|  |  |  |
| --- | --- | --- |
| American University of Sharjah  College of Engineering  Department of Computer Science & Engineering  P. O. Box 26666, Sharjah, UAE |  | **Instructors:** Dr. Michel Pasquier  **Lab Instructor:** Praveena Kolli  **Office:** EB2-126  **Phone**: 971-6-5152352  **e-mail**: pkolli@aus.edu  **Semester**: Spring 2021 |

**CMP 305L - Data Structures and Algorithms Lab**

**Lab. Assignment 1**

**Design, implementation and testing of a Matrix Class**

***Objectives:***

* Review and implement C++ classes and templates, dynamic memory allocation, using pointers.
* Review and implement constructors and the “Big 3”, member and friend functions, operator overloading.

**Note:**

***Lab:*** Exercises 1 (10 marks)

***Bonus and optional*:** Exercise 2 (2 marks)

***Exercise 1:***

Create a template class **Matrix** that has two-dimensional array of any number of rows (with a default of 4 rows) and any number of columns (with a default of 4 columns). **Matrix** should make use of a standard C++, dynamically allocated two-dimensional array to store its elements.

**Design, implement and test** the **Matrix** class. Your class should contain at least the following features, namely:

         A default constructor and a destructor.

         A copy constructor and an assignment operator “=”.

         A comparison operator “==”.

         Matrix addition “+” and subtraction “-“ operators.

         Matrix addition assignment “+=” and subtraction assignment “- =” operators.

         Matrix insertion “<<” and extraction “>>” operators (using row-major format).

* **operator ()** to provide indexing. For example, in a 3-by-5 **Matrix** called **A**, the programmer could write A (1,3) to access the element at row 1 and column 3. There should be two versions of **operator( )**, one that returns **int &** so an element of a **Matrix** can be used as an *lvalue* and one that returns **const int &** so an element of a Matrix can be used as an *rvalue*.
* Two versions of operator ++: post and pre increment operators that adds 1 to every element of the matrix.

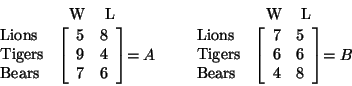
***Note****:* You must create an **array of pointers** to implement Matrix.

Class definition and driver is provided in Lab1.cpp file.

***Bonus and optional***

***Exercise 2:***

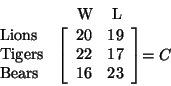
Matrix *A* represents the number of wins and losses for three teams in one year and *B* represents the number of wins and losses for the next year. 



Create matrices A and B using the Matrix class and initialize them with respective number of wins and losses.

* Print the teams’ records for the two years combined.

If the three seasons record for these teams is represented by *C*,



* How many games did each team win and lose in the third year?
* Calculate and print the average of total wins and losses. Use () operator and update matrix class as required.