OS HW4

mutex & semaphore

Operating System 108 Fall

Professor: W.J. TSAI

Deadline：2019/12/19 23:59

Objective - Series

- (40%) Calculate the total number of occurrences of each number in the series

- Ex.
  - Input : 1233211234567890
  - Output :
    0 : 1
    1 : 3
    2 : 2
    3 : 3
    4 : 1
    5 : 1
    6 : 1
    7 : 1
    8 : 1
    9 : 1

Objective - Pi

- (50%) Estimating the value of Pi using Monte Carlo
- The "Monte Carlo Method" is a method of solving problems using statistics.
- Given the probability, P, that an event will occur in certain conditions, a computer can be used to generate those conditions repeatedly. The number of times the event occurs divided by the number of times the conditions are generated should be approximately equal to P.
- Algorithm reference：

  https://www.geeksforgeeks.org/estimating-value-pi-using-monte-carlo/

Synchronization - mutex lock

- Only use:
  #include <pthread.h>
- Declare: (global variable)
  pthread_mutex_t mutex1 = PTHREAD_MUTEX_INITIALIZER;
- Functions:
  - pthread_mutex_lock()
    - acquire a lock on the specified mutex variable. If the mutex is already locked by another thread, this call will block the calling thread until the mutex is unlocked.
  - pthread_mutex_unlock()
    - unlock a mutex variable. An error is returned if mutex is already unlocked or owned by another thread.
  - pthread_mutex_trylock()
    - attempt to lock a mutex or will return error code if busy. Useful for preventing deadlock conditions.

Synchronization - semaphore

- #include <pthread.h>
  - Declare: (global variable)
    pthread_cond_t cond1 = PTHREAD_COND_INITIALIZER;
  - Functions:
    - pthread_cond_wait
    - pthread_cond_signal
    - pthread_cond_broadcast

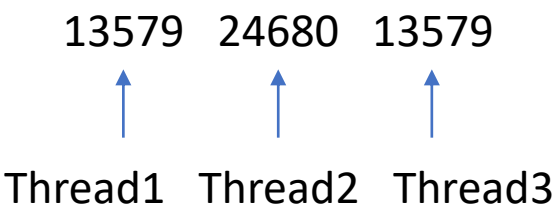
- #include <semaphore.h>
  - Declare: (global variable)
    sem_t sem1;
  - Functions:
    - int sem_post(sem_t *);
    - int sem_wait(sem_t *);
    - int sem_close(sem_t *);

Input and Output - Series

- The following should be as same as your Terminal：

Compiled execution file ->
You need to print your student ID first ->
Thread(s) that would be use (1 ~ 4) ->
Series length (1 ~ 10^8) ->
The series ->
Your output ->

Red words are inputs that TA will provide



Compile Commands：
$ g++ -Wall -o hw4_Series hw4_Series.c -lpthread

```
$ ./hw4_Series
0756613
3
15
135792468013579
The total number of 0 : 1
The total number of 1 : 2
The total number of 2 : 1
The total number of 3 : 2
The total number of 4 : 1
The total number of 5 : 2
The total number of 6 : 1
The total number of 7 : 2
The total number of 8 : 1
The total number of 9 : 2
```

Input and Output - Pi

- The following should be as same as your Terminal：

Compiled execution file ->
Your student ID ->
Thread(s) would be use (1 ~ 4) ->
The point number (1 ~ 10^8) ->
Your output ->

printf("Pi : %lf\n", your_answer); ->

```
$ ./hw4_Pi
0756613
4
100000
Thread 0, There are 19626 points in the circle
Thread 1, There are 19612 points in the circle
Thread 2, There are 19643 points in the circle
Thread 3, There are 19632 points in the circle
Pi : 3.140520
```

Compile Commands：
$ g++ -Wall -o hw4_Pi hw4_Pi.c -lpthread

Red words are inputs that TA will provide

Requirements and Suggestion

- Hang in hw4_Series.c and hw4_Pi.c *without compress*
- Series
  - You must and can only use pthread
  - You must and can only use mutex or semaphore
  - ~~More threads should spend less time~~
  - Use only one thread result will not be used for scoring
  - Use long or long long to declare variables
  - Use malloc and char format to save input series
- Pi
  - You must use pthread and Monte Carlo Algorithm
  - You must and can only use mutex or semaphore
  - More threads should spend less time
  - Random functions have a great impact on speed

Grading

- Any cheating situation or delay (both hw4 0 point)
- Any file format error (hw4 0 point)
- Compress files into one file (hw4 0 point)
- Series (40%) hw4_Series.c
  - Without use pthread / mutex or semaphore (-40%)
  - ~~More threads spend more or same time (-40%)~~
  - Pass two testbench each get 20%
- Pi (50%) hw4_Pi.c
  - Any Condition that do not meet the requirements (-50%)
  - More threads spend more or same time (-50%)
  - Pass two testbench each get 25%
- Report (10%) hw4_0756613.pdf (use your student ID)
  - Without any format restriction but at least 100 words.