# CS 403 Algorithm Design & Analysis Lab
# Assignment 3

- Submit a report (with full explanation of your algorithm's running time and complexity) along with the codes and read me file in a zipped folder. The report should be in PDF format as a single document. If you want to assume something during coding, then mention in your report.
- The deadline of the submission is 11:49 am 28 March 2018 . Late submissions will have penalty of 15% per day (that is 15% per day will be reduced on the score you achieve as the late submission penalty).
- You have to do code for all questions and give a good explanation in your report. Your reports would be evaluated thoroughly. Please provide pseudo codes in report.
- We will provide test data sets at the time of evaluation. In that case, your code should be well generalized. Analyze your codes with different test sets during implementations of algorithms.
- Submit your assignments only to coursetacs403@gmail.com

1) Implement a recursive algorithm for weighted interval scheduling using
   a) Compute-Opt($j$)
   b) M-Compute-Opt($j$) (Refer section 6.1 from book)
   c) Find-Solution (Refer page 258)

   Compute the running time and draw recursion tree.

| Tasks | Start time (sec) | End time (sec) | Weights |
|---|---|---|---|
| Task 1 | 3 | 7 | 3 |
| Task 2 | 5 | 8 | 2 |
| Task 3 | 8 | 15 | 4 |
| Task 4 | 9 | 17 | 1 |

2) Implement merge-and-count and sort-and-count for some input dataset. Fix you input data size and use any random number generator. Also find running time (Refer section 5.3).

3) Implement closest-pair and find running time for your input dataset (refer section 5.4).

4) Write pseudo code for an algorithm to compute all least squared errors in $O(n^2)$. Implement the algorithm (Refer section 6.3).

5) Given a graph $G$, find the shortest path between $s$ and $t$ node using Dijkstra algorithm. You are expected to use a priority queue for the implementation of the algorithm.
   Input : The first line contains two integers $V, E$ denoting the no. of vertices($V$) and no. of edges($E$) in a graph. Next, $E$ lines contain three integers $x, y$, and $w$ which denotes the edge $x - y$ with $w$ weight. The last line contains an integer denoting the source vertex s.
   Output : From $s$ to every other reachable node, print the shortest path and its value.