Laboratory IX
Semaphores, Mutual Exclusion

1. Semaphores

- **Semaphore**; code34.c
  - A common strategy to avoid race conditions is to use semaphores.
  - The use of semaphores is important to prevent simultaneous access to system resources by separate processes or separate threads inside the same process.
  - Three system calls to create, use, and release semaphores:
    - **semget** - Returns an integer semaphore index that is assigned by the kernel
    - **semop** - Performs operations on the semaphore set
    - **semctl** - Performs control operations on the semaphore set

1. Semaphores

- The program shows how to create a semaphore set and how to access the elements of that set. Does the following:
  - Creates a unique key and creates a semaphore,
  - Checks to make sure that the semaphore is created OK,
  - Prints out the value of the semaphore at index 0 (should be 1),
  - Sets the semaphore (decrements the value of semaphore at index 0 to 0),
  - Prints out the value of the semaphore at index 0 (should be 0),
  - Unsets the semaphore (increments the value of semaphore at index 0 back to 1),
  - Prints out the value of the semaphore at index 0 (should be 1),
  - Removes the semaphore.
- Study the code.
- Execute several times and observe that how the output changes.
- Is there any possible race conditions? Explain.

2. Mutual Exclusion (Mutex)

- **Mutex**; code32.c
  - Several threads and shared data.
  - Mutex mechanism (pthread mutex lock) is used for concurrent executing.
  - Execute code several times and observe that how the execution order of the threads changes.

3. Starving Philosophers Simulation

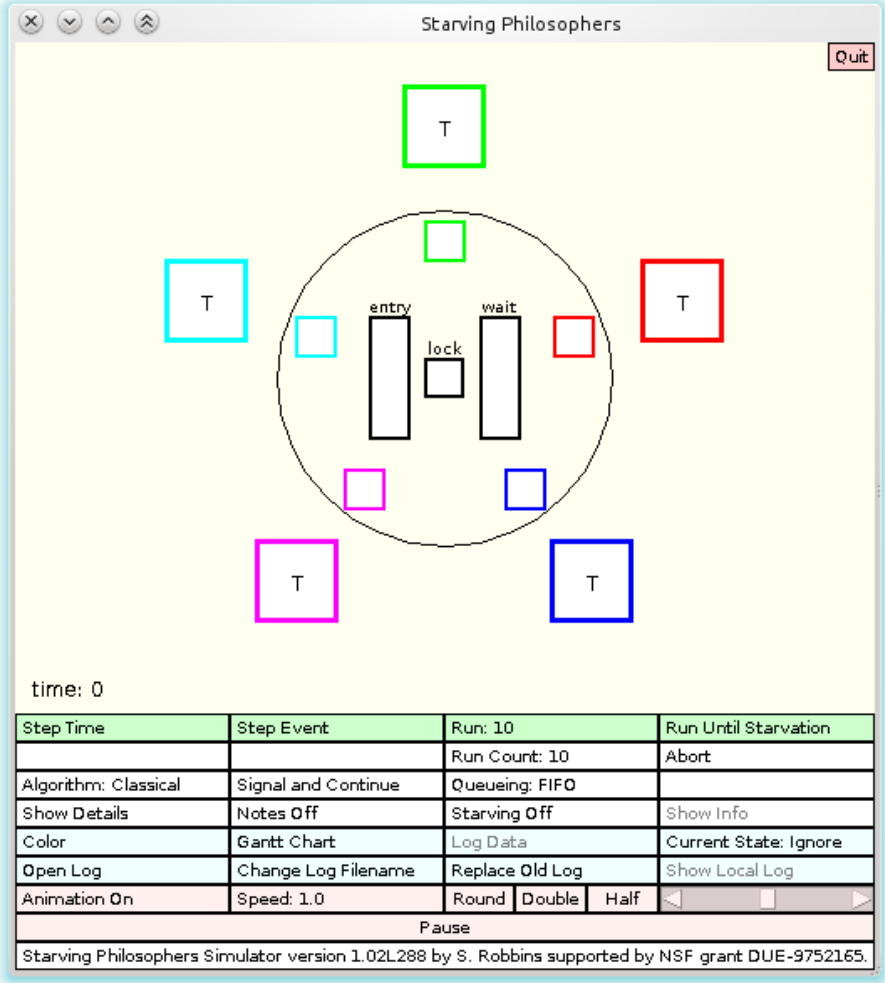
- Download the simulator from link: StarvingPhilosophers.tar.gz.
- Unpack the simulator with the following command:

  cd Downloads
  tar zxvf StarvingPhilosophers.tar.gz

- Run the simulator with the following command:

  cd StarvingPhilosophers
  chmod +x runsp
  ./runsp

- First of all, read the **spdoc.html** file carefully. This file contains detailed information about running and modifying the parameters of the simulation.



3. Starving Philosophers Simulation

- Each time you want to modify simulation parameters such as number of philosophers or whether starving is enabled or not, you have to edit the **spconfig** file and restart the simulator. The contents of spconfig file look like this:

```
number 5
starving off
animate on
queueing fifo
eatingdist constant 100
thinkingdist constant 100
starvingdist constant 900
thinkingdist1 constant 3
thinkingdistvalue1 2
eatingdist1 constant 2
thinkingdist2 constant 1
eatingdist2 constant 4
eatingdistvalue2 2
```

- Each line in this file hold various parameters related to the simulation.
- After starting the simulator, click on the "**Run Until Starvation**" button. This will start the simulation.
- Animated sequence of simulation will start and continue until simulation has been aborted or starving has occured.

3. Starving Philosophers Simulation

- You may draw a gannt chart for displaying how long each philosopher spent their time for thinking, eating or as hungry by clicking on the "**Gannt Chart**" button.
- For different sample runs, have a look at **demos** directory.
- **Exercise:** Modify the simulation parameters to create a deadlock (starvation). You are free to modify any parameters you like to, but you should explain your method.