Andrew Bare
Dante Gordon
SysNet 1
Nov 5th 2023

Lizard Threads Analysis

The object of this project is to create a shared mutually exclusive resource that is accessed by multiple threads. There are N lizard threads and M cat threads, the lizard threads must limit the occupancy of the sidewalk or else the M cat threads will wake up and terminate the program. The lizard threads must maintain a limit on how many lizards can use the sidewalk because the cats will wake up and terminate the entire program if too many lizards are using the shared sidewalk at once. We handle this requirement by using a set of counting semaphores that are assigned a #define MAX_LIZARD_CROSSING value at compile time to adjust the limit. We also have to implement a few mutexes to ensure mutual exclusion of the lizard counting variables and of the global world running state variable. Overall this project effectively simulates thread synchronization and the issues involved with engineering a solution to this type of problem. Some of the issues you can run into are deadlocks, race conditions and bottlenecks from over use of mutexes.

Below is a table of values documenting our results.

WORLDEND (s)	Maximum Number of Lizards Crossing	Lizards safe?
30	4	Yes
120	4	Yes
600	8	Yes
1800	4	Yes

In conclusion, we ran the program with various settings changing amount of run time and number of max lizards crossing. Changing these values did not impact the safety of the threads nor did it cause the program to crash. The hardest part of this project was resolving a deadlock during development, however there weren’t too many problems during development.