COSC1112/1114: Operating Systems Principles

Lab 7 (week 9)

Temperature Convergence (2)

Four external processes will communicate temperatures to a central process, which in turn will reply with its own temperature and will indicate whether the entire system has stabilized. Each process will receive its initial temperature upon creation and will recalculate a new temperature according to two formulas:

*new external temp = (10*myTemp + centralTemp) / 11;*

new central temp = (centralTemp + 1000 temp sum of external processes) / (1000*number of external processes + 1);*

Initially, each external process will send its temperature to the central process through mailbox. If all external temperatures are exactly the same as that of the central process, the system has stabilized. In this case, the central process will notify each external process that it is now finished (along with the central process itself), and each process will output the final stabilized temperature. If the system has not yet become stable, the central process will send its new temperature to each of the external processes through mailbox and await their replies. The processes will continue to run until the temperature has stabilized. The C programs of the external and central are provided in separate file (note don't remove instruction *usleep(100000)* in all tasks in this assignment). Compile and run as follows

#./external 10 1 &
#./external 100 2 &
#./external 1000 3 &
#./external 10000 4 &
#./central 10 &

- *Note when the processes of different students are running at the same time, the mailbox number are shared among processes of different students. So, each student should have own unique mailbox id. The rule is to use your student ID as the mailbox id. Suppose your student ID is sxxxxxxx. Then, xxxxxxx will be your mailbox id; for the two mailboxes in task-3, the mail ids will be xxxxxxx00 and xxxxxxx11 respectively.*
- *The process runing in background (i.e., ./process &) should be killed once it is not terminated successfully. To kill such process, you need find the process id first using:*

  ps -aux | grep "name of the process"

  You will see something, for example, like"xxxxx 34770 0.0 0.0 3920 348 pts/22 S 17:54 0:00 ./process" where process id is 34770; then, kill the process use:

  kill 34770
- *Each of the two groups in task-2 and 3 has its own converged temperature.*

Simple Extension

Write central8.c and external8.c to perform the same task as described by the specifications where 8 external processes are considered. Compile and run as follows:

```
#./external8 10 1 &
#./external8 100 2 &
#./external8 1000 3 &
#./external8 10000 4 &
#./external8 50 5 &
#./external8 500 6 &
#./external8 5000 7 &
#./external8 50000 8 &
#./central8 10&
```

Serve Two Group

Now consider there are two groups of external processes where the first group has external process 1-4 and the second group has external process 5-8. Each group of external processes communicate with the central to perform the same task as described by the specifications. The central process is with a single thread of control. For the first and second group, the initial temperature in the central is 10 and 100 respectively. Write central44.c and external44.c. Compile and run as follows (record the central process running time):

```
#./external44 10 1 1200 &
#./external44 100 2 1200 &
#./external44 1000 3 1200 &
#./external44 10000 4 1200 &
#./external44 50 5 1300 &
#./external44 500 6 1300 &
#./external44 5000 7 1300 &
#./external44 50000 8 1300 &
#./central44 10 100 1200 1300 &
```

(Note: 1200 and 1300 are two central mailbox numbers respectively, each for one group of external processes)