Dining Philosophers Problem

Dining Philosophers Problem

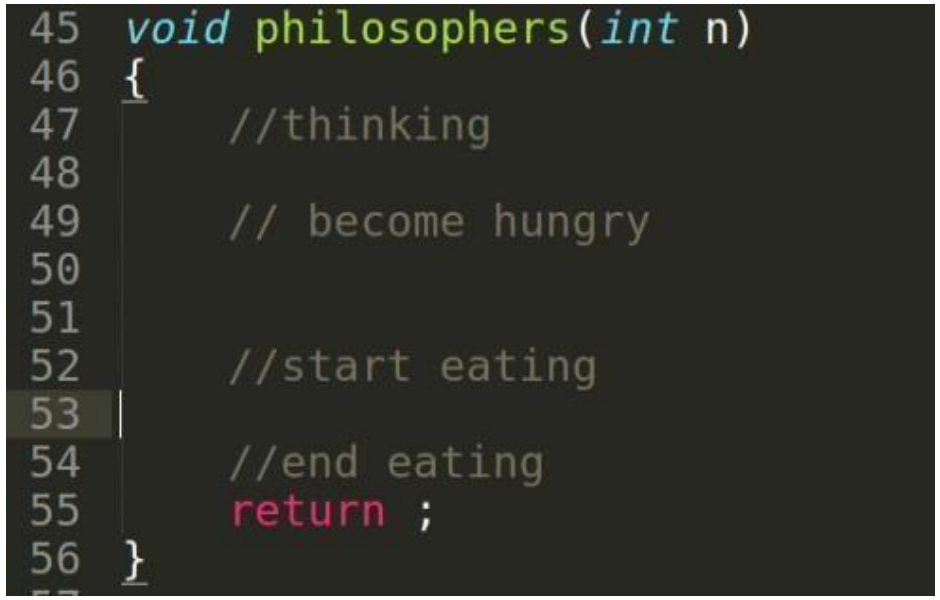
- In this homework, you need to write a program to simulate the famous dining philosophers problem.

- This problem will require implementing a solution using Pthreads mutex locks and condition variables.

Implemetaion

- Begin by creating five philosophers, each identified by a number 0, 1, 2, 3 and 4. Each philosopher will run as a separate thread.
- Philosophers alternate between three states which is thinking, hungry and eating. To simulate thinking and eating, have the thread sleep for a random period from one to three seconds.
- Each philosopher should think for a while and then become hungry.

- The thread of each philosopher will be created and joined in order.
- Use a philosophers function as the input of pthread_create() to control the philosophers' actions.



```
45  void philosophers(int n)
46  {
47      //thinking
48
49      // become hungry
50
51
52      //start eating
53
54      //end eating
55      return ;
56  }
```

- Invoke the function named pickup_forks as below when a philosopher finishes thinking. where philosopher number indicates which philosopher calls this function.

```
pickup_forks(int philosopher_number)
```

- Invoke the function named test as below when a philosopher tries to eat.

```
test(int philosopher_number)
```

- Invoke the function named return_forks as below when a philosopher finishes eating.

```
return_forks(int philosopher_number)
```

- Since Pthreads is typically used in C programs—and since C does not have a monitor— we accomplish locking by associating a condition variable with a mutex lock.
- Condition variables in Pthreads use the data type pthread_cond_t and are initialized using the pthread_cond_init() function.
- E.g.

```
pthread_mutex_t mutex;
pthread_cond_t cond_var;

pthread_mutex_init(&mutex,NULL);
pthread_cond_init(&cond_var,NULL);
```

- You should print these lines out in the correct situations:
  - Philosopher %d is now THINKING for %d seconds.
  - Philosopher %d is now HUNGRY and trying to pick up forks.
  - Philosopher %d can’t pick up forks and start waiting.
  - Philosopher %d is now EATING.
  - Philosopher %d returns forks and then starts TESTING %d and %d.

- E.g.

```
rohan@rohan-VirtualBox:~/Desktop$ ./hw3.out
Philosopher 0 is now THINKING for 2 seconds
Philosopher 1 is now THINKING for 2 seconds
Philosopher 2 is now THINKING for 1 seconds
Philosopher 3 is now THINKING for 2 seconds
Philosopher 4 is now THINKING for 3 seconds
Philosopher 2 is now HUNGRY and trying to pick up forks.
Philosopher 2 IS NOW EATING.
Philosopher 0 is now HUNGRY and trying to pick up forks.
Philosopher 0 IS NOW EATING.
Philosopher 1 is now HUNGRY and trying to pick up forks.
Philosopher 1 fails to pick up forks and then starts waiting.
Philosopher 3 is now HUNGRY and trying to pick up forks.
Philosopher 3 fails to pick up forks and then starts waiting.
Philosopher 4 is now HUNGRY and trying to pick up forks.
Philosopher 4 fails to pick up forks and then starts waiting.
Philosopher 2 returns forks and then starts TESTING 1 and 3.
Philosopher 3 IS NOW EATING.
Philosopher 0 returns forks and then starts TESTING 4 and 1.
Philosopher 1 IS NOW EATING.
Philosopher 3 returns forks and then starts TESTING 2 and 4.
Philosopher 4 IS NOW EATING.
Philosopher 1 returns forks and then starts TESTING 0 and 2.
Philosopher 4 returns forks and then starts TESTING 3 and 0.
rohan@rohan-VirtualBox:~/Desktop$
```

File format

- Hw4_{studentID}.zip
  - Hw4.c(90%)
  - Hw4_report.pdf(10%)

    Tell us how you implement your homework and show us your results.

Cautions

- Hand in your homework in right format, or deduction will be applied to your grade.
- 0 will be given to cheaters, so don’t copy & paste your friend’s code
  directly.
- Deadline: 6/9(Tue.)
- we will pick ¼ of students to demo in person.