**Advanced Java: Multi-threading Part 5 -- Thread Pools**

https://www.youtube.com/watch?v=KUdro0G1BV4&t=306s

**import** java.util.concurrent.ExecutorService;

**import** java.util.concurrent.Executors;

**import** java.util.concurrent.TimeUnit;

**class** Processor **implements** Runnable{

**private** **int** id;

**public** Processor(**int** id){

**this**.id=id;

}

**public** **void** run(){

System.*out*.println("Starting: "+id);

**try** {

Thread.*sleep*(5000);

} **catch** (InterruptedException e) {

}

System.*out*.println("Completed: "+id);

}

}

**class** apples {

**public** **static** **void** main(String args[]){

ExecutorService executor = Executors.*newFixedThreadPool*(2);

//We're assigning two workers. After finishing their threads, they'll start anew

**for**(**int** i=0;i<5;i++){

executor.submit(**new** Processor(i));

}

executor.shutdown();

System.*out*.println("All tasks submitted.");

**try** {

executor.awaitTermination(1, TimeUnit.*DAYS*);

} **catch** (InterruptedException e) {

}

System.*out*.println("All tasks completed.");

}

}

**Important notes:**

-By writing “executor.submit(**new** Processor(i));”, the executor automatically uses *start()* method of the Processor class.

-The programmer also used a similar method to *sleep()*, which is “executor.awaitTermination(1, TimeUnit.*DAYS*);”. When I removed this part, the line “All tasks completed.” popped up right after “All tasks submitted.” without waiting.