

Corrigendum to "A New Exercise in Concurrency"

John A. Trono
St. Michael's College
Winooski Park
Colchester, VT 05439
JAT@smcvax.smcvt.edu

A friend of mine (Stephen Hartley - Drexel University) pointed out that there is a race condition in the solution presented in my paper [1]; the Santa process can not reset the `rein_ct` shared variable without having the `P(rmutex)` and `V(rmutex)` operations surrounding that statement. If they are left off, as given in my solution, the ninth reindeer and the third elf, arriving at approximately the same time, can follow the scenario given below and invalidate the proposed solution. Adding the `P` and `V` as mentioned above should fix the problem.

Scenario: an elf process increments `elf_ct` (to 3) and performs `V(Santa)` and a reindeer process increments `rein_ct` (to 9). The Santa process wakes up, sees `rein_ct` is 9 and therefore resets it to zero. The reindeer process will then see that `rein_ct` is zero and wait on the `rein_wait` semaphore, whereas it should be waiting on the sleigh semaphore! (This will eventually lead to all reindeer waiting on the `rein_wait` semaphore next Christmas.)

I would like to thank Stephen for pointing this out to me so quickly, and I would also like to thank those who have e-mailed me and indicated that they liked the example and indicated that they are planning on using it in the future.

References

- [1] Trono, John A. A New Exercise in Concurrency, SIGCSE Bulletin, Volume 26, #3, September 1994.