```
//compute method returns workLoad * 3
//As workload is 200 and threshold is 20 --> it gets recursive divided and then mergedResult
200
100                                          100
50,50                                        50,50
25,25 ,25,25                          25,25,25,25
12,12,12,12,12,12,12,12     12,12,12,12,12,12,12,12

12*3*16=576-->mergedResult
-------------MyrecursiveTask.java----------
package com.ameya.test;

import java.util.ArrayList;
import java.util.List;
import java.util.concurrent.RecursiveTask;

public class MyRecursiveTask extends RecursiveTask<Long> {
    private long workLoad = 0;
    public MyRecursiveTask(long workLoad) {
        this.workLoad = workLoad;
    }
    protected Long compute() {
        // if work is above threshold, break tasks up into smaller tasks
        if (this.workLoad > 20) {
            System.out.println("Splitting workLoad : " + this.workLoad);
            List<MyRecursiveTask> subtasks = new ArrayList<MyRecursiveTask>();
            subtasks.addAll(createSubtasks());
            for (MyRecursiveTask subtask : subtasks) {
                subtask.fork();
            }
            long result = 0;
            for (MyRecursiveTask subtask : subtasks) {
                result += subtask.join();
            }
            return result;

        } else {
            System.out.println("Doing workLoad myself: " + this.workLoad);
            return workLoad * 3;
        }
    }
    private List<MyRecursiveTask> createSubtasks() {
        List<MyRecursiveTask> subtasks = new
        ArrayList<MyRecursiveTask>();
        MyRecursiveTask subtask1 = new MyRecursiveTask(this.workLoad
        / 2);
        MyRecursiveTask subtask2 = new MyRecursiveTask(this.workLoad
        / 2);
        subtasks.add(subtask1);
        subtasks.add(subtask2);
        return subtasks;
    }
}
--------------TestForkJoinPool.java----------
package com.ameya.test;

import java.util.concurrent.ForkJoinPool;

public class TsstForkJoinPool {

    public static void main(String[] args) {
        ForkJoinPool forkJoinPool = new ForkJoinPool();
        MyRecursiveTask myRecursiveTask = new MyRecursiveTask(200);
         System.out.printf("Main: Parallelism: %d\n",
         forkJoinPool.getParallelism());
        long mergedResult = forkJoinPool.invoke(myRecursiveTask);
         System.out.printf("Main: Active Threads: %d\n",
         forkJoinPool.getActiveThreadCount());
        System.out.println("mergedResult = " + mergedResult);
         System.out.printf("Main: Active Threads: %d\n",
         forkJoinPool.getActiveThreadCount());

    }

}

```