COP 4600 - Homework 2

Due November 10, 2014

This problem demonstrates the use of semaphores to coordinate three types of processes. Santa Claus sleeps in his shop at the North Pole and can only be woken by either:

* All nine reindeer being back from their vacation in the South Pacific or
* Some of the elves having difficulties making toys.

To allow Santa to get some sleep, the elves can only wake him when three of them are having problems. When three elves are having their problems solved, any other elves wishing to visit Santa must wait for those elves to return. It is assumed that the reindeer do not want to leave the tropics, and therefore stay there until the last possible moment – they only return in December. The last reindeer to arrive must get Santa while the others wait in a warming hut before being harnessed to the sleigh.

The objective of this homework is for you to implement this problem as a multithreaded Java using the Java synchronization primitives. Note that implementing this problem requires you to deploy a number of different techniques.

As this is a relatively complex problem, I have decomposed it in a number of steps. You need to submit solutions to Steps 1…5. Normally, the solution for a step is the starting point for the next step.

# Step 0:

Download the Santa-Fall2014.zip file from the webpage. Compile and run (the main function is in SantaScenario.java). It should create a window which displays the current state of the participants. Study the code and notice that:

* in many places in to the code there are FIXME comments which give an idea of what is supposed to go there.
* for the time being, there are no semaphores or other synchronization primitives in the code
* the updateVisualizer function in SantaScenario updates the visualizer. This is now called once at the beginning of each day, so it won’t capture changes as they occur during the day… If you want that, you will need to call the updateVisualizer from other places in the code as well.

# Step 1: (20 pts)

Observe that the threads corresponding to Santa, the reindeer and the elves never terminate. Implement a way to terminate these threads at day 370 using deferred termination (i.e. do not kill the threads explicitly).

# Step 2: (25 pts)

Starting from step 1, create a version where:

-there is no reindeer present

-as soon as an elf runs into a problem, it goes to Santa’s door

-as soon as an elf is at Santa’s door and Santa is sleeping, he wakes up Santa

-if woken up, Santa solves the problems of all elves who are at the door.

# Step 3: (25 pts)

Starting from step 2, create a version where

-there is no reindeer present

-unless there are three elves which are in TROUBLE, the elves stay in TROUBLE. When the third elf gets in TROUBLE, they go to Santa’s door.

-if there is any elf at Santa’s door, the elves who get into TROUBLE, they stay there, and only go to Santa’s door when the previous elves came back.

 -as soon as an elf is at Santa’s door and Santa is sleeping, he wakes up Santa

-if woken up, Santa solves the problems of all elves who are at the door.

# Step 4: (25 pts)

Now, notice that Step 3 still did not use any synchronization primitives – even when in TROUBLE or at Santa’s door, the elf threads are spinning.

Using semaphores, change the code from Step 3 in such a way that the threads of the Elves are in waiting in the semaphore’s acquire() state when they are in the TROUBLE mode.

# Step 5: (25 pts)

Now, bring in the reindeers, and implement the code necessary such that:

-the first 8 reindeers to come back from BEACH stay at the WARMING\_SHED.

-the reindeers in the WARMING\_SHED are in the acquire() state

-the last reindeer wakes up Santa

-Santa hooks up the reindeers to the SLEIGH and wakes their thread up

## What to submit:

* The code as a zipped Java code with the subdirectories: Step 1, Step 2, … Step 5.