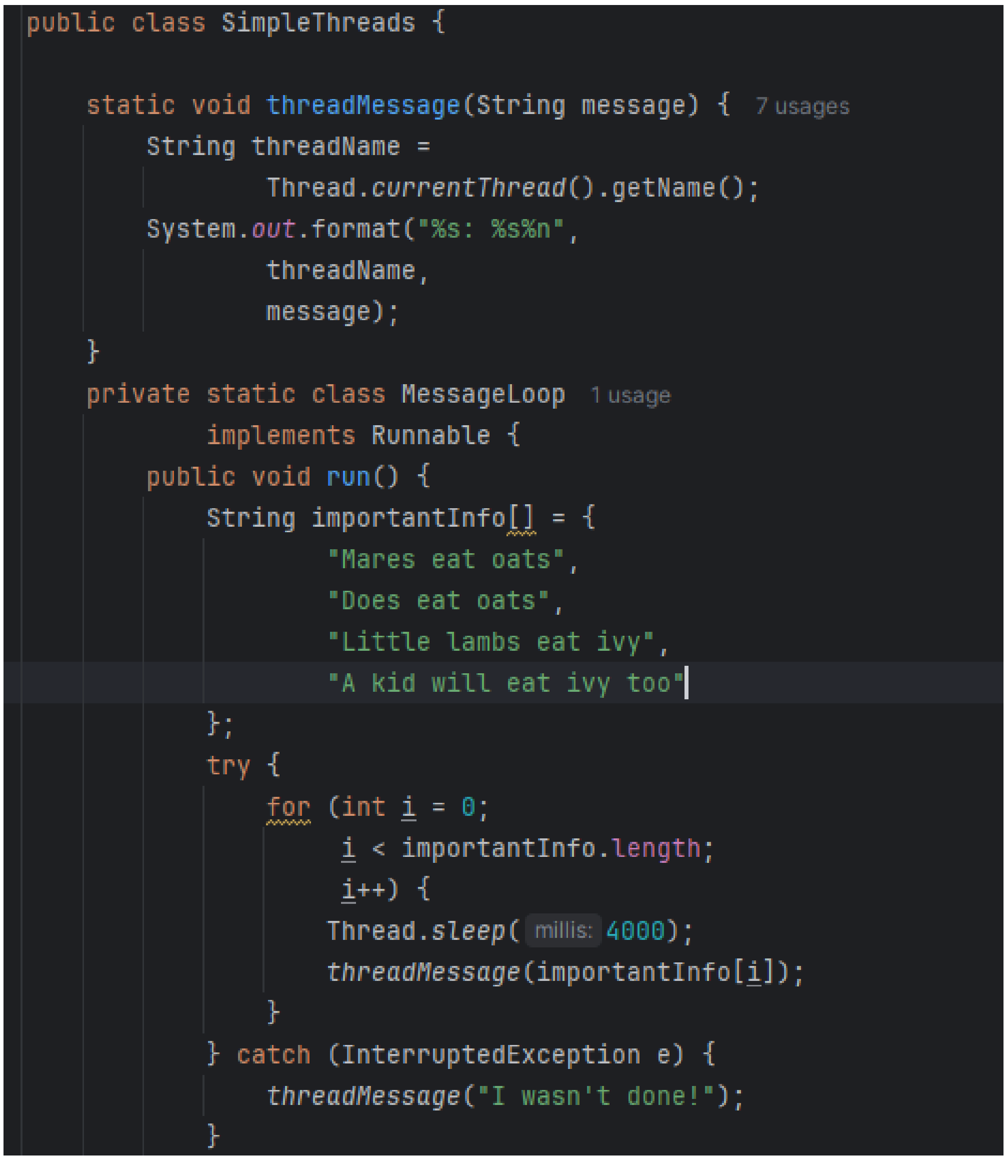
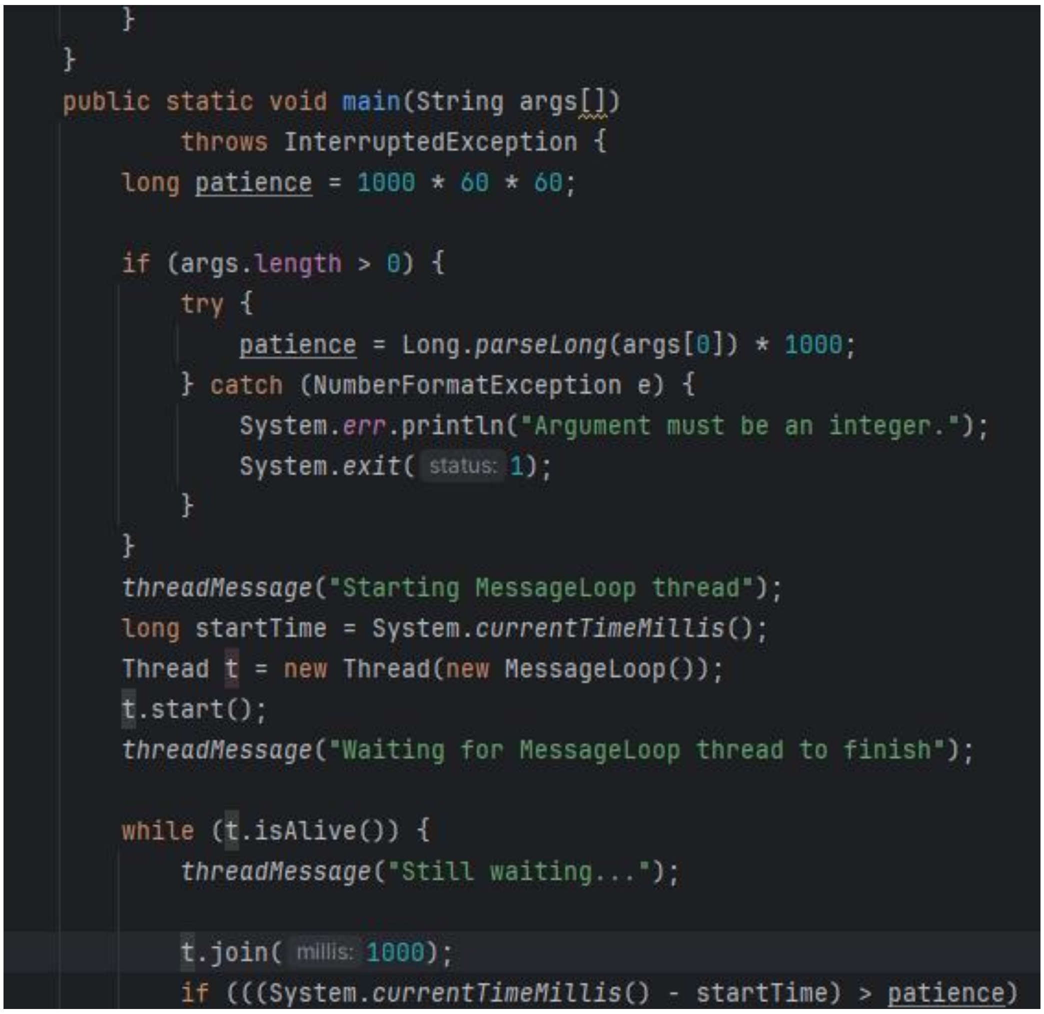
NUR AZYYATI BITNI ABU BAKAR | 291560

TUTORIAL 1 :
THE CODE:

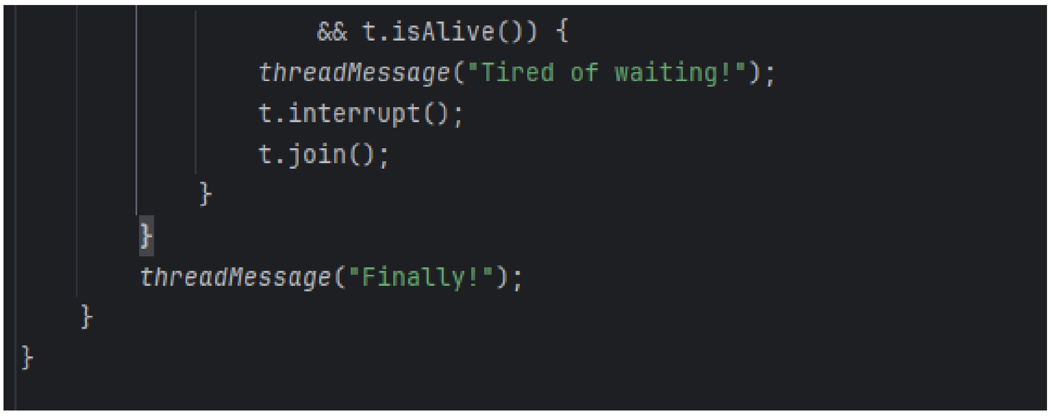
SimpleThreads.java



```java
public class SimpleThreads {

    static void threadMessage(String message) {  7 usages
        String threadName =
                Thread.currentThread().getName();
        System.out.format("%s: %s%n",
                threadName,
                message);
    }
    private static class MessageLoop  1 usage
            implements Runnable {
        public void run() {
            String importantInfo[] = {
                    "Mares eat oats",
                    "Does eat oats",
                    "Little lambs eat ivy",
                    "A kid will eat ivy too"
            };
            try {
                for (int i = 0;
                     i < importantInfo.length;
                     i++) {
                    Thread.sleep( millis: 4000);
                    threadMessage(importantInfo[i]);
                }
            } catch (InterruptedException e) {
                threadMessage("I wasn't done!");
            }
```

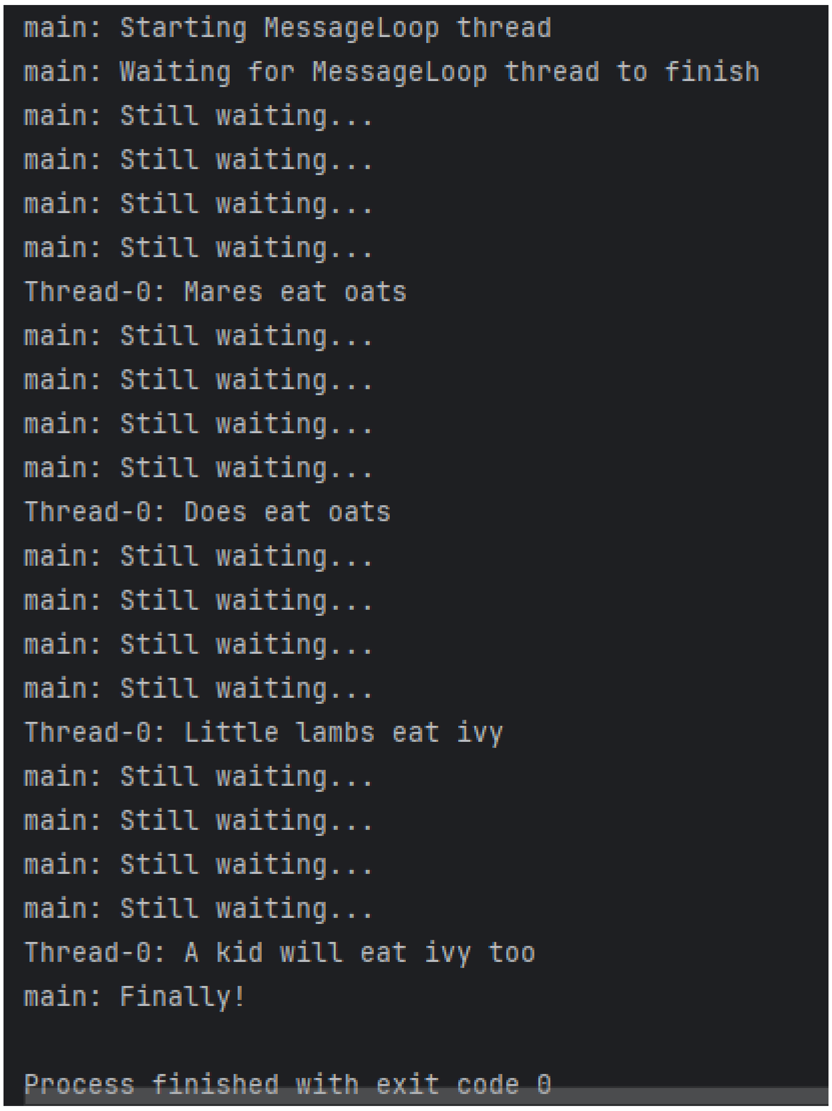


```
        }
    }
    public static void main(String args[])
            throws InterruptedException {
        long patience = 1000 * 60 * 60;

        if (args.length > 0) {
            try {
                patience = Long.parseLong(args[0]) * 1000;
            } catch (NumberFormatException e) {
                System.err.println("Argument must be an integer.");
                System.exit( status: 1);
            }
        }
        threadMessage("Starting MessageLoop thread");
        long startTime = System.currentTimeMillis();
        Thread t = new Thread(new MessageLoop());
        t.start();
        threadMessage("Waiting for MessageLoop thread to finish");

        while (t.isAlive()) {
            threadMessage("Still waiting...");

            t.join( millis: 1000);
            if (((System.currentTimeMillis() - startTime) > patience)
```



```
                    && t.isAlive()) {
                threadMessage("Tired of waiting!");
                t.interrupt();
                t.join();
            }
        }
        threadMessage("Finally!");
    }
}
```

THE OUTPUT:



```
main: Starting MessageLoop thread
main: Waiting for MessageLoop thread to finish
main: Still waiting...
main: Still waiting...
main: Still waiting...
main: Still waiting...
Thread-0: Mares eat oats
main: Still waiting...
main: Still waiting...
main: Still waiting...
main: Still waiting...
Thread-0: Does eat oats
main: Still waiting...
main: Still waiting...
main: Still waiting...
main: Still waiting...
Thread-0: Little lambs eat ivy
main: Still waiting...
main: Still waiting...
main: Still waiting...
main: Still waiting...
Thread-0: A kid will eat ivy too
main: Finally!

Process finished with exit code 0
```