Assignment: To implement dining philosophers' problem using MongoDB

1. TestDP.java

```
import java.util.*;
import java.util.concurrent.locks.Lock;
import java.util.concurrent.locks.ReentrantLock;
import java.util.logging.Level;
import java.util.logging.Level;
import java.util.logging.Logger;
import com.mongodb.*;

public class TestDP {

	public static void main(String[] args) {
		  final int num=8; //number of philosophers
		Lock fork[]=new ReentrantLock[num];
		Philosopher P[]=new Philosopher[num];
		Thread T[]=new Thread[num];
		for(int i=0;i<num;i++){
			fork[i]=new ReentrantLock();
		}
		for(int i=0;i<num;i++){
			P[i]=new Philosopher(fork[i==0?num-1:i-1], fork[i], " "+i);
			System.out.println("Philosopher: "+i+" got left fork: "+(i==0?num-1:i-
1)+" right fork: "+i);
			T[i]=new Thread(P[i]);
		}
		try
		{
		  Logger mongoLogger = Logger.getLogger( "org.mongodb.driver" );
		  mongoLogger.setLevel(Level.SEVERE);
		    MongoClient mongoclient=new MongoClient("localhost");
		    System.out.println("\nConnection to mongodb successful!");
		    DB db=mongoclient.getDB("mydb");
		    System.out.println("Database mydb created.\n");
		    DBCollection coll=db.createCollection("mycol",null);
		    //System.out.println("collection mycol created");
		}
		catch(Exception e){
		}
		for(int i=0;i<num;i++){
			T[i].start();
		}
	}
}
```

2. Philosopher.java

```
import java.util.concurrent.locks.Lock;
import java.util.logging.Level;
import java.util.logging.Logger;
import com.mongodb.*;
import com.mongodb.BasicDBObject;
import com.mongodb.DBCollection;
import com.mongodb.MongoClient;


public class Philosopher implements Runnable{
	Lock leftFork,rightFork;
    String name;
```

```
    public Philosopher(Lock leftFork, Lock rightFork, String name) {
        this.leftFork = leftFork;
        this.rightFork = rightFork;
        this.name = name;
    }
    public void eat() throws InterruptedException{
        try {
            leftFork.lock();
            rightFork.lock();
            MongoClient mongoclient=new MongoClient("localhost");
            DB db=mongoclient.getDB("mydb");
            DBCollection coll=db.getCollection("mycol");
            System.out.println("Philosopher"+name +" eating.");
            BasicDBObject doc1=new BasicDBObject(name , " eating.");
            coll.insert(doc1);
            Thread.sleep(1000); //eating for a fixed amount of time.
        }

        catch(Exception e){
          e.printStackTrace();
        }

        finally {
            MongoClient mongoclient=new MongoClient("localhost");
            DB db=mongoclient.getDB("mydb");
            DBCollection coll=db.getCollection("mycol");
            System.out.println("Philosopher"+name +" full.");
            BasicDBObject doc2=new BasicDBObject(name ," full.");
            coll.insert(doc2);
            leftFork.unlock();
            rightFork.unlock();
        }
}
    public void think() throws InterruptedException{
        MongoClient mongoclient=new MongoClient("localhost");
        DB db=mongoclient.getDB("mydb");
        DBCollection coll=db.getCollection("mycol");
        System.out.println("Philosopher"+name +" thinking.");
        BasicDBObject doc=new BasicDBObject(name ," thinking.");
        coll.insert(doc);
        Thread.sleep(1000);
    }
    @Override
    public void run() {
       try {
         Logger mongoLogger = Logger.getLogger( "org.mongodb.driver" );
         mongoLogger.setLevel(Level.SEVERE);
           eat();
           System.out.println("Philosopher"+name+" done eating.");
           think();

       }
       catch (InterruptedException ex) {
           Logger.getLogger(Philosopher.class.getName()).log(Level.SEVERE, null, ex);
       }
    }
}
```

3. Console Output

```
Philosopher: 0 got left fork: 7 right fork: 0
Philosopher: 1 got left fork: 0 right fork: 1
Philosopher: 2 got left fork: 1 right fork: 2
Philosopher: 3 got left fork: 2 right fork: 3
Philosopher: 4 got left fork: 3 right fork: 4
Philosopher: 5 got left fork: 4 right fork: 5
Philosopher: 6 got left fork: 5 right fork: 6
Philosopher: 7 got left fork: 6 right fork: 7

Connection to mongodb successful!
Database mydb created.

Philosopher 6 eating.
Philosopher 0 eating.
Philosopher 2 eating.
Philosopher 4 eating.
Philosopher 4 full.
Philosopher 0 full.
Philosopher 0 done eating.
Philosopher 6 full.
Philosopher 2 full.
Philosopher 0 thinking.
Philosopher 6 done eating.
Philosopher 2 done eating.
Philosopher 7 eating.
Philosopher 2 thinking.
Philosopher 6 thinking.
Philosopher 3 eating.
Philosopher 4 done eating.
Philosopher 1 eating.
Philosopher 5 eating.
Philosopher 4 thinking.
Philosopher 7 full.
Philosopher 7 done eating.
Philosopher 7 thinking.
Philosopher 5 full.
Philosopher 3 full.
Philosopher 5 done eating.
Philosopher 1 full.
Philosopher 5 thinking.
Philosopher 1 done eating.
Philosopher 1 thinking.
Philosopher 3 done eating.
Philosopher 3 thinking.
```

4. MongoDB Output

```
[ccoew@localhost ~]$ su
Password:
[root@localhost ccoew]# service mongod start
Starting mongod (via systemctl):                           [  OK  ]
[root@localhost ccoew]# mongo
MongoDB shell version: 2.6.12
connecting to: test
> show dbs
admin       (empty)
local       0.078GB
mydb        0.078GB
philosoper  0.078GB
> use mydb
switched to db mydb
> show collections
mycol
system.indexes
> db.mycol.find({}, {_id: 0})
{ " 0" : " eating." }
{ " 2" : " eating." }
{ " 6" : " eating." }
{ " 4" : " eating." }
{ " 0" : " full." }
{ " 6" : " full." }
{ " 2" : " full." }
{ " 4" : " full." }
{ " 0" : " thinking." }
{ " 7" : " eating." }
{ " 2" : " thinking." }
{ " 6" : " thinking." }
{ " 1" : " eating." }
{ " 3" : " eating." }
{ " 4" : " thinking." }
{ " 5" : " eating." }
{ " 7" : " full." }
{ " 7" : " thinking." }
{ " 5" : " full." }
{ " 1" : " full." }
Type "it" for more
> it
{ " 3" : " full." }
{ " 5" : " thinking." }
{ " 1" : " thinking." }
{ " 3" : " thinking." }
```