```java
------------------TaskExecutorUsage.java----------
package com.ameya.test;

import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
import java.util.concurrent.Future;

import com.ameya.tasks.TaskOne;
import com.ameya.tasks.TaskTwo;

public class TestExecutorUsage {
    private static ExecutorService executor = null;
    private static volatile Future taskOneResults =null;
    private static volatile Future taskTwoResults = null;

    private static void checkTasks()throws Exception{
        if(taskOneResults ==null || taskOneResults.isDone()||
        taskOneResults.isCancelled()) {
            taskOneResults=executor.submit(new TaskOne());
        }
        if(taskTwoResults ==null || taskTwoResults.isDone()||
        taskTwoResults.isCancelled()) {
            taskTwoResults=executor.submit(new TaskTwo());
        }
    }

    public static void main(String[] args) {
        executor=Executors.newFixedThreadPool(2);
        while(true) {
            try {
                checkTasks();
                Thread.sleep(1000);
            }catch(Exception e) {
                System.out.println(e.getMessage());
            }
        }

    }

}
-----------------TaskOne.java----------
```

```java
package com.ameya.tasks;

public class TaskOne implements Runnable {

    @Override
    public void run() {
        System.out.println("Executing Task One");
        try {
            Thread.sleep(2000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        System.out.println("TaskOne Terminates....");
    }

}
-----------------TaskTwo.java----------
package com.ameya.tasks;

public class TaskTwo implements Runnable {

    @Override
    public void run() {
        System.out.println("Executing Task Two");
        try {
            Thread.sleep(2000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        System.out.println("Task Two Terminates....");
    }
}
----------------TaskThree.java----------
package com.ameya.tasks;

public class TaskThree implements Runnable {

    @Override
    public void run() {
        System.out.println("Executing Task Three");
        try {
                Thread.sleep(2000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            System.out.println("TaskThree Terminates....");
        }

}
-------------MultiRunnable.java----------
package com.ameya.tasks;

import java.util.List;

public class MultiRunnable implements Runnable {
    private List<Runnable> runnables;
    public MultiRunnable(List<Runnable> runnables) {
        this.runnables=runnables;
    }
    @Override
    public void run() {
        for(Runnable runnable : runnables) {
            new Thread(runnable).start();
        }

    }

}
-----------MultiRunnableTaskExecutor.java----------
package com.ameya.test;

import java.util.ArrayList;
import java.util.List;
import java.util.concurrent.ArrayBlockingQueue;
import java.util.concurrent.BlockingQueue;
import java.util.concurrent.RejectedExecutionHandler;
import java.util.concurrent.ThreadPoolExecutor;
import java.util.concurrent.TimeUnit;

import com.ameya.handlers.RejectedExecutionHandlerImpl;
import com.ameya.tasks.MultiRunnable;
import com.ameya.tasks.TaskOne;
import com.ameya.tasks.TaskThree;
import com.ameya.tasks.TaskTwo;

public class MultiRunnableTaskExecutor {

    public static void main(String[] args) {
        BlockingQueue<Runnable> worksQueue=new
        ArrayBlockingQueue<Runnable>(10);
        RejectedExecutionHandler rejectionHandler=new
        RejectedExecutionHandlerImpl();
        ThreadPoolExecutor executor=new
        ThreadPoolExecutor(3,3,10,TimeUnit.SECONDS,
                worksQueue,rejectionHandler);
        executor.prestartAllCoreThreads();
        List<Runnable> taskGroup=new ArrayList<Runnable>();
        taskGroup.add(new TaskOne());
        taskGroup.add(new TaskTwo());
        taskGroup.add(new TaskThree());
        taskGroup.add(new TaskTwo());
        taskGroup.add(new TaskThree());
        taskGroup.add(new TaskTwo());
        taskGroup.add(new TaskThree());
        worksQueue.add(new MultiRunnable(taskGroup));
    }

}
---------------RejectedExecutionHandlerImpl.java----------------
package com.ameya.handlers;

import java.util.concurrent.RejectedExecutionHandler;
import java.util.concurrent.ThreadPoolExecutor;

public class RejectedExecutionHandlerImpl implements
RejectedExecutionHandler {

    @Override
    public void rejectedExecution(Runnable r, ThreadPoolExecutor
    executor) {
        System.out.println(r.toString()+" Has Been Rejected ! ");

    }

}
------------------
```