Assignment – 1

"Write a java program which accepts multiple employees details,

1)Create thread class

2)Execute them using frokjoinpool

3)make the use of runnable interface in it."

*Employee_Threadd.java ×

```java
2 import java.util.ArrayList;
3 import java.util.Collection;
4 import java.util.List;
5 import java.util.concurrent.RecursiveAction;
6 import java.util.concurrent.ForkJoinPool;
7 //-----------------------------------
8 class NewTask extends RecursiveAction{
9     private static final long serialvalueuid = 1L;
10    private long Load =0;
11    public NewTask(long load) {
12        this.Load = Load;
13    }
14    protected void compute() {
15        List<NewTask>subtasks = new ArrayList<NewTask>();
16        subtasks.addAll(createsubtasks());
17        for(RecursiveAction subtask: subtasks) {
18            subtask.fork();
19        }
20    }
21    private List<NewTask>createsubtasks(){
22        List<NewTask>subtasks = new ArrayList<NewTask>();
23        NewTask subtask1 = new NewTask(this.Load/2);
24        NewTask subtask2 = new NewTask(this.Load/2);
25        NewTask subtask3 = new NewTask(this.Load/2);
26        return subtasks;
27
28    }
29 }
30 //-----------------------------------
31 public class Employee_Threadd {
32
33     public static void main(final String[] args) throws InterruptedException{
34        // TODO Auto-generated method stub
35        int proc = Runtime.getRuntime().availableProcessors();
36        System.out.println("Number of available core in the processor is:");
37        ForkJoinPool Pool = ForkJoinPool.commonPool();
38        System.out.println("Number of active thread before invoking: "+Pool.getActiveThreadCount());
39        NewTask t = new NewTask(400);
40        Pool.invoke(t);
41        System.out.println("Common Pool Size: "+Pool.getPoolSize());
42        Pool.getPoolSize();
43    }
44 }
```

Console ×

Number of available core in the processor is:
Number of active thread before invoking: 0
Common Pool Size: 1

Activate Wi
Go to Settings t