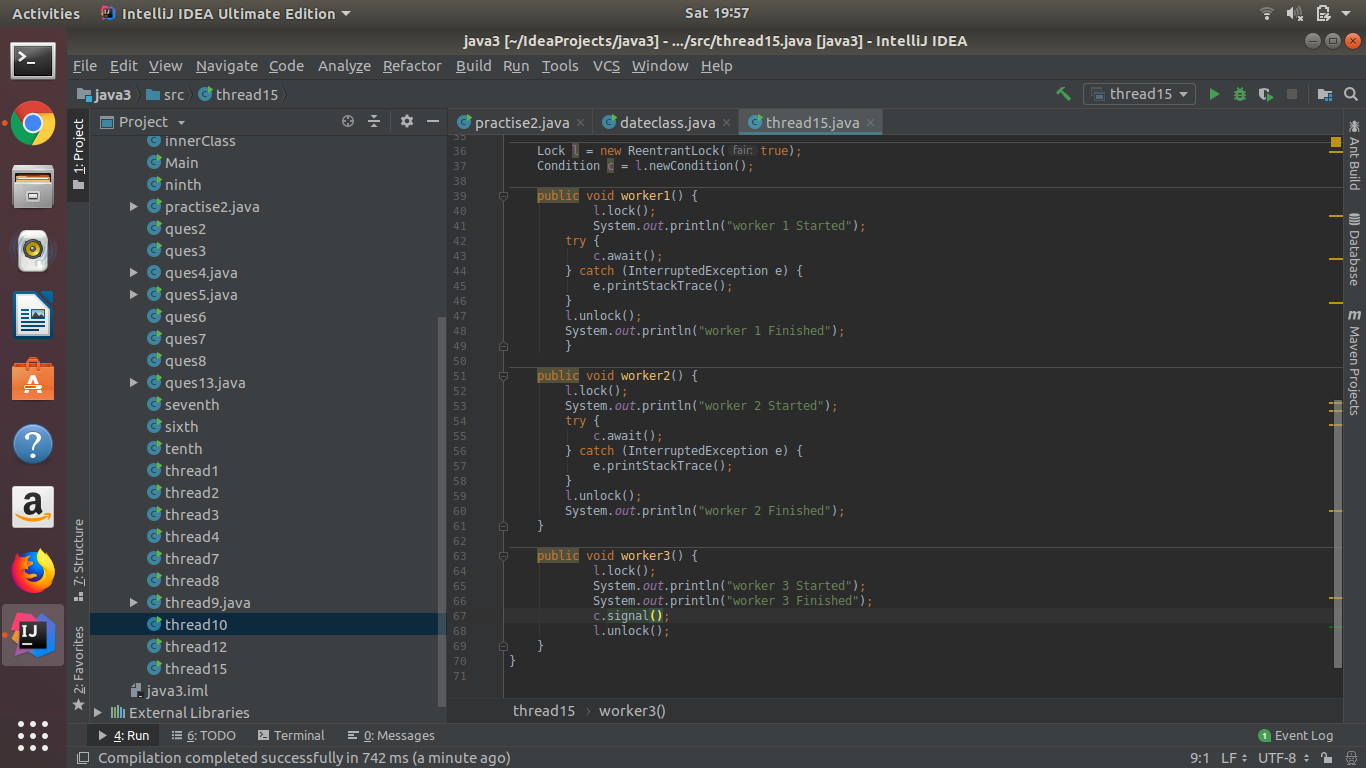
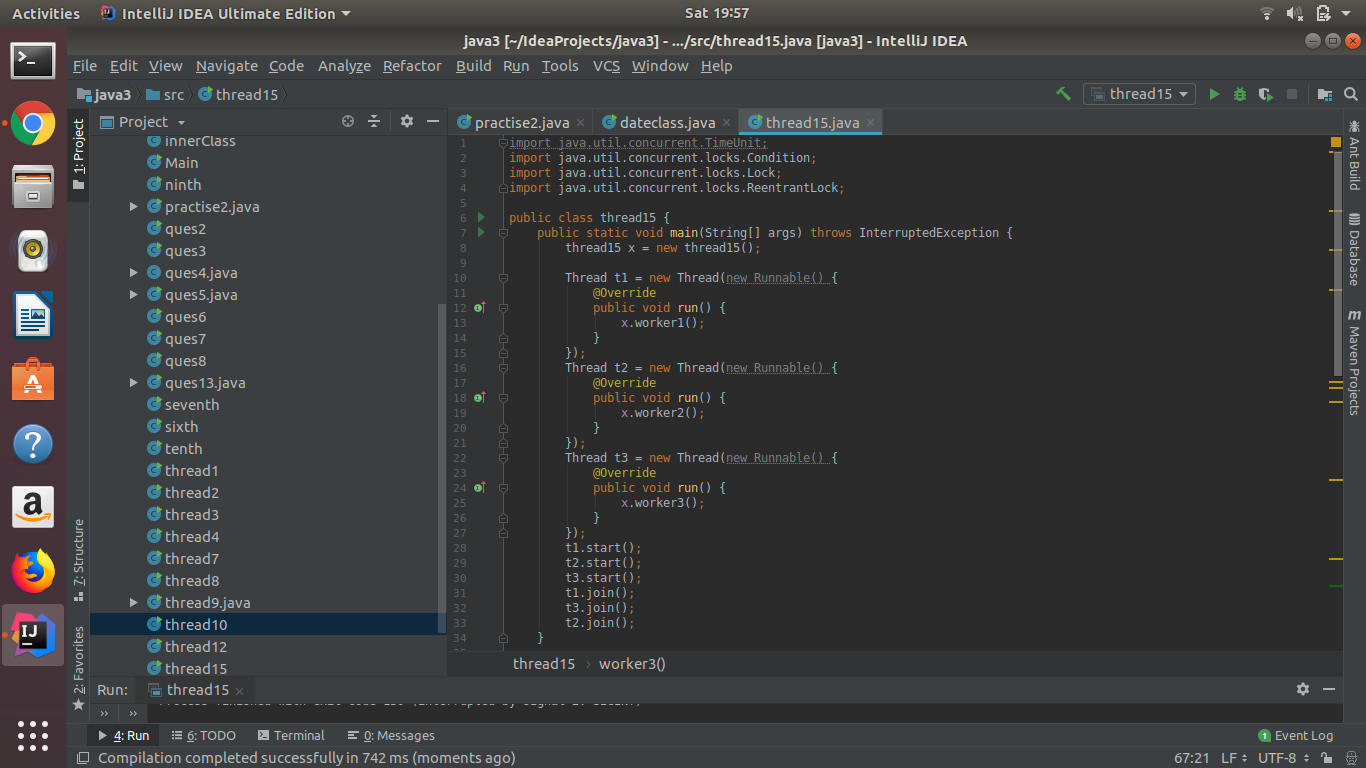
****

****

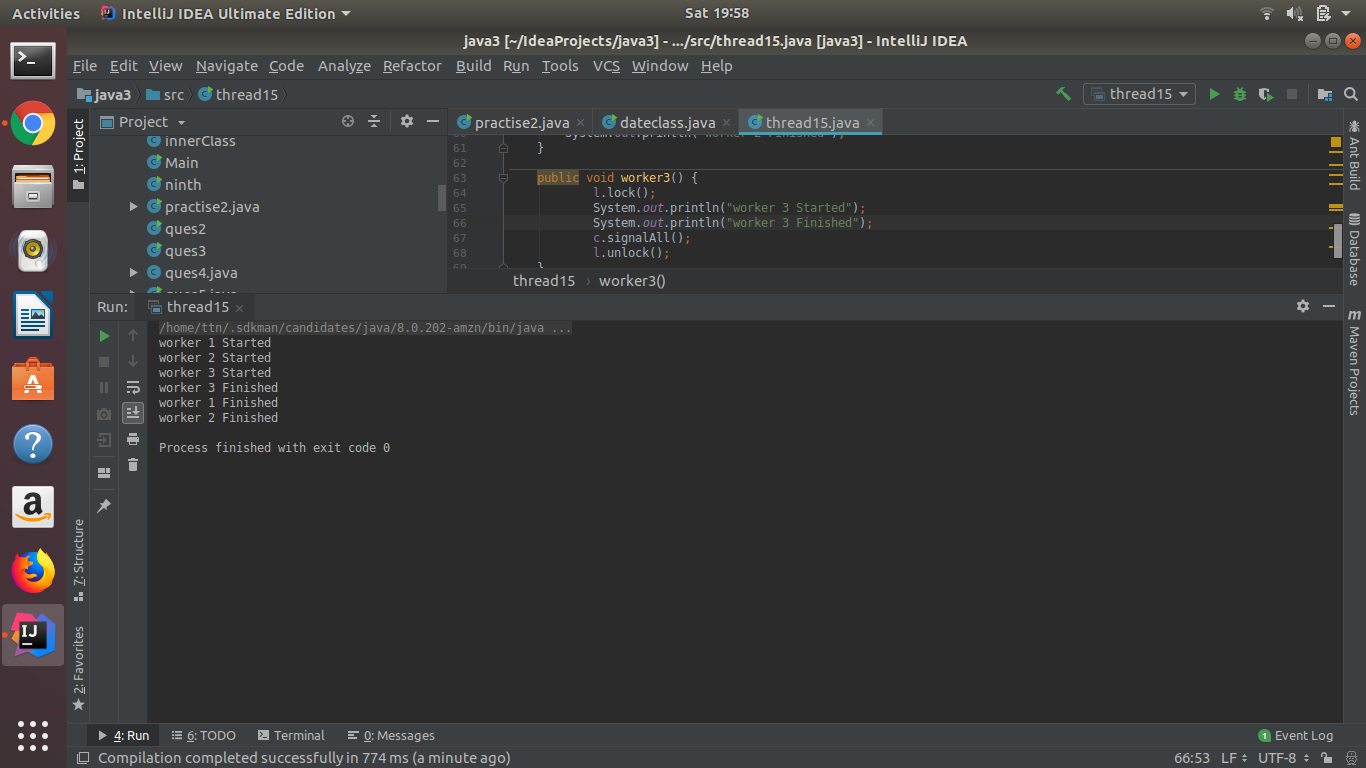
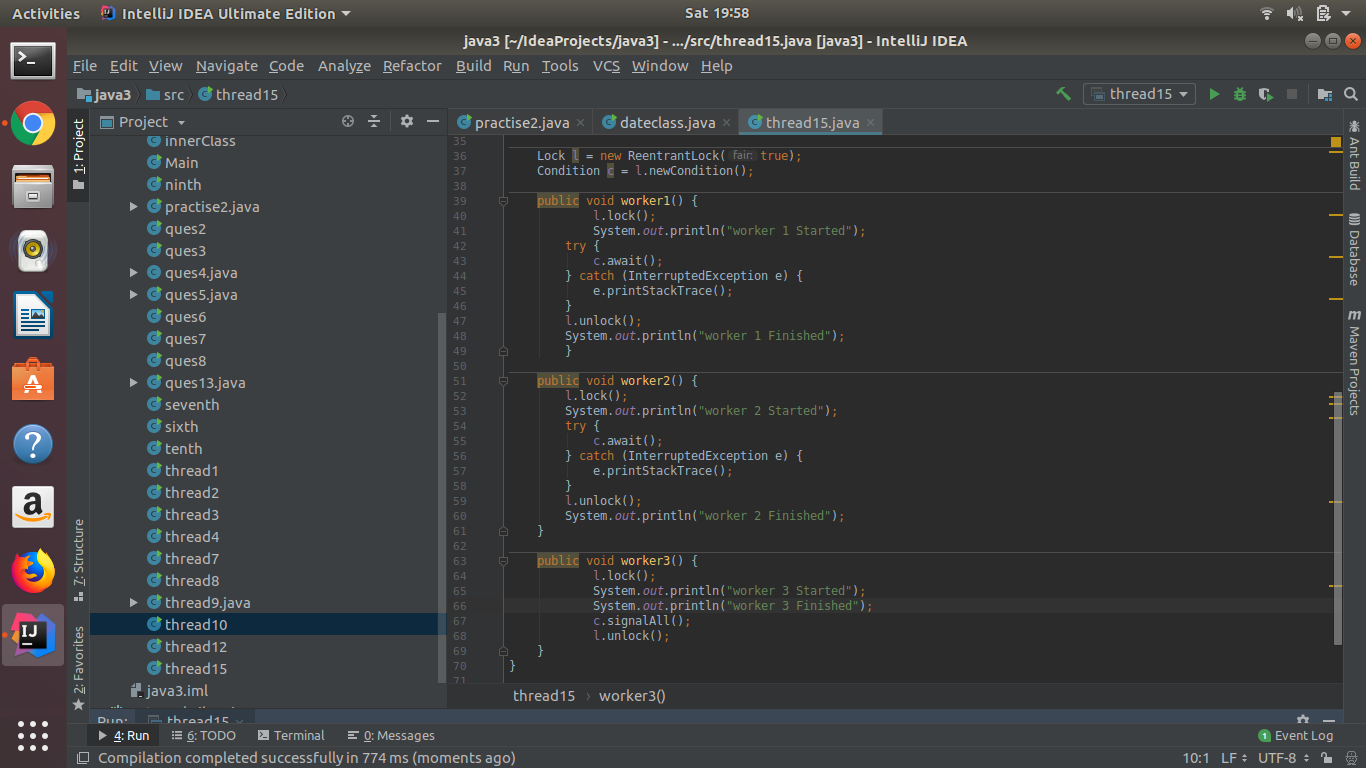
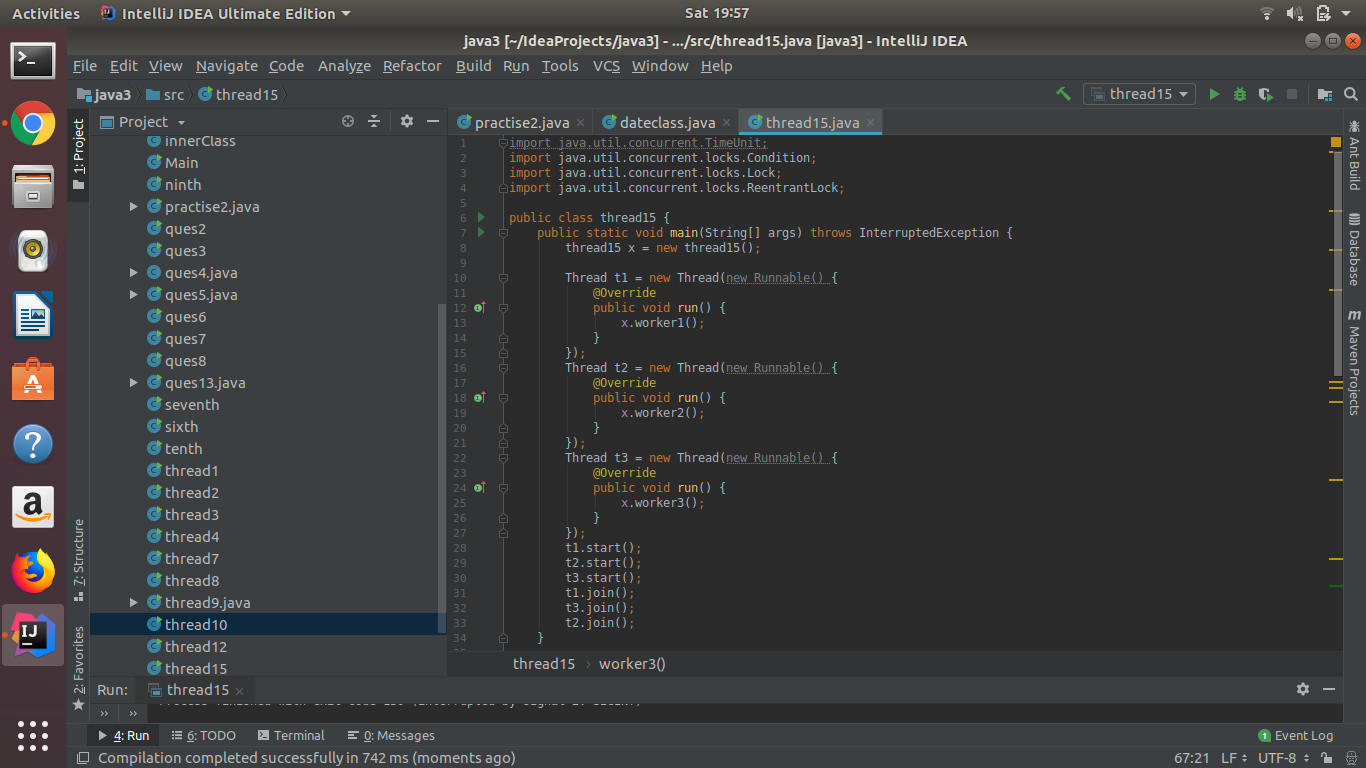
****

**15.Use Reentract lock for coordinating 2 threads with signal(), signalAll() and wait().**

**signal() and wait()**

****

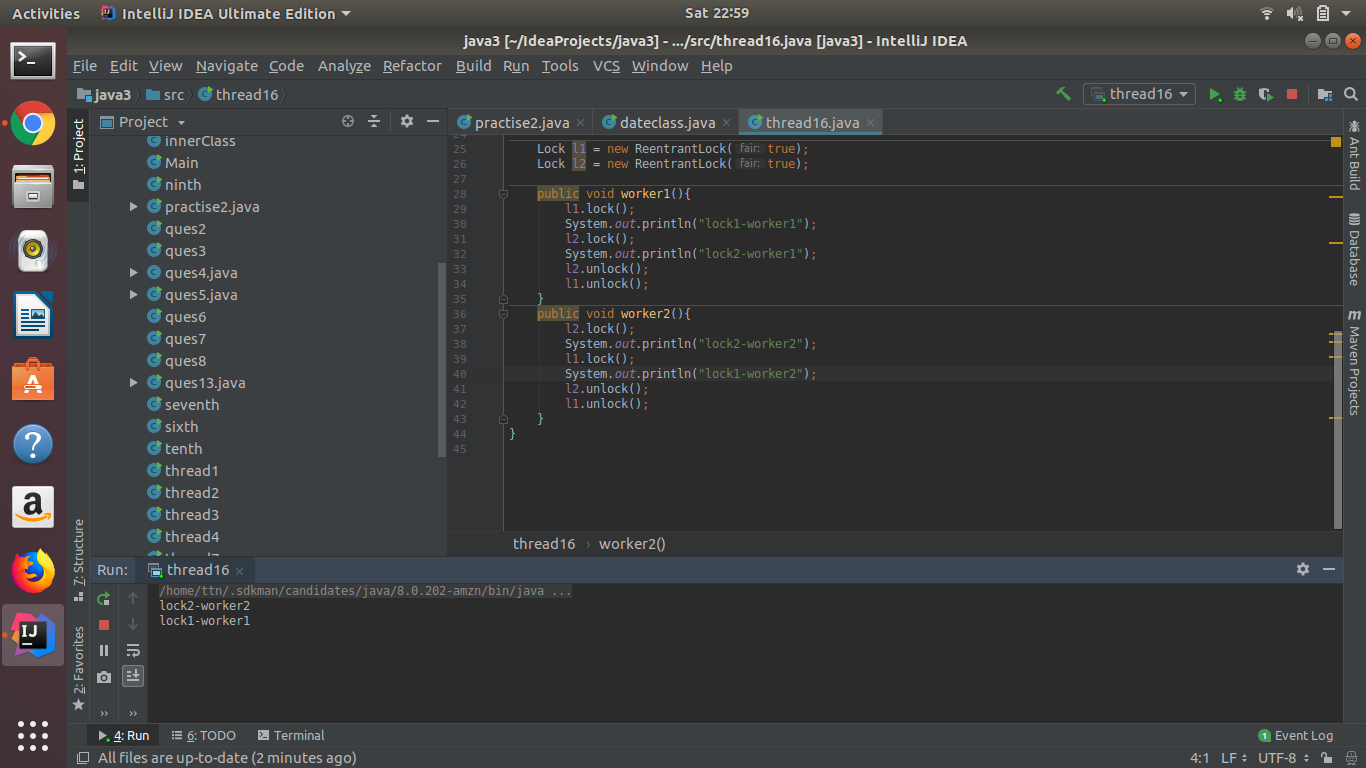
**signalAll() and wait()**

****

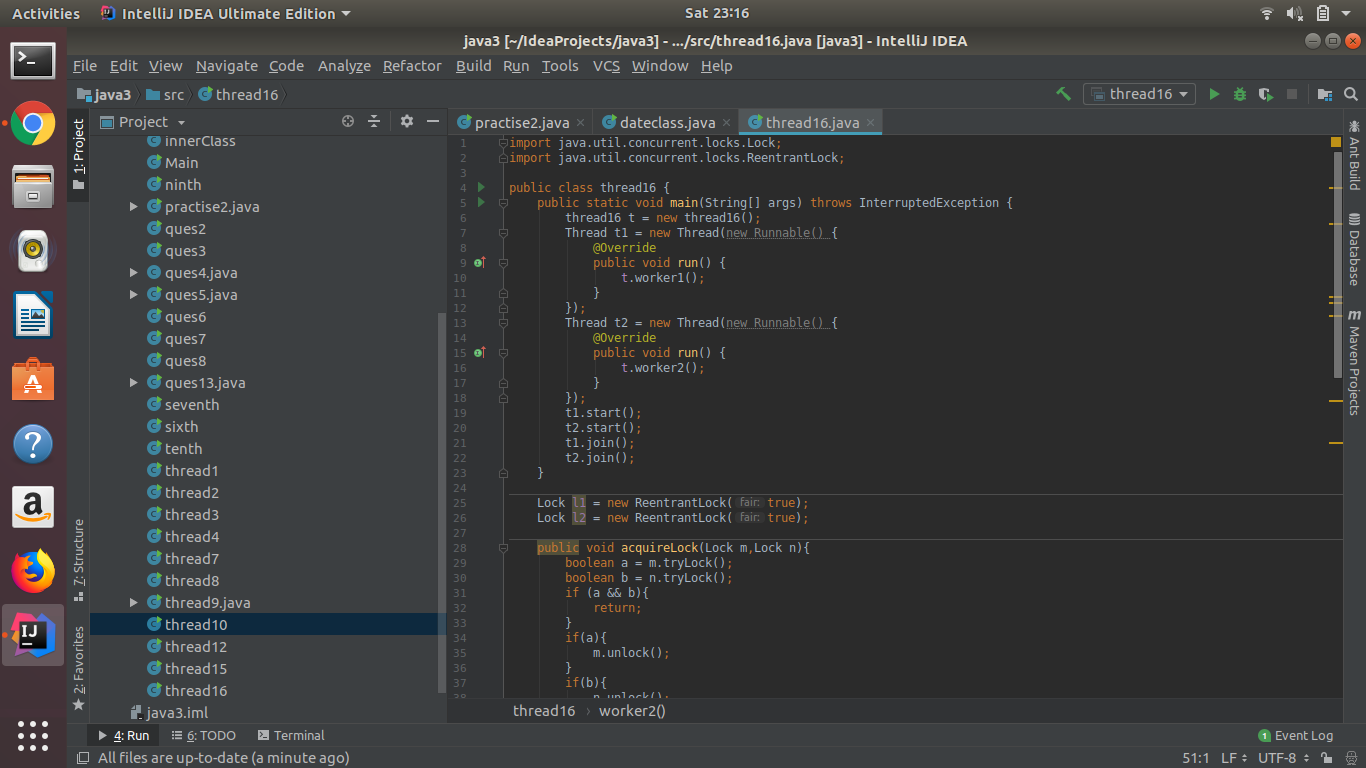
**16.Create a deadlock and Resolve it using tryLock().**

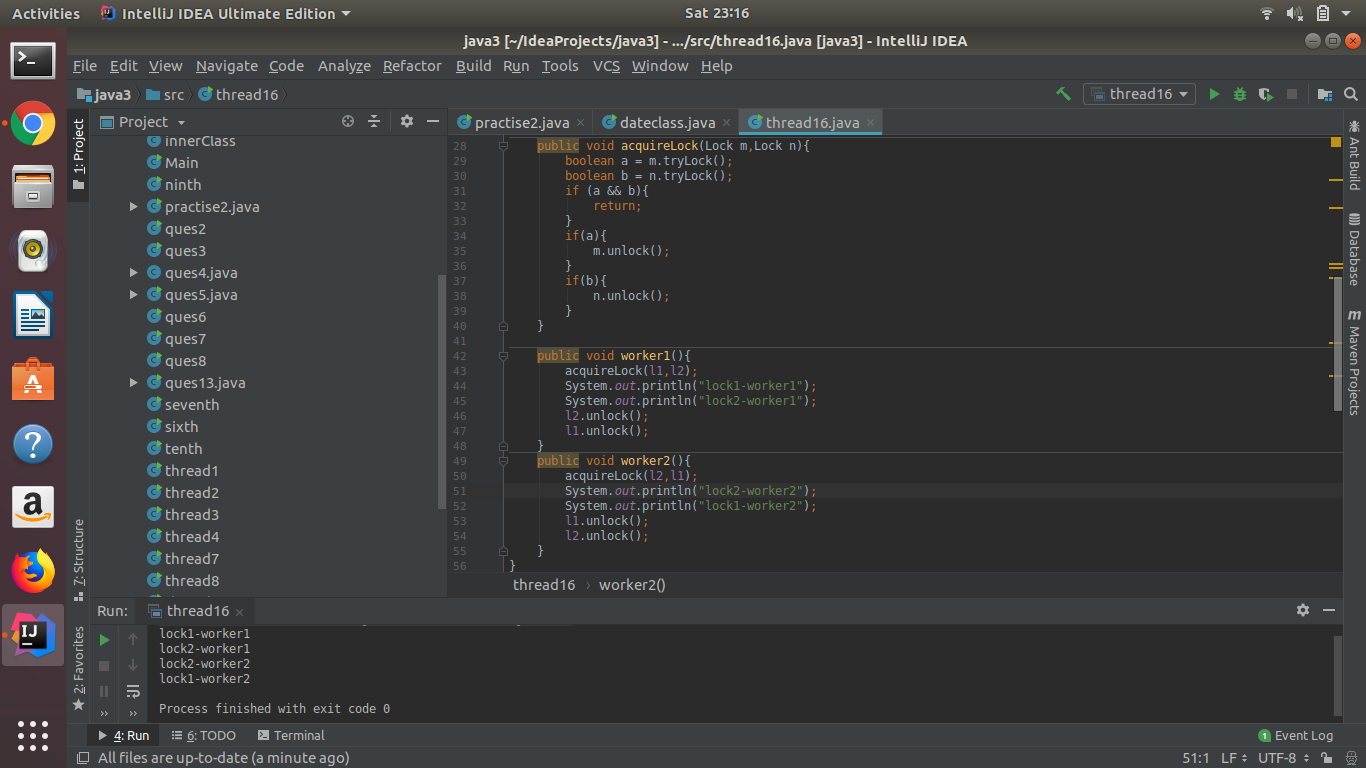
**Deadlock ->**

****

****

**Solution using tryLock()**

****

****