Exercise

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

```
/home/ttn/.sdkman/candidates/java/8.0.202-amzn/bin/java
Thread in Execution using Runnable Interface.
Thread in Execution using Thread Class
```

2. Use sleep and join methods with thread.

```
Exercise3
Empty Thread
Thread With Sleep.
Main Thread Executed Successfully.
Thread with loop.
0
1
2
3
4
5
6
7
8
9
ExecutorService Shutdown Successfully.
```

3. Use a singleThreadExecutor to submit multiple threads.

```
Exercise4
/home/ttn/.sdkman/candidates/java/8.0.202-amzn/bin/java ...
Thread with loop.
0
1
2
3
4
5
6
7
8
9
[java.util.concurrent.FutureTask@1540e19d, java.util.concurrent.FutureTask@677327b6]
Main Thread Executed Successfully.
```

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

```
Exercise3 ×   Exercise4 ×
/home/ttn/.sdkman/candidates/java/8.0.202-amzn/bin/java ...
Thread with loop.
java.lang.InterruptedException: sleep interrupted
0
1
2
    at java.lang.Thread.sleep(Native Method)
3
4
5
    at com.ttn.assignment.feb_21_multithreading.Exercise4$3.run(Exercise4.java:37)
6 <1 internal call>
7
8
9 <1 internal call>
Thread With 3 second Sleep.
Main Thread Executed Successfully. <2 internal calls>
    at java.lang.Thread.run(Thread.java:748)
```

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

```
Empty Thread
Thread with loop.
0
1
2
3
4
5
6
7
8
9
Thread With Sleep.
ExecutorService Terminated Successfully.
Main Thread Executed Successfully.

Process finished with exit code 0
```

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.

```
Exercise5 ×   Exercise6 ×
/home/ttn/.sdkman/candidates/java/8.0.202-amzn/bin/java ...
Exception in thread "main" java.util.concurrent.CancellationException <2 internal calls>
    at com.ttn.assignment.feb_21_multithreading.Exercise6.main(Exercise6.java:26)
Your task has been cancelled
```

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

```
Exercise5 ×   Exercise7 ×
/home/ttn/.sdkman/candidates/java/8.0.202-amzn/bin/java ...
Thread Waiting 1 seconds.
Thread Waiting 2 seconds.
ExecutorService waited for Two Seconds, tasks are still remaining

Process finished with exit code 0
```

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

```
Exercise8 ×
/home/ttn/.sdkman/candidates/java/8.0.202-amzn/bin/java ...
ScheduleWithFixedDelay Scheduled Task to executed after fixed interval
Task executed after 1 second
```

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

```
Exercise5 ×   Exercise9 ×
/home/ttn/.sdkman/candidates/java/8.0.202-amzn/bin/java
Thread Executing from newCachedThreadPool
Thread name :pool-1-thread-1 Start :1
Thread name :pool-1-thread-2 Start :2
Thread name :pool-1-thread-3 Start :3
Thread name :pool-1-thread-4 Start :4
Thread name :pool-1-thread-5 Start :5
Thread name :pool-1-thread-6 Start :6
Thread name :pool-1-thread-7 Start :7
Thread name :pool-1-thread-8 Start :8
Thread name :pool-1-thread-9 Start :9
Thread name :pool-1-thread-10 Start :10
Thread Executing from newFixedThreadPool
Thread name :pool-2-thread-2 Start :1
Thread name :pool-2-thread-3 Start :2
Thread name :pool-2-thread-4 Start :3
Thread name :pool-2-thread-5 Start :4
Thread name :pool-2-thread-1 Start :5
rminal   9: Version Control   4: Run   6: TODO
```

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

```
Exercise10 x   Exercise10 x
From Synchronized method : Thread-0
From Synchronized method : Thread-0
From Synchronized method : Thread-0
From Synchronized method : Thread-0
From Synchronized method : Thread-0
From Synchronized method : Thread-2
From Synchronized method : Thread-2
From Synchronized method : Thread-2
From Synchronized method : Thread-2
From Synchronized method : Thread-2
From Synchronized method : Thread-1
From Synchronized method : Thread-1
From Synchronized method : Thread-1
From Synchronized method : Thread-1
From Synchronized method : Thread-1
```

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

```
Exercise12 x
1
2
3
4
5
6
7
8
9
10
```

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

```
Exercise12
1
2
3
4
5
6
7
8
9
10
```

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

```
Exercise13
/home/ttn/.sdkman/candidates/java/8.0.20
Printing from 1
Printing from 2
Printing from 2 Completed.
Printing from 1 Completed.
```

14. Coordinate mulitple threads using wait() and notifyAll()

```
/home/ttn/.sdkman/candidates/java/8.0.202
1 On waiting.
3 On waiting.
Printing from 2.
Printing from 2 Completed.
Waiting over for 3.
Waiting over for 1.
```

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

```
Exercise15
/home/ttn/.sdkman/candidates/java/8.
consumer Started
producer Started
producer Finished
consumer Finished

Process finished with exit code 0
```

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

```
Exercise16
/home/ttn/.sdkman/candidates/java/
consumer Started
consumer Finished
producer Started
producer Finished
consumer Started
consumer Finished
```