We define here a step by step explanation of the santa claus problem ::::::::::::::::

1. Reindeer < id >: on holiday . . . wish you were here, :)

2. Reindeer < id >: back from holiday . . . ready for work, :(

3. Santa: Ho-ho-ho . . . the reindeer are back!

4. Santa: harnessing reindeer < id > . . .

5. Santa: mush mush . . .

6. Reindeer < id >: delivering toys

7. Santa: woah . . . we’re back home!

8. Reindeer: < id >: all toys delivered

9. Santa: un-harnessing reindeer < id > . . .

In addition to the above, all Reindeer must report before Santa can report .The following messeges shows how Santa and Elves communicate with each other::::::::::::

1. Elf < id >: need to consult santa, :(

2. Santa: Ho-ho-ho . . . some elves are here!

3. Santa: hello elf < id > . . .

4. Elf < id >: about these toys . . . ???

5. Santa: consulting with elves . . .

6. Santa: OK, all done - thanks!

7. Elf < id >: OK . . . we’ll build it

8. Santa: goodbye elf < id > . . .

9. Elf < id >: working, :)

In addition to the message ordering, the Reindeer have priority over the elves, and only three Elves at a time may consult

with Santa. Moreover, freedom from deadlock and livelock are necessary; no process may

halt its execution indefinitely and the states of the entities must proceed as per the problem

description.