# Department of Computing

**CS 213: Advanced Programming**

**Class: BSCS 5 AB**

# Lab 1: Matrix Computer

**Date: September 21st, 2017**

**Time: Thursday (10:00-12:50 & 14:00 – 16:50)**

# Instructor: Fahad Ahmed Satti

# 

# Lab 1: Matrix Computer

## Introduction

In this lab, students will create a Java based matrix computer, which will take as input multiple matrices and apply user requested operations on them. Students will also create unit tests to ensure their code is working properly and within defined constraints and will use GitHub to manage their solutions.

## Objectives

After performing this lab students will be able to understand:

* Java
* Unit Tests
* GitHub as a VCS
* Profiling

## Tools/Software Requirement

* Solutions should be made using Java only.
* You can take help from internet but remember **no plagiarism.**

## Description

In mathematics, a matrix (plural matrices) is a rectangular array of numbers, symbols, or expressions, arranged in rows and columns. Matrices are used in several fields (Physics, computer graphics, probability theory, etc) to study scientific phenomena’s. A major branch of numerical analysis is devoted to the development of efficient algorithms for matrix computations, a subject that is centuries old and is today an expanding area of research.

Applications of matrices are found in most scientific fields. In every branch of physics, including classical mechanics, optics, electromagnetism, quantum mechanics, and quantum electrodynamics, they are used to study physical phenomena, such as the motion of rigid bodies. In computer graphics, they are used to project a 3-dimensional image onto a 2-dimensional screen. In probability theory and statistics, stochastic matrices are used to describe sets of probabilities; for instance, they are used within the PageRank algorithm that ranks the pages in a Google search.

In this lab you will build a Matrix Calculator that will allow the user to input named matrices (Allow the user to give names to his matrices like A, B, C, d, firstMatrix, secondMatrix, e.t.c.) followed by an equation with the matrix names and operations(Sum, Difference, Scalar Multiplication, Transpose, Matrix Multiplication, and Matrix Inverse). Your program should then replace the Matrix names with the Matrix, and apply the operations, finally producing the result.

Use Unit Tests to ensure your application is working appropriately. Write Unit Tests for individual operations, in memory Matrix Storage, and one with an equation with all operations.

Each student must, individually build the complete application on their own. Students must upload their solutions on LMS to qualify for evaluation.

In order to qualify for maximum marks in Code Management, also perform profiling and use Maven.

## Lab Task

1. Create a Matrix Computer which allows the user to evaluate a matrix equation and get the ultimate results.
   1. All Matrices should be kept in-memory till the application closes and must have a unique name
   2. Allow the user to enter an equation with the Matrix Names and Operations
   3. Evaluate the equation with appropriate values
   4. The allowed operations on already provided matrices will include any combination of:
      1. Matrix Sum
      2. Matrix Difference
      3. Matrix Multiplication
      4. Scalar Multiplication
      5. Matrix Transpose
      6. Matrix Inverse
   5. Apply basic checks on sanity of the equation (e.g. Matrix Multiplication rule)
2. Unit Tests to prove the correctness of your solution.
   1. Write individual Unit tests to verify the individual Matrix Operations
   2. Write a unit test to check all operations in a single equation
   3. Check the in-memory storage of Matrices is correct
   4. Any other appropriate unit tests.
3. Final application source code should be uploaded on GitHub

## Deliverables

* Each submission is individual with the following composition:
  + Source Code
  + Unit Tests
  + Documentation(Introduction, Approach, Design, How to Run and Analysis including Profiling results, Link to the public repo on GitHub, Screen Shots of the running application)
* Convert your submission files into a zip folder and name it as given below, finally upload the zip folder to LMS.
  + Name – Registration No. – Section

## Grade Criteria

This lab will be graded on the following rubric, with min marks 0 and max marks10:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ****Criteria**** | ****Wt.**** | ****0**** | ****1**** | ****2**** | ****3**** | ****4**** |
| ****Analysis and**** ****approach**** | **0.5** | Unable to plan and set objectives for the realization of the lab task. | In between | Adequate analysis of the project. Objectives have been set, but strategies to follow are not clearly stated. | In between | Complete analysis of the task has been done. Objectives have been set. Strategies to follow have been defined. |
| ****Clarity**** | **1** | Student has no knowledge of both problem and solution. Cannot answer basic questions | Student does not have grasp of information; student cannot answer questions about subject matter | Student is uncomfortable with information and is able to answer only rudimentary questions | Student has competent knowledge and is at ease with information. Can answer questions but fails to elaborate. | Student has presented full knowledge of both problem and solution. Answers to questions are strengthen by rationalization and explanation |
| ****Completeness and Accuracy**** | **1.5** | The system failed to produce the right accurate results | The system execution led to inaccurate or incomplete results. It was not correctly functional or not all the features were implemented. | In between | The system was correctly functional and most of the features were implemented | The system was correctly functional and all of the features were implemented |
| ****Coding Standards (Packaging, Unit Tests, Profiling, Debugging, Testing)**** | **1** | Coding standards, best programming practices are not followed. Students cannot understand the code. | Coding standards, best programming practices are not followed. | Coding standards, best programming practices are rarely followed. | Coding standards, best programming practices are followed appropriately | Coding standards, best programming practices are followed extensively |
| ****Originality**** | **0.5** | Most part of the working product is copied | Working product is uninspired and straightforward work with little to no creative potential. | Working product has some potential for making a creative contribution. | Working product has some creative /original /inventive element and a potential for making a creative contribution | Working product has several creative /original /inventive /innovative elements and a clear potential for making a creative contribution. |
| ****Code Documentation**** | **0.5** | Code has not been commented or only small meaning-less comments are present. | The documentation is simply comments embedded in the code and does not help the reader understand the code. | The documentation is simply comments embedded in the code with some simple header comments separating routines. No sample run is shown. | The documentation consists of embedded comment and some simple header documentation that is somewhat useful in understanding the code. Sample run is clearly shown covering some objectives. | The documentation is well written and clearly explains what the code is accomplishing and how. Sample run is clearly shown covering all objectives. Profiling of the application has been done appropriately and is shown in the documentation. |