# Homework 5 for CSC246

Homework 5 consists of 2 questions. Respond to each question and submit your work electronically using Moodle. Please submit your non-program answers in either plain text, pdf, rtf or Microsoft Word format (doc or docx is fine). For the programming questions, submit all source files. **Also include a readme file with compilation instructions (or include them in a comment block at the top of your code)!** Be sure to include the school environment you ran the code on

For this homework, the 1st problem can be answered using either c/c++ pthreads or Java threads.

If you need help, please contact the TA at once and arrange to get help. The TA's email address is on the syllabus.  
  
This homework is due on the date mentioned on the main website **by midnight.** This homework contains 50 points and is 5% of your total course grade.

*25 points*

## Question 1: Dining Threads (Barriers)

In this question you will implement a barrier for threads. A barrier is a thread synchronization mechanism that allows a thread to do some work, and then block the thread until some condition is satisfied.  
  
Here is the scenario to implement. Four hungry threads are meeting at a restaurant to eat. Since these threads have good manners, they will not begin eating until the last thread has arrived. Instead, while they are waiting they will begin doing some work they brought with them. **Each thread must work on something different.**   
  
When each thread arrives at the restaurant, output, to standard console output, "I am thread [number of thread], working on Task [type of task], and waiting to eat".  
  
For example, when thread 1 arrives to the Restaurant, it should output:  
  
*I am thread 1, working on Task A, and waiting to eat.*  
  
When thread 2 arrives to the Restaurant, it should output:  
  
*I am thread 2, working on Task B, and waiting to eat.*  
  
Designate different tasks with different capital letters.  
  
When the fourth thread has arrived at the Restaurant, and only when the last thread has arrived, each thread should output "I am thread [number of thread] and I am eating."

The last thread to arrive should output two messages, one that it is waiting as well as one that it is eating.  
  
For example, thread 1 would output:  
  
*I am thread 1 and I am eating.*  
  
Finish the partially completed code provided on the main page to implement a barrier according to the specifications in this problem. You may use the C++ or Java barrier code. Turn in completed work for either

 barrier.cpp or

 Barrier.java and DiningThread.java

Java programmers should review the code provided in the Producer/Consumer/CubbyHole linked on the Resources page. This code demonstrates the use of a synchronized bounded buffer. The same synchronization technique can be used in the barrier.

*25 points*

## Question 2: Sleeping Professor

This problem involves finding the values of semaphores. A professor decides to use counting semaphores to manage her office hours with students. The professor's office consists of a waiting area with 3 chairs and a room with one chair to accommodate one student at a time.

If there are no students to be seen, the professor goes to sleep. When a student arrives, the student has to awaken the sleeping professor.

If a student enters the waiting area and all chairs are occupied, then the student leaves. If the professor is busy with another student, but chairs are available in the waiting area, then the student sits in one of the free chairs. If the professor is asleep, the student wakes up the professor.

By the way, this problem is similar to various queuing situations such as a multi-person help desk with a computerized call waiting system for holding a limited number of incoming calls. View the counting semaphore code for the Sleeping Professor on the course website and answer the following questions:

 Assume the professor arrives for her office hours and then a student arrives. Assume the professor is currently consulting with this student. Provide each semaphore count:

|  |  |
| --- | --- |
| **Semaphore** | **Count** |
| students | 0 |
| mutex | 1 |
| professor | 1 |

 While the professor is talking with the first student, 4 more students arrive. Show the resulting count for the semaphores:

|  |  |
| --- | --- |
| **Semaphore** | **Count** |
| students | 3 |
| mutex | 1 |
| professor | -2 |

 The professor finishes talking with the first student and he leaves. The professor begins talking with the second student. Show the resulting count for the semaphores while the professor is talking with the second student

|  |  |
| --- | --- |
| **Semaphore** | **Count** |
| students | 2 |
| mutex | 1 |
| professor | -1 |