# UE14CS311: Advanced Algorithms (5th Sem Elective-1)

# Assignment 1: Growth of Functions, Complexity – Analysis

## FAQs:

1. Deadline extended?
    a. Till 7th Sep 11:59 PM for online submits.
    b. Till 8th Sep 3:45 PM for the graph plotted on a sheet of paper.
2. Submitting the source-code is not allowed because **"We couldn't save your response because it was too long."**.
    **a.** When you tried to submit your response to the Google Form, if it says **"We couldn't save your response because it was too long."**, it may mean that your sourcecode is too long. In that case, reduce the code size and submit again. **Alternatively, you can drop a zip file of all sourcecodes at https://drive.google.com/folderview?id=0B-E2SGzqkISrcEJKemJwTy16bXM&usp=sharing. Your filename must be your SRN/USN, all in caps.**
3. How submit the graph plotted?
    a. Plot the graph with five curves on a linear scale. Normally, the curves should compare as expected, but it's okay if your results are not convincing. You could write the graph on a sheet of paper or take a print of the graph. Submit the sheet of paper (couple of sheets in case of more analysis done on the results) with your USN/SRN and name written on it. Prof. Rashmi and Prof. Divya will be collecting them for their respective sections (B-201 and B-202). Make sure you submit that before 3:45 PM on Thursday.
4. Parallelized versions are taking more time than the counterparts.
    a. 4th and 5th Sourcecodes are "parallelized" versions of 2nd and 3rd. Parallelized versions **should not take more time** than their counterparts. If creating a thread is an overhead for smaller sized matrices, don't create threads at that stage. If even after your good efforts, the results are convincing, don't worry, just report your observations.
5. The execution time is in seconds or millisecond? The form says in seconds up to microseconds precision but the document says correct to a millisecond.
    a. It's corrected. It's in seconds with precision up to a microsec.
6. The output of each source code is just the execution time or the product matrix and the execution time?
    a. Just the execution time to be entered in the Submit Form. However, make sure the product matches the expected product matrix.
7. If your implementation cannot handle larger matrix sizes, just mention the exception name like **java.lang.outofmemoryException** in place of the execution time in the Submit Form.
8. The sourcecode for matrix multiplication should read 'n' indicating nxn matrix followed by two sets of n^2 elements. Output to have the execution time as the last thing to display. Everything you display before that will be ignored and only you should make sure the correctness of the output. **Do not submit** the code you might have written to **generate input-cases** with random numbers.

## Guidelines:

1. Code can be written in any of C/C++/C#/Java/Python (Open source compiler IDEs).
2. Submission will have be to done through the following Google Form on or before the deadline (6th Sep 2016). You will have to paste your source code in the Google Form https://goo.gl/forms/0shXOKDteDleN99L2 during submission along with timing (in seconds with least count of a microsecond) data for specific input sizes. It requires you to submit 5 sourcecodes, 9 execution times for each sourcecode, and conclusion. A graph plotted with 5 superimposed curves, one for each sourcecode to be submitted on a sheet of paper in the classroom.
3. Follow fair code of ethics and develop your own version of the code. Plagiarism is to be avoided. Discussion with others is encouraged though.
4. You will be called upon to demo the assignment, to match with submission data you have provided in the Google Form.

## Problem Definition, Data Generation, Testing and Logging Time

1. Generate two NxN matrices M1 and M2, with float data type, using built in random number functions, of your dev environment.
2. Generate the product M1xM2 using
    a. Standard inner product based row-col multiplication algorithm. (sourcecode = S1, result = M1, Time taken = T1)
    b. O(n^3) Divide-n-Conquer strategy-based multiplication algorithm. (sourcecode = S2, result = M2, Time taken = T2)
    c. Strassen's recursive divide-n-conquer strategy-based algorithm. (sourcecode = S3, result = M3, Time taken = T3)
    d. Parallelize O(n^3) Divide-n-Conquer strategy-based multiplication algorithm. (sourcecode = S4, result = M4, Time taken = T4)
    e. Parallelize Strassen's recursive divide-n-conquer strategy-based algorithm. (sourcecode = S5, result = M5, Time taken = T5)
3. Verify equality of M1, M2, M3, M4, and M5.
4. Vary the input size to cover the following input sizes N = 16, 32, 64, 128, 256, 512, 1024, 2048, 4096 (nine input sizes N1 to N9) and log the times T1, T2, T3, T4, and T5 and check against theoretical complexity growth expected.
5. Finally let us know your observations, approximate number of hours spent on the assignment and learning outcomes about the assignment.