The University of the South Pacific

School of Computing, Information & Mathematical Sciences

CS214 Design & Analysis of Algorithms

Lab Week 5

This tutorial covers *big O* order and running the code in parallel (racing) using multi-threading.

Question 1. (big O order)

Show that $f(n) = n^2 + 3n^3 \in O(n^3)$ i.e., $0 \leq f(n) \leq cn^3$ for all $n > k$. Find c and k.

Question 2. (big O order)

What is the Big O order of function $f(n) = n^2 + n\log_2 n + \log_2 n$. Show your working

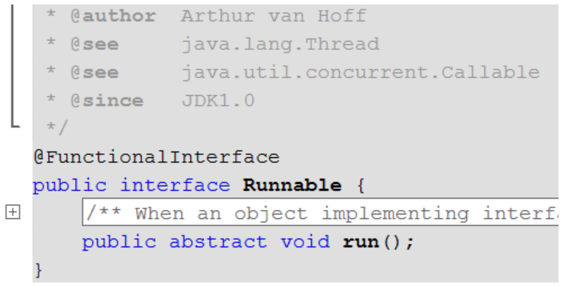
Question 3. (multi-threading in Java)

In computer architecture, multithreading is the ability of a central processing unit to execute multiple threads concurrently, supported by the operating system.

The operating system frequently switches back and forth between threads (tasks), giving illusion that they run in parallel. Java virtual machine executes each thread for a short amount of time and then switches to another thread. You can visualize the thread as programs executing in parallel to each other.

Creating a thread in Java:

1) Define a class which you want to run as a separate thread and implement the *Runnable* interface. This interface has a single method called run.

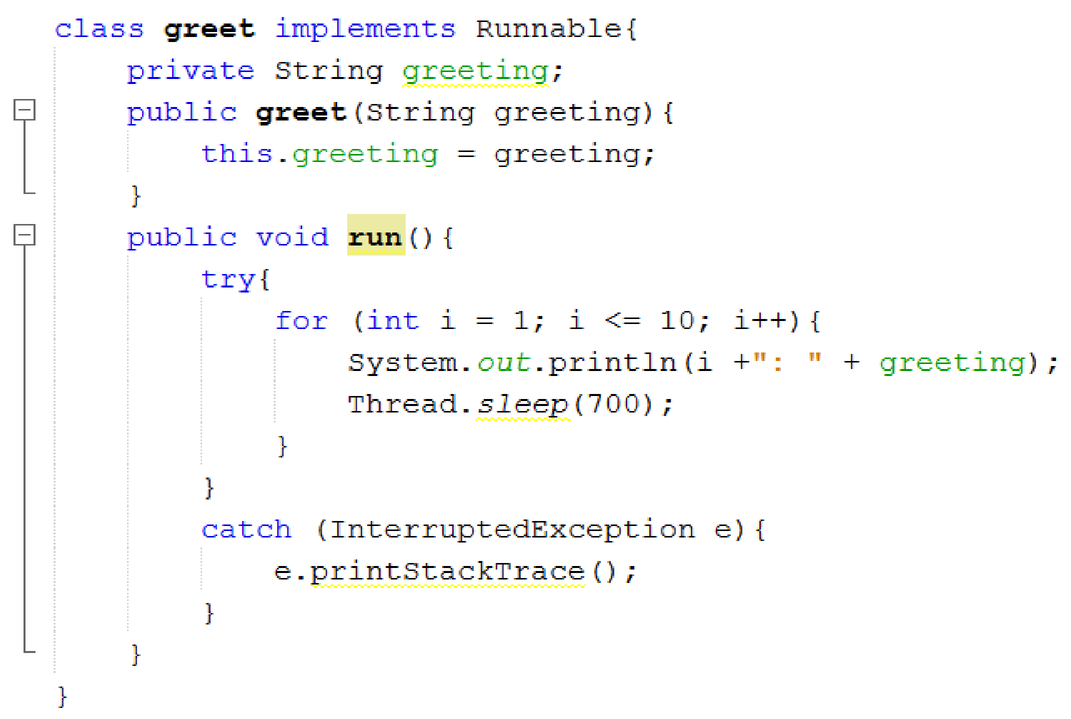


```
 * @author  Arthur van Hoff
 * @see     java.lang.Thread
 * @see     java.util.concurrent.Callable
 * @since   JDK1.0
 */
@FunctionalInterface
public interface Runnable {
    /** When an object implementing interf
    public abstract void run();
}
```

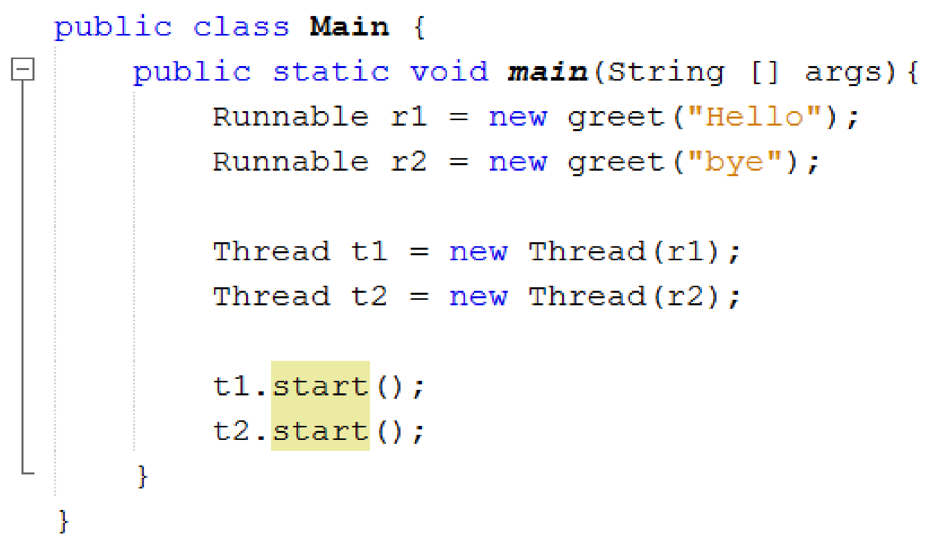
2) Place the code for the task into the run method of the class.
3) Create an object of the class
4) Construct a *Thread* object and supply the above object in the constructor.
5) Call the start method of the *Thread* object to start the thread.

Run the following code in Java:

- A. Sharma



```java
class greet implements Runnable{
    private String greeting;
    public greet(String greeting){
        this.greeting = greeting;
    }
    public void run(){
        try{
            for (int i = 1; i <= 10; i++){
                System.out.println(i +": " + greeting);
                Thread.sleep(700);
            }
        }
        catch (InterruptedException e){
            e.printStackTrace();
        }
    }
}
```



```java
public class Main {
    public static void main(String [] args){
        Runnable r1 = new greet("Hello");
        Runnable r2 = new greet("bye");

        Thread t1 = new Thread(r1);
        Thread t2 = new Thread(r2);

        t1.start();
        t2.start();
    }
}
```

What is your observation on the output?

- A. Sharma