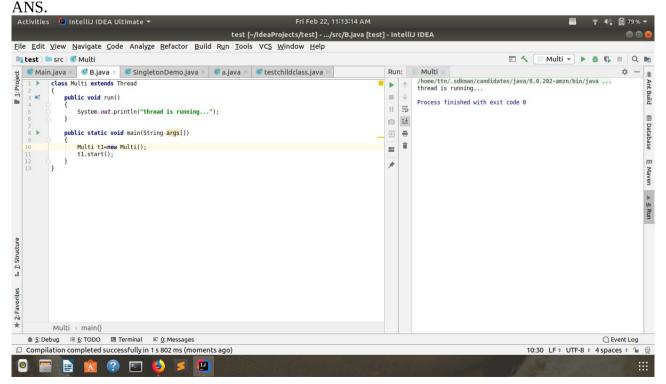
MULTITHREADING

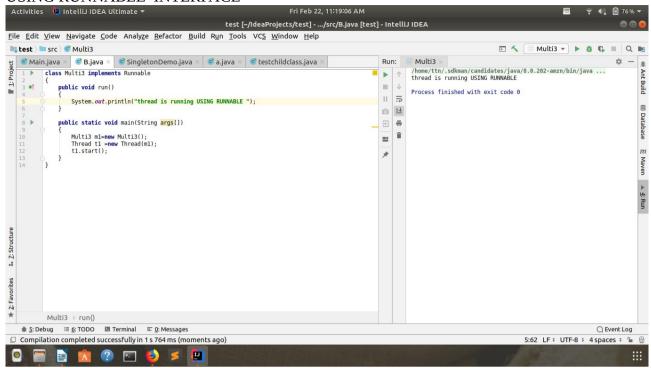
- 1. Create and Run a Thread using Runnable Interface and Thread class.
- 2. Use sleep and join methods with thread.
- 3. Use a singleThreadExecutor to submit multiple threads.
- 4. Try shutdown() and shutdownNow() and observe the difference.
- 5. Use isShutDown() and isTerminate() with ExecutorService.
- 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().
- 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 notify().
- 14. Coordinate mulitple threads using wait() and notifyAll()
- 15. Use Reentract lock for coordinating 2 threads with signal(), signalAll() and wait().
- 16. Create a deadlock and Resolve it using tryLock().

ANSWERS

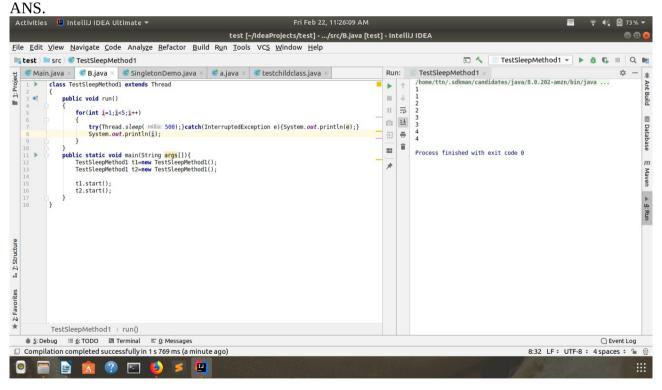
Q1]Create and Run a Thread using Runnable Interface and Thread class.



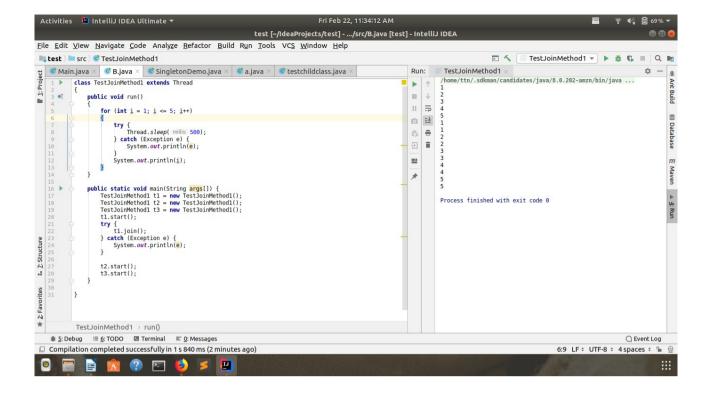
USING RUNNABLE INTERFACE



Q2.Use sleep and join methods with thread.



Using join



Q3.Use a singleThreadExecutor to submit multiple threads.

```
Activities 🚇 IntelliJ IDEA Ultimate 🔻
                                                             test [~/ideaProjects/test] - .../src/multi.java [test] - IntelliJ IDEA
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
SingleExecutorServiceDemo

Main.java & B.java & multi.java ×

in import java.util.concurrent.ExecutorServiceDemo

in import java.util.concurrent.ExecutorServiceDemo
                                                                                                                       SingleExecutorServiceDemo ×
                                                                                                              Run:
                                                                                                                      /home/ttn/sdkman/candidates/java/8.0.202-amzn/bin/java ...
pool-1-thread-1 performing Job-1
pool-1-thread-1 performing Job-2
wait for 2 sec
                                                                                                                                                                                         Ant Build
           import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
                                                                                                              .
                                                                                                              11 5
                                                                                                                      pool-1-thread-1 performing Job-3
           class SingleExecutorServiceDemo
                                                                                                             ⊙ <u>=</u>↓
               public static void main(String[] args)
                                                                                                              ň
                                                                                                                  ē
                                                                                                                  ŧ
                   ExecutorService executorService = Executors.newSingleThreadExecutor();
                                                                                                              ==
                                                                                                                                                                                         ≅ Maven
                   executorService.submit(new Runnable() {
                       @Override
public void run() {
    System.out.println(Thread.currentThread().getName()+" performing Job-1");
                   3).
                   executorService.submit(new Runnable() {
                       coverride
public void run() {
    System.out.println(Thread.currentThread().getName()+" performing Job-2");
                   executorService.submit(new Runnable() {
                       public void run() {
                              System.out.println("wait for 2 sec\n");
Thread.sleep( millis: 3000);
            SingleExecutorServiceDemo > main()
    ≡ <u>6</u>: TODO ■ Terminal ≡ <u>0</u>: Messages
 ☐ Compilation completed successfully in 4 s 910 ms (a minute ago)
                                                                                                                                                       42:12 LF + UTF-8 + 4 spaces + 🐿
             📔 🛕 🕐 🖭 😉 🗲 📳
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
class SingleExecutorServiceDemo
         public static void main(String[] args)
                   ExecutorService executorService = Executors.newSingleThreadExecutor();
                   executorService.submit(new Runnable() {
                            @Override
                            public void run() {
```

```
System.out.println(Thread.currentThread().getName()+" performing Job-
1");
         });
         executorService.submit(new Runnable() {
             @Override
             public void run() {
                  System.out.println(Thread.currentThread().getName()+" performing Job-
2");
             }
         });
         executorService.submit(new Runnable() {
             @Override
             public void run() {
                  try
                  {
                       System.out.println("wait for 2 sec\n");
                       Thread.sleep(3000);
                  }
                  catch (Exception e)
                  {
                       e.printStackTrace();
                  System.out.println(Thread.currentThread().getName()+" performing Job-
3");
             }
         });
         executorService.shutdown();
    }
}
```

Q4.Try shutdown() and shutdownNow() and observe the difference.

```
ANS.
     Activities ■ IntelliJ IDEA Ultimate ▼
                                                                                                                                                                                                                                                                                                                              Fri Mar 1, 9:26:53 AM
                                                                                                                                                                                                                                           test [~/IdeaProjects/test] - .../src/multi.java [test] - IntelliJ IDEA
  File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

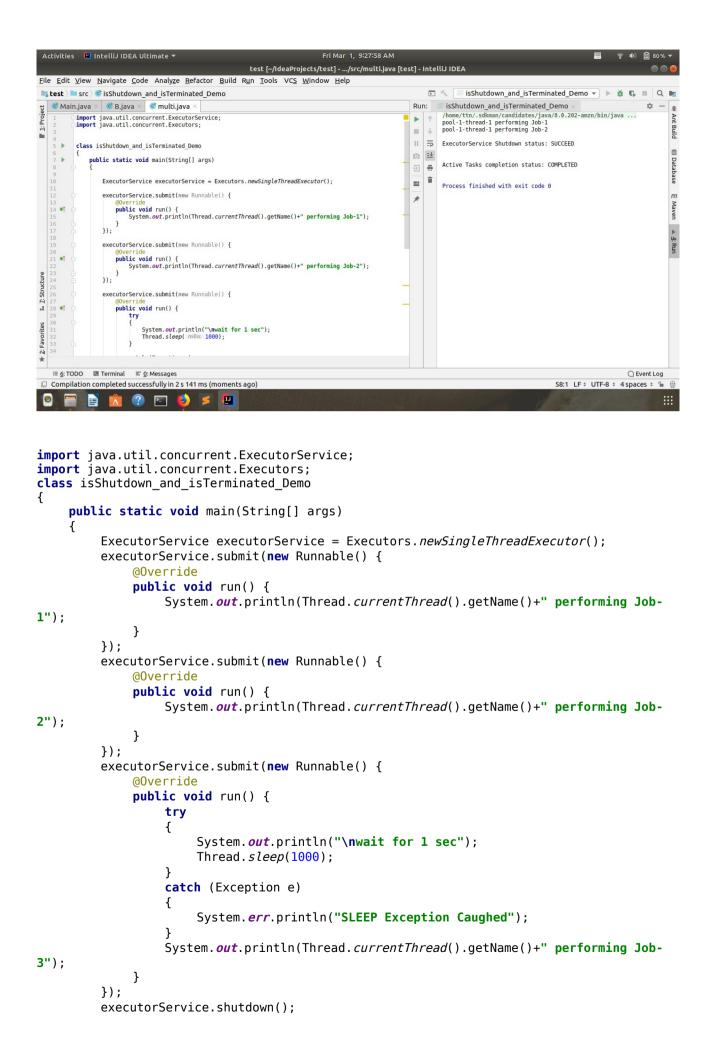
ightharpoonup test 
angle \equiv {
m src} 
angle 	ext{ @ Shutdown_and_Shutdown_Demo}
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            Shutdown_and_Shutdown_Demo ▼ ▶ 🍎 🖏
             Shutdown_and_Shutdown_Demo
                                                                                                                                                                                                                                                                                                                                                                                                                                                                   And thread threa
                                           import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    Ant Build
                                                                                                                                                                                                                                                                                                                                                                                                                                   || = Process finished with exit code 0
                                           class Shutdown_and_Shutdown_Demo
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    ())) Database

    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
   □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □
                                                          public static void main(String[] args)
                                                                                                                                                                                                                                                                                                                                                                                                                                                  ē
                                                                                                                                                                                                                                                                                                                                                                                                                                                   î
                                                                        ExecutorService executorService = Executors.newSingleThreadExecutor();
                                                                                                                                                                                                                                                                                                                                                                                                                                   ==
                                                                         executorService.submit(new Runnable() {
                                                                                                                                                                                                                                                                                                                                                                                                                                     *
                                                                                       goverrise
public void run() {
   System.out.print().fthread.currentThread().getName()+" performing Job-1");
   if(Thread.currentThread().isInterrupted()==true)
   System.out.println("Inside Job-1: interrupted by shutdownNow()");
                                                                         executorService.submit(new Runnable() {
                                                                                       @Override
public void run() {
    System.out.println(Thread.currentThread().getName()+" performing Job-2");
    if(Thread.currentThread().isInterrupted()==true)
    System.out.println("Inside Job-2: interrupted by shutdownNow()");
  . Z: Structure
  ¥ 2: Favorites
                                                                        executorService.submit(new Runnable() {
                                                                                       public void run() {
                                             Shutdown\_and\_Shutdown\_Demo \rightarrow main() \rightarrow new Runnable \rightarrow run()
                ≡ 6: TODO ■ Terminal ≡ 0: Messages
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    ☐ Event Log
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               23:32 LF + UTF-8 + 4 spaces + 1
   ☐ Compilation completed successfully in 2 s 48 ms (moments ago)
                                                    📔 🙍 🕜 🖭 😉 🗲 🛄
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
class Shutdown and Shutdown Demo
```

```
public static void main(String[] args)
    {
         ExecutorService executorService = Executors.newSingleThreadExecutor();
         executorService.submit(new Runnable() {
             @Override
             public void run() {
                  System.out.println(Thread.currentThread().getName()+" performing Job-
1");
                  if(Thread.currentThread().isInterrupted()==true)
                      System.out.println("Inside Job-1: interupted by shutdownNow()");
             }
         });
         executorService.submit(new Runnable() {
             @Override
             public void run() {
                  System.out.println(Thread.currentThread().getName()+" performing Job-
2");
                  if(Thread.currentThread().isInterrupted()==true)
                      System.out.println("Inside Job-2: interupted by shutdownNow()");
             }
         });
         executorService.submit(new Runnable() {
             @Override
             public void run() {
                  try
                  {
                      System.out.println("wait for 1 sec\n");
                      Thread.sleep(1000);
                  }
                  catch (Exception e)
                  {
                      e.printStackTrace();
                  System.out.println(Thread.currentThread().getName()+" performing Job-
3");
                  if(Thread.currentThread().isInterrupted()==true)
                      System.out.println("Inside Job-3: interupted by shutdownNow()");
             }
         });
         executorService.shutdownNow();
         executorService.shutdown();
    }
}
```

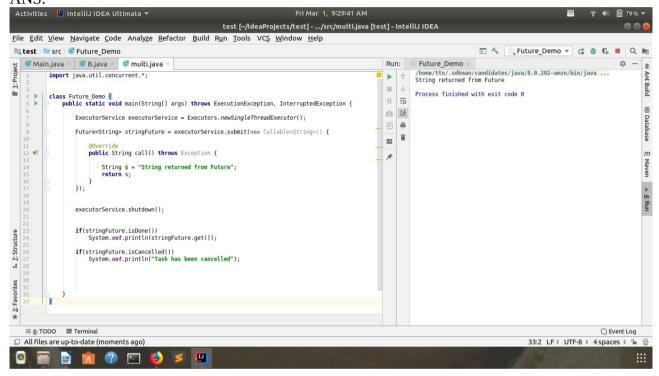
Q5.Use isShutDown() and isTerminate() with ExecutorService.

ANS.

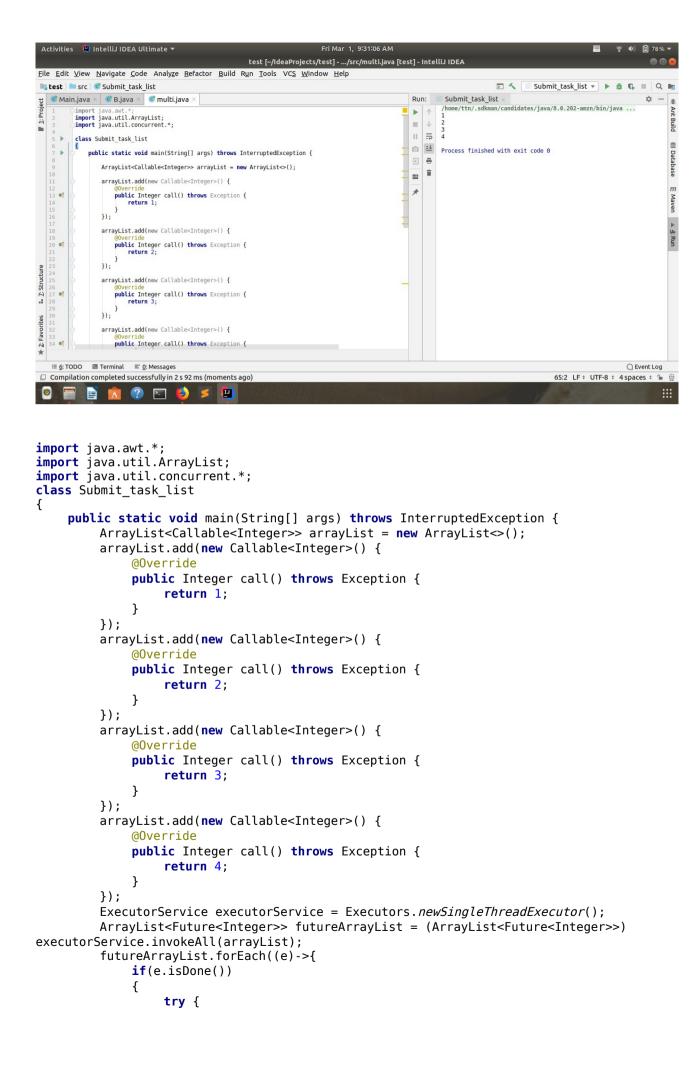


Q6.Return a Future from ExecutorService by using callable and use get(), isDone(), isCancelled() with the Future object to know the status of task submitted.

ANS.

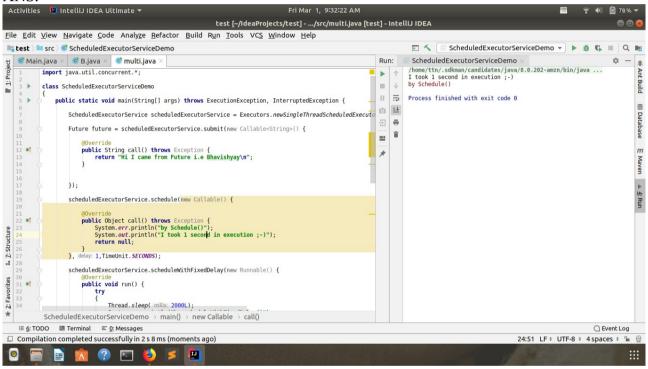


Q7.Submit List of tasks to ExecutorService and wait for the completion of all the tasks. ANS.



```
System.out.println(e.get());
}
catch (ExecutionException el)
{
        el.printStackTrace();
}
catch (InterruptedException el)
{
        el.printStackTrace();
}
}
};
executorService.shutdown();
}
```

Q8.Schedule task using schedule(), scheduleAtFixedRate() and scheduleAtFixedDelay() ANS.



```
import java.util.concurrent.*;
class ScheduledExecutorServiceDemo
    public static void main(String[] args) throws ExecutionException,
InterruptedException {
         ScheduledExecutorService scheduledExecutorService =
Executors.newSingleThreadScheduledExecutor();
         Future future = scheduledExecutorService.submit(new Callable<String>() {
             @Override
             public String call() throws Exception {
                  return "Hi I came from Future i.e Bhavishyay\n";
        });
         scheduledExecutorService.schedule(new Callable() {
             @Override
             public Object call() throws Exception {
                  System.err.println("by Schedule()");
                  System.out.println("I took 1 second in execution ;-)");
                  return null;
```

```
},1,TimeUnit.SECONDS);
         scheduledExecutorService.scheduleWithFixedDelay(new Runnable() {
             @Override
             public void run() {
                  try
                      Thread. sleep(2000L);
                      System.err.println("by scheduleWithFixedDelay()");
                      System.out.println("ScheduleWithFixedDelay Scheduled Task to
executed after fixed interval\n");
                  }
                  catch (InterruptedException e)
                      e.printStackTrace();
                  }
         },0,1,TimeUnit.SECONDS);
         scheduledExecutorService.scheduleAtFixedRate(new Runnable() {
                                                                @Override
                                                                 public void run() {
                                                                     try {
Thread. sleep(2000L);
System.err.println("by scheduleWithFixedRate()");
System.out.println("ScheduleAtFixedRate Scheduled Task to executed after fixed
interval");
                                                                     } catch
(InterruptedException e) {
e.printStackTrace();
                                                                     }
                                                                }
                                                            }
                  Θ,
                  1,
                  TimeUnit.SECONDS);
         if(future.isDone())
             System.out.println(future.get());
         scheduledExecutorService.shutdown();
    }
}
Q9.Increase concurrency with Thread pools using newCachedThreadPool() and
newFixedThreadPool().
```

ANS.

```
Activities 🙎 IntelliJ IDEA Ultimate 🤻
                                                    test [~/IdeaProjects/test] - .../src/multi.java [test] - IntelliJ IDEA
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
multi.java

Main.java 

B.java 

multi.java

Minuti.java

import java.util.concurrent.Executorse

import java.util.concurrent.Executorse
                                                                                            Run: Demo1 x
                                                                                                   /home/ttn/.sdkman/candidates/java/8.0.202-amzn/bin/java ...
pool-2-thread-1
         import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
                                                                                           •
                                                                                                   pool-2-thread-2
          class Thread1 implements Runnable
                                                                                             Access can be private more... (Ctrl+F1)
                                                                                                   pool-2-thread-1
             int id;
                                                                                            □ =+
                                                                                                   pool-2-thread-2
             public Thread1(int i)
                                                                                                =
                                                                                                   pool-2-thread-1
                this.id =i;
                                                                                                   pool-2-thread-2
                                                                                            ==
                                                                                                   Process finished with exit code 130 (interrupted by signal 2: SIGINT)
             @Override
public void run() {
    System.out.println(Thread.currentThread().getName()+"\n");
    try
    {
                   Thread.sleep( millis: 1000L):
                 catch (Exception e)
                   e.printStackTrace();
. Z: Structure
          class Demol
             public static void main(String[] args) {
                for (int \underline{i} = \theta; \underline{i} \leftarrow 5; \underline{i} \leftrightarrow \emptyset {
    ≣ <u>6</u>: TODO ■ Terminal ≡ <u>0</u>: Messages
                                                                                                                                                 ☐ Event Log
                                                                                                                                40:2 LF ÷ UTF-8 ÷ 4 spaces ÷ 1 €
 ☐ Compilation completed successfully in 1 s 949 ms (moments ago)
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
class Thread1 implements Runnable
        int id;
        public Thread1(int i)
        {
                this.id =i;
        }
        @Override
        public void run() {
                System.out.println(Thread.currentThread().getName()+"\n");
                {
                        Thread.sleep(1000L);
                }
                catch (Exception e)
                        e.printStackTrace();
                }
        }
class Demo1
        public static void main(String[] args) {
                ExecutorService executorService = Executors.newCachedThreadPool();
                ExecutorService executorService1 = Executors.newFixedThreadPool(2);
                for (int i = 0; i <=5; i++) {
                        executorService1.submit(new Thread1(i));
                executorService.shutdown();
        }
}
```

Q10.Use Synchronize method to enable synchronization between multiple threads trying to access method at same time.
ANS.

```
Activities 🚇 IntelliJ IDEA Ultimate 🔻
                                                                                                                                            test [~/ideaProjects/test] - .../src/multi.java [test] - IntelliJ IDEA
Analyze Refared Main.java × B.java × multi.java ×

| Compared Comp
 File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
                                                                                                                                                                                                                                                                            Driver ×
/home/ttn/, sdkman/candidates/java/8.0.202-amzn/bin/java ...
by Thread-0 5
By Thread-0 15
By Thread-0 15
By Thread-0 25
By Thread-0 25
By Thread-0 25
By Thread-0 30
By Thread-0 30
By Thread-0 30
By Thread-0 40
By Thread-0 45
By Thread-0 46
By Thread-1 12
By Thread-1 12
By Thread-1 13
By Thread-1 24
By Thread-1 24
By Thread-1 30
By Thread-1 30
By Thread-1 42
By Thread-1 36
By Thread-1 48
By Thread-1 56
By Thread-1 56
By Thread-1 68
                                                                                                                                                                                                                                                           Run: Driver x
                                                                                                                                                                                                                                                        •
                                                                                                                                                                                                                                                            II ≒
                                            for(int <u>i</u>=1;<u>i</u><=10;<u>i</u>++)
                                                                                                                                                                                                                                                          System.out.println("By "+Thread.currentThread().getName()+" "+x*i);
                                                                                                                                                                                                                                                          ∄ =
                                                                                                                                                                                                                                                          ==
                           class Job1 extends Thread
                                   Table t;
Job1 (Table table)
                                           this.t = table:
                                   public void run()
                                                                                                                                                                                                                                                                              Process finished with exit code 0
  . Z: Structure
                          class Job2 extends Thread
                                    Table <mark>t</mark>;
Job2 (Table table)
                                   public void run()
           ≡ <u>6</u>: TODO     Terminal   ≡ <u>0</u>: Messages
                                                                                                                                                                                                                                                                                                                                                                                                         C Event Log
                                                                                                                                                                                                                                                                                                                                                         ☐ Compilation completed successfully in 2 s 30 ms (moments ago)
 class Table
                      synchronized void print(int x)
                                           for(int i=1;i<=10;i++)</pre>
                                           {
                                                                 System.out.println("By "+Thread.currentThread().getName()+" "+x*i);
                                           }
                      }
class Job1 extends Thread
                     Table t;
                     Job1 (Table table)
                                           this.t = table;
                     }
                     public void run()
                                           t.print(5);
 class Job2 extends Thread
                     Table t;
                     Job2 (Table table)
                       {
                                           this.t = table;
                     }
                     public void run()
                                           t.print(6);
                      }
class Driver
                     public static void main(String[] args) {
                                           Table t = new Table();
                                           Thread j1 = new Job1(t);
```

```
Thread j2 = new Job2(t);
    //j1.start();
    j2.start();
    j1.start();
}
```

Q11.Use Synchronize block to enable synchronization between multiple threads trying to access method at same time.

ANS.

```
Fri Mar 1, 9:36:29 AM
 Activities 🚇 IntelliJ IDEA Ultimate
                                                                         test [~/IdeaProjects/test] - .../src/multi.java [test] - IntelliJ IDEA
 <u>F</u>ile <u>E</u>dit <u>V</u>iew <u>N</u>avigate <u>C</u>ode Analy<u>z</u>e <u>R</u>efactor <u>B</u>uild <u>Run <u>T</u>ools VC<u>S <u>W</u>indow <u>H</u>elp</u></u>
                                                                                                                                                                           Driver1 ▼ ▶ # C ■
 test > src > multi.java
                                                                                                                                           Driver1 ×

/home/ttn/.sdkman/candidates/java/8.0.202-amzn/bin/java ...

By Thread-0 10

By Thread-0 15

By Thread-0 25

By Thread-0 25

By Thread-0 30

By Thread-0 30

By Thread-0 40

By Thread-0 45

By Thread-0 46

By Thread-1 12

By Thread-1 12

By Thread-1 18

By Thread-1 30

By Thread-1 30

By Thread-1 18

By Thread-1 36

By Thread-1 36

By Thread-1 48

By Thread-1 56

By Thread-1 69

Process finished with evit code 20

Process finished with evit code 20
     class Table1
                                                                                                                                                                                                                             Ant Build
                  void print(int x)
                                                                                                                                   н
                       \begin{array}{c} synchronized(this)\{\\ & for \ (int \ \underline{i} = 1; \ \underline{i} <= 10; \ \underline{i} ++) \ \{\\ & System.out.println("By" + Thread.currentThread().getName() + "" + x * \underline{i}); \\ & \end{array} 
                                                                                                                                   II 등
                                                                                                                                                                                                                            ())) Database
                                                                                                                                  ⊕ ⊕
                                                                                                                                   ==
             class Jobl1 extends Thread
                  Table1 t:
                  Job11(Table1 table1)
                      this.t = table1;
                  public void run()
                                                                                                                                             Process finished with exit code 0
 ... 7: Structure
                      t.print(5);
             class Job12 extends Thread
                  Table1 t;
                  Job12(Table1 table1)
                      this.t = table1:
     ≡ 6: TODO ■ Terminal ≡ 0: Messages
                                                                                                                                                                                                              ○ Event Log
 ☐ Compilation completed successfully in 2 s 167 ms (moments ago)
                                                                                                                                                                                    55:1 LF + UTF-8 + 4 spaces + 1
               class Table1
           void print(int x)
           {
                      synchronized(this){
                                 for (int i = 1; i <= 10; i++) {</pre>
                                            System.out.println("By " + Thread.currentThread().getName() + " " + x *
i);
                                 }
                      }
           }
class Job11 extends Thread
           Table1 t;
           Job11(Table1 table1)
           {
                      this.t = table1;
           }
           public void run()
           {
                      t.print(5);
class Job12 extends Thread
           Table1 t;
           Job12(Table1 table1)
```

```
{
         this.t = table1:
    }
    public void run()
    {
         t.print(6);
class Driver1
    public static void main(String[] args) {
         Table1 t = new Table1();
         Thread j1 = new Job11(t);
         Thread j2 = new Job12(t);
         //i1.start();
         j2.start();
         j1.start();
    }
}
```

Q12.Use Atomic Classes instead of Synchronize method and blocks.

```
■ IntelliJ IDEA Ultimate ▼
                                                                                 Fri Mar 1, 9:37:41 AM
                                                           test [~/ideaProjects/test] - .../src/multi.java [test] - IntelliJ IDEA
<u>F</u>ile <u>E</u>dit <u>V</u>iew <u>N</u>avigate <u>C</u>ode Analy<u>z</u>e <u>R</u>efactor <u>B</u>uild <u>Run <u>T</u>ools VC<u>S <u>W</u>indow <u>H</u>elp</u></u>
SynchronizeDemo

Main.java & B.java & multi.java ×

| Image: simport java.util.concurrent.atomic.At
| Class SynchronizeDemo | |
                                                                                                                                          SynchronizeDemo ▼ ▶ # G ■ Q ■
                                                                                                          Run: SynchronizeDemo ×

/home/ttn/.sdkman/candidates/java/8.0.202-amzn/bin/java ...
2000
           import java.util.concurrent.atomic.AtomicInteger;
                                                                                                        - ↑
                                                                                                                                                                                    Ant Build
                                                                                                          Process finished with exit code 0
                                                                                                          Ⅱ 등
              AtomicInteger count= new AtomicInteger():
                                                                                                          o ±
               public void incrementCount()
                                                                                                          Ð =
                  count.incrementAndGet();
                                                                                                               -
   10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
                                                                                                          ==
              public void worker1() {
    for (int i = 1; i <= 1000; i++) {
        count.incrementAndGet();
}</pre>
                                                                                                           *
              . Z: Structure
               public static void main(String[] args) throws InterruptedException {
                  SynchronizeDemo synchronizeDemo = new SynchronizeDemo();
Thread thread1 = new Thread(new Runnable() {
                      public void run() {
    synchronizeDemo.worker1();
}
                   });
Thread thread2 = new Thread(new Runnable() {
    : <u>6</u>: TODO ■ Terminal = <u>0</u>: Messages
                                                                                                                                                                       ☐ Event Log
 ☐ Compilation completed successfully in 1 s 986 ms (moments ago)
                                                                                                                                                    46:2 LF ‡ UTF-8 ‡ 4 spaces ‡ 1 =
              📔 🙇 🕜 🖭 🔮 🥦
import java.util.concurrent.atomic.AtomicInteger;
class SynchronizeDemo {
         AtomicInteger count= new AtomicInteger();
         public void incrementCount()
         {
                  count.incrementAndGet();
         public void worker1() {
                  for (int i = 1; i \le 1000; i++) {
                           count.incrementAndGet();
         }
         public void worker2() {
                  for (int i = 1; i \le 1000; i++) {
                           count.incrementAndGet();
                  }
```

```
}
    public static void main(String[] args) throws InterruptedException {
         SynchronizeDemo synchronizeDemo = new SynchronizeDemo();
         Thread thread1 = new Thread(new Runnable() {
             public void run() {
                  synchronizeDemo.worker1();
         });
         Thread thread2 = new Thread(new Runnable() {
             public void run() {
                  synchronizeDemo.worker2();
             }
         });
         thread1.start();
         thread2.start();
         thread1.join();
         thread2.join();
         System.out.println(synchronizeDemo.count);
    }
}
```

Q13.Coordinate 2 threads using wait() and notify().

```
ANS.
  Activities ☐ IntelliJ IDEA Ultimate ▼
                                                              test [~/IdeaProjects/test] - .../src/multi.java [test] - IntelliJ IDEA
<u>F</u>ile <u>E</u>dit <u>V</u>iew <u>N</u>avigate <u>C</u>ode Analy<u>z</u>e <u>R</u>efactor <u>B</u>uild <u>Run <u>T</u>ools VC<u>S <u>W</u>indow <u>H</u>elp</u></u>
Demo2 ▼ ▶ # C₁ ■ Q ■
                                                                                                                Run:
                                                                                                                          Demo2 ×
                                                                                                                        ▶ ↑
                                                                                                                .
                                                                                                                11 5
                   System.out.println("MyThread1 started");
try
                                                                                                                ⊕ ⊕
                                                                                                                     ŧ
                    catch(InterruptedException e)
                                                                                                                ...
                      e.printStackTrace();
                                                                                                                         Process finished with exit code 0
                   System.out.println("MyThread1 Completed");
           class MyThread2 extends Thread
               public void run()
                   System.out.println("MyThread2 started");
System.out.println("MyThread2 completed");
notify();
. Z: Structure
           class Demo2
¥ 2: Favorites
               public static void main(String[] args) {
                   Thread t1 = new MyThread1();
Thread t2 = new MyThread2();
            MyThread1
    ≡ 6: TODO ☑ Terminal
                                                                                                                                                                                 ○ Event Log
                            ≡ 0: Messages
                                                                                                                                                            1:1 LF ÷ UTF-8 ÷ 4 spaces ÷ ៕
 ☐ Compilation completed successfully in 2 s 185 ms (moments ago)
                                                      I.J
                                  >_
```

```
class MyThread1 extends Thread
{
    public void run()
    {
        System.out.println("MyThread1 started");
        try
        {
            wait();
        }
        catch(InterruptedException e)
        {
            e.printStackTrace();
        }
        System.out.println("MyThread1 Completed");
}
```

```
class MyThread2 extends Thread
{
    public void run()
    {
        System.out.println("MyThread2 started");
        System.out.println("MyThread2 completed");
        notify();
    }
}
class Demo2
{
    public static void main(String[] args) {
        Thread t1 = new MyThread1();
        Thread t2 = new MyThread2();
        t1.start();
        t2.start();
    }
}
```

```
Q14.Coordinate mulitple threads using wait() and notifyAll()
ANS.
                           IntelliJ IDEA Ultimate •
                                                                                                                                  test [~/ideaProjects/test] - .../src/multi.java [test] - IntelliJ IDEA
 <u>F</u>ile <u>E</u>dit <u>V</u>iew <u>N</u>avigate <u>C</u>ode Analy<u>z</u>e <u>R</u>efactor <u>B</u>uild R<u>u</u>n <u>T</u>ools VC<u>S</u> <u>W</u>indow <u>H</u>elp
□ A □ Demo2 ▼ ▶ # C □ Q ■
                                                                                                                                                                                                                                                     Run:
                                                                                                                                                                                                                                      .
                                                                                                                                                                                                                                      System.out.println("MyThread1 started");

    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
   □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □ 
    □
                                                                                                                                                                                                                                     ⊕ ⊕
                                        catch(InterruptedException e)
                                                                                                                                                                                                                                      ==
                                            e.printStackTrace();
                                       System.out.println("MyThread1 Completed");
                                                                                                                                                                                                                                                        WyThread4 completed
Exception in thread "Thread-3" java.lang.IllegalMonitorStateExceptior
at java.lang.Object.notify(Native Method)
at MyThread4.run(multi.java:61)
                       class MyThread2 extends Thread
                               public void run()
                                                                                                                                                                                                                                                        Process finished with exit code \boldsymbol{\theta}
                                       System.out.println("MyThread2 started");
                                        catch(InterruptedException e)
                                              e.printStackTrace();
                                       System.out.println("MyThread2 Completed");
                                                                                                                                                                                                                                                                                                                                                                         ☐ Event Log
         ≡ 6: TODO ■ Terminal ≡ 0: Messages

☐ Compilation completed successfully in 1 s 820 ms (moments ago)

                              📑 🖍 ? 🖭 🐸
class MyThread1 extends Thread
                   public void run()
                                       System.out.println("MyThread1 started");
                                       try
                                       {
                                                          wait();
                                       catch(InterruptedException e)
                                       {
                                                          e.printStackTrace();
                                       System.out.println("MyThread1 Completed");
                   }
class MyThread2 extends Thread
```

```
{
    public void run()
    {
         System.out.println("MyThread2 started");
         try
         {
             wait();
         }
         catch(InterruptedException e)
         {
             e.printStackTrace();
         System.out.println("MyThread2 Completed");
    }
class MyThread3 extends Thread
    public void run()
    {
         System.out.println("MyThread3 started");
         try
         {
             wait();
         }
         catch(InterruptedException e)
         {
             e.printStackTrace();
         System.out.println("MyThread3 Completed");
    }
class MyThread4 extends Thread
    public void run()
         System.out.println("MyThread4 started");
         System.out.println("MyThread4 completed");
         notify();
    }
}
class Demo2
    public static void main(String[] args) {
         Thread t1 = new MyThread1();
         Thread t2 = new MyThread2();
         Thread t3 = new MyThread3();
         Thread t4 = new MyThread4();
         t1.start();
         t2.start();
         t3.start();
         t4.start();
    }
}
```

Q15.Use Reentract lock for coordinating 2 threads with signal(), signalAll() and wait(). ANS.

```
Activities 🚇 IntelliJ IDEA Ultimate 🤻
                                             test [~/IdeaProjects/test] - .../src/multi.java [test] - IntelliJ IDEA
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
⊫ test ⟩ ■ src ⟩ © Demo
  © Main.java × © B.java × © multi.java ×
                                                                                 Run: Demo x
                                                                                                                                         ∰ Ant Build
                                                                                /home/ttn/.sdkman/candidates/java/8.0.202-amzn/bin/java ...
worker 1 Started
worker 2 Started
worker 2 Finished

worker 1 Finished
        import java.util.concurrent.locks.Condition;
import java.util.concurrent.locks.Lock;
import java.util.concurrent.locks.ReentrantLock;
                                                                               - ↑
                                                                                                                                        ())) Database
                                                                                ☐ ➡ Process finished with exit code 0
           Lock lock = new ReentrantLock( fair: true);
Condition condition = lock.newCondition();
                                                                                 ⊕ =
                                                                                    ŧ
           public void worker1() {
                                                                                 ==
             ≅ Maven
                                                                                 *
             } finally {
lock.unlock();
. Z: Structure
           public void worker2() {
              try{
lock.lock();
             lock.lock();
System.out.println("worker 2 Started");
System.out.println("worker 2 Finished");
condition.signal();
}finally {
    lock.unlock();
   ≣ <u>6</u>: TODO ■ Terminal ≡ <u>0</u>: Messages
                                                                                                                               C Event Log
☐ Compilation completed successfully in 1 s 862 ms (moments ago)
                                                                                                                ? -
import java.util.concurrent.locks.Condition;
import java.util.concurrent.locks.Lock;
import java.util.concurrent.locks.ReentrantLock;
class Demo {
      Lock lock = new ReentrantLock(true);
      Condition condition = lock.newCondition();
      public void worker1() {
             try {
                    lock.lock();
                    System.out.println("worker 1 Started");
                    condition.await();
                    System.out.println("worker 1 Finished");
             } catch (InterruptedException e) {
                    e.printStackTrace();
             } finally {
                    lock.unlock();
      public void worker2() {
             try{
                    lock.lock();
                    System.out.println("worker 2 Started");
                    System.out.println("worker 2 Finished");
                    condition.signal();
             }finally {
                    lock.unlock();
      public static void main(String[] args) throws InterruptedException {
             Demo demo = new Demo();
             Thread thread1 = new Thread(new Runnable() {
                    @Override
                    public void run() {
                           demo.worker1();
             });
             Thread thread2 = new Thread(new Runnable() {
                    @Override
                    public void run() {
```

```
demo.worker2();
}
});
thread1.start();
thread2.start();
thread1.join();
thread2.join();
}
}
```

Q16.Create a deadlock and Resolve it using tryLock(). ANS

```
Activities 🚇 IntelliJ IDEA Ultimate 🔻
                                                                             Fri Mar 1, 9:42:27 AM
                                                         test [~/ideaProjects/test] - .../src/multi.java [test] - IntelliJ IDEA
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
 test > src > d Demo
                                                                                                                                               Demo ▼ ▶ # C
   Run:
                                                                                                             | Mome/ttn/.sdkman/candidates/java/8.0.202-amzn/bin/java...
| lock 1 worker 1 | lock 1 worker 2 |
| Exception in thread "Thread-1" java.lang.IllegalMonitorStateException | at Demo.worker?[multi_java:25] | at Demos2: run(multi_java:47) | at java.lang.Thread.run(<u>Thread.java:748</u>)
          import java.util.concurrent.locks.Lock;
import java.util.concurrent.locks.Condition;
import java.util.concurrent.locks.ReentrantLock;
                                                                                                    ▶ ↑
                                                                                                     =
                                                                                                         \downarrow
                                                                                                    11 등
          class Demo {
                                                                                                    o ±
             Lock lock1 = new ReentrantLock( fair: true);
Condition condition1 = lock1.newCondition();
Lock lock2 = new ReentrantLock( fair true);
Condition condition2 = lock1.newCondition();
                                                                                                             Process finished with exit code 130 (interrupted by signal 2: SIGINT)
                                                                                                     5 B
                                                                                                     a
                                                                                                         i
              public void worker1() throws InterruptedException {
                                                                                                     ==
                 lock1.lock();
System.out.println("lock 1 worker 1");
condition1.await();
System.out.println("lock 2 worker 1");
//lock2.unlock();
//lock1.unlock();
                                                                                                      *
             public void worker2() throws InterruptedException {
   lock2.lock();
   System.out.println("lock 1 worker 2");
   condition2.await();
   System.out.println("lock 2 worker 2");
   Alock2 worker 2");
}
. Z: Structure
¥ 2: Favorites
              public static void main(String[] args) throws InterruptedException {
                 Demo demo = new Demo();
Thread thread1 = new Thread(new Runnable() {
          Demo → main() → new Runnable
    ≡ 6: TODO ■ Terminal ≡ 0: Messages
                                                                                                                                                               C Event Log

☐ Stop process

                                                                                                                                           51:14 LF + UTF-8 + 4 spaces + 1
            =
                 ing 🕜 🖭 😆 🤰 🛄
import java.util.concurrent.locks.Lock;
import java.util.concurrent.locks.Condition;
import java.util.concurrent.locks.ReentrantLock;
class Demo {
        Lock lock1 = new ReentrantLock(true);
        Condition condition1 = lock1.newCondition();
        Lock lock2 = new ReentrantLock(true);
        Condition condition2 = lock1.newCondition();
        public void worker1() throws InterruptedException {
                 lock1.lock();
                 System.out.println("lock 1 worker 1");
                 condition1.await();
                 System.out.println("lock 2 worker 1");
                 //lock2.unlock();
                 //lock1.unlock();
        public void worker2() throws InterruptedException {
                 lock2.lock();
                 System.out.println("lock 1 worker 2");
                 condition2.await();
                 System.out.println("lock 2 worker 2");
                 //lock2.unlock();
                 //lock1.unlock();
        }
        public static void main(String[] args) throws InterruptedException {
```

```
Demo demo = new Demo();
         Thread thread1 = new Thread(new Runnable() {
             @Override
             public void run() {
                  try {
                      demo.worker1();
                  } catch (InterruptedException e) {
                      e.printStackTrace();
             }
         });
         Thread thread2 = new Thread(new Runnable() {
             @Override
             public void run() {
                  try {
                      demo.worker2();
                  } catch (InterruptedException e) {
                      e.printStackTrace();
             }
         });
         thread1.start();
         thread2.start();
         thread1.join();
         thread2.join();
    }
}
```