**Overview:** Modern computing systems have multiple processes running for very few, or even a single processor. A processor can only hand the instructions of a single process at a time. In order to give concurrency, we must have the processes share time on processor usage. Many different strategies exist for scheduling processes and there is no true silver bullet. Each one has advantages and disadvantages. This lab will explore some of these techniques.

**Description:** You have been given the files 341HW5.cpp, OS\_Threads.h and input.txt. These files will provide a simulation of a single processor having to schedule multiple different simultaneous threads. The main file creates the object containing our simulated threads and runs all the scheduling functions. You must modify the code in OS\_Threds.h to implement the functions for running the 3 following scheduling techniques: Round Robin, Shortest Remaining Time Next, Highest Response Ratio Next. You must print out the turn around time (TAT) for each thread and the normalized TAT. Once all threads are complete you must print out the average (mean) Normalized TAT for all the threads.

**Input:** You have been provided an input file but it is advisable you make your own to thoroughly test your program. The input file is structure like so:

* First number is the number of threads to create
* Each line afterwards contains 2 numbers separated by a single space
  + First number is the arrival time of the thread
  + Second number is the service time of the thread
* You can always assume that each line will have an increasing arrival time (meaning line 3 will never have an earlier arrival time than line 2)

**Submission:** Submit your source code (both of them, the modified driver and your header file) to the server. You must also write a lab report and upload it to dropbox on D2L. You lab report should contain the following information

* Title Page
* Description of the problem
* How did you approach the problem?
* Give an analysis of how the different scheduling techniques compare
  + Which ones were faster?
  + Give advantages and disadvantages to each?
* Which of the methods implemented can result in starvation?
  + The ones that do not have starvation, how are they able to prevent it?