# Taller Grupo 9
# El problema de la cena de los filósofos

1. Cree una archivo .java llamada “Folk” , y agregue el siguiente código (Folk.java)

```java
import java.util.concurrent.locks.ReentrantLock;

public class Folk {

  private ReentrantLock lock;

  public Folk() {
     this.lock = new ReentrantLock();
  }

  public void take() {
     lock.lock();
  }

  public void drop() {
     if (!isHeld())
        return;
     lock.unlock();
  }

  public boolean isHeld() {
     return lock.isHeldByCurrentThread();
  }
}
```

2. Cree una archivo .java llamada “Philosopher” , y agregue el siguiente código (Philosopher.java)

```java
public class Philosopher implements Runnable {

  private String name;
  private Table table;
  private Folk right;
  private Folk left;
  private boolean isLeftHanded;

  public Philosopher(String name, Table table, Folk left, Folk right, boolean isLeftHanded) {
     this.name = name;
     this.table = table;
     this.right = right;
     this.left = left;
     this.isLeftHanded = isLeftHanded;
  }

  public void think() throws InterruptedException {
        long time = table.getTime();
        System.out.println(name + " thinking during " + time + "ms");
        spendTime(time);
    }

    public void eat() throws InterruptedException {
        takeForks();
        long time = table.getTime();
        System.out.println(name + " eating during " + time + "ms");
        spendTime(time);
        dropForks();
    }

    public void run() {
        while (true) {
            try {
                think();
                eat();
            } catch (Exception e) {
                 System.out.println(e.getMessage());
            }
        }
    }

    private void takeForks() {
        if (isLeftHanded) {
            left.take();
            right.take();
        } else {
            right.take();
            left.take();
        }
    }

    private void dropForks() {
        if (isLeftHanded) {
            left.drop();
            right.drop();
        } else {
            right.drop();
            left.drop();
        }
    }

    private void spendTime(long time) throws InterruptedException {
        Thread.sleep(time);
    }
}
```

3. Cree una archivo .java llamada “Table” , y agregue el siguiente código (Table.java)

```
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
```

```java
import java.util.Random;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;

public class Table implements Runnable {

  private List<Folk> forks;
  private List<Philosopher> philosophers;
  private Iterator<Long> times;

  public Table(int numPhilosophers) {
    if (numPhilosophers < 2) {
      throw new IllegalArgumentException("There should be more than one philosopher");
    }

    this.forks = new ArrayList<>();
    this.philosophers = new ArrayList<>();
    this.times = new Random().longs(2000, 7000).iterator();

    for (int i = 0; i < numPhilosophers; ++i) {
      Folk f = new Folk();
      forks.add(f);
    }
    for (int i = 0; i < numPhilosophers; ++i) {
      int n = (i + 1) % numPhilosophers;
      Folk left = forks.get(i);
      Folk right = forks.get(n);
      boolean isLeftHanded = (n == 0);

      Philosopher p = new Philosopher("Philosopher " + (i + 1),this, left, right, isLeftHanded);
      philosophers.add(p);
    }
  }

  public synchronized long getTime() {
    return times.next();
  }

  public void run() {
    ExecutorService executorService = Executors.newFixedThreadPool(philosophers.size());
    for (Philosopher p : philosophers) {
      executorService.submit(p);
    }
  }
}
```

4. Cree una archivo .java llamada “Main” , y agregue el siguiente código (Main.java)

```java
public class Main {

  public static void main(String[] args) throws Exception {

    System.out.println("Setuping dinner...");
    Table table = new Table(5);
```

```
      Thread dinner = new Thread(table);

      System.out.println("Starting dinner...");
      dinner.start();
      dinner.join();
    }
}
```

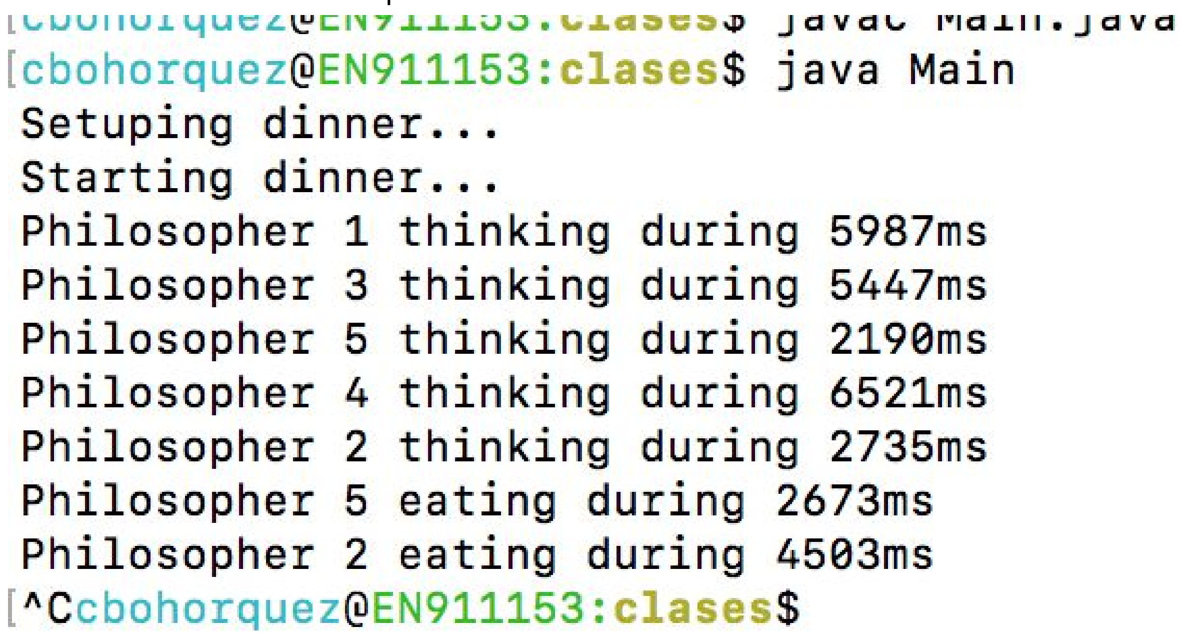
5. Compile los archivos

   ```
   javac Philosopher.java
   javac Folk.java
   javac Table.java
   Main.java
   ```

6. Ejecute la clase main

   ```
   java Main
   ```

7. El resultado debe ser parecido a esto:



```
[cbohorquez@EN911153:clases$ javac Main.java
[cbohorquez@EN911153:clases$ java Main
 Setuping dinner...
 Starting dinner...
 Philosopher 1 thinking during 5987ms
 Philosopher 3 thinking during 5447ms
 Philosopher 5 thinking during 2190ms
 Philosopher 4 thinking during 6521ms
 Philosopher 2 thinking during 2735ms
 Philosopher 5 eating during 2673ms
 Philosopher 2 eating during 4503ms
[^Ccbohorquez@EN911153:clases$
```