Department of Computer and Software Engineering – ITU

CE101T: Object Oriented Programming

Course Instructor: Nadir Abbas	**Dated: 23/02/2025**
Teaching Assistant: Zainab, Nahal, Bilal	**Semester: Spring 2025**
Session: 2024-2028	**Batch: BSCE2024**

Assignment 5. Operator Overloading And Friend Functions

Name	Roll number	Obtained Marks/35

Checked on: _____

Signature: _____

Submission:

- Email instructor or TA if there are any questions. You cannot look at others' solutions or use others' solutions, however, you can discuss it with each other. Plagiarism will be dealt with according to the course policy.
- Submission after due time will not be accepted.

In this assignment you have to do following tasks:

Task 1: Ensure that you have installed all three softwares in your personal computer (Github, Cygwin & CLion). Now, accept the assignment posted in classroom (e.g Google, LMS etc) and after accepting, clone the repository to your computer. Make sure you have logged into the github app with your account.

Task 2: Open Cygwin app, Move to your code directory with following command "cd <path_of_folder>"
<path_of_folder> can be automatically populated by dragging the folder and dropping it to the cygwin window.
Run the code through Cygwin, use command "make run", to get the output of the code

Task 3: Solve the given problems, write code using **CLion** or any other IDE.
Task 4: Keep your code in the respective git cloned folder.
Task 5: Commit and Push the changes through the Github App
Task 5: Write the code in separate files **(as instructed)**. Ensure that file names are in lowercase (e,g **main.cpp**).

Task 6: Run '**make run**' to run C++ code
Task 7: Run '**make test**' to test the C++ code

Question 1: Define the Matrix Class

Create a class **Matrix** that represents a **3×3 matrix**.

Members:

- **int mat[3][3]**: 3×3 integer matrix

Constructors:

- **Default Constructor:** Initializes all elements to zero.
  Matrix();
- **Parameterized Constructor:** Initializes with a **2D array**.
  Matrix(int arr[3][3]);
- **Copy Constructor:** Copies elements from another Matrix.
  Matrix(const Matrix& other);
- **Destructor:** Releases resources if needed.
  ~Matrix();

Getter and Setter Functions:

- **void setElement(int row, int col, int value);** Set a value at a specific position.
- **int getElement(int row, int col) const;** Get a value from a specific position.

Question 2: Overload the + Operator

Overload the + operator to **add two matrices element-wise**.

Matrix operator+(const Matrix& m) const;

Question 3: Overload the - Operator

Overload the - operator to **subtract matrices element-wise**.

Matrix operator-(const Matrix& m) const;

Question 4: Overload the * Operator

Overload the * operator to perform **matrix multiplication**.

Matrix operator*(const Matrix& m) const;

Question 5: Overload == and != Operators

- Overload == to check if **two matrices are identical**.
  bool operator==(const Matrix& m) const;
- Overload != to check if **two matrices are different**.
  bool operator!=(const Matrix& m) const;

Question 6: Overload < and > Operators

Compare matrices **based on the sum of their elements**.

< should return true if the sum of **first matrix is smaller**.
bool operator<(const Matrix& m) const;

> should return true if the sum of **first matrix is greater**.
bool operator>(const Matrix& m) const;

Question 7: Overload the Unary - Operator

Overload - to **negate all matrix elements**.

Matrix operator-() const;

Question 8: Overload Pre-Increment (++)

Pre-Increment: Increments all elements by **1**.

Matrix& operator++();

Question 9: Overload Input (>>) and Output (<<) Operators

Use **friend functions** to handle I/O operations.

<< should **display** the matrix.
friend ostream& operator<<(std::ostream& out, const Matrix& m);

>> should **take user input** for a matrix.
friend istream& operator>>(std::istream& in, Matrix& m);

Example Input:

Enter matrix elements:
1 2 3
4 5 6
7 8 9

Example Output:

Matrix:
1 2 3
4 5 6
7 8 9

Question 10: Overload the Assignment Operator (=)

Overload the = operator to correctly assign **one matrix to another**.

Matrix& operator=(const Matrix& m);

Question 11: Create a UML Diagram for the Matrix Class

Draw a UML class diagram representing the Matrix class. Your diagram should include:

- **Class Name**
- **Attributes** (data members)
- **Methods** (constructors, setters, getters, operator overloads, etc.)
- **Access Specifiers** (public/private)

Save the UML diagram as an image and include it in your assignment submission folder.

Assessment Rubric for Assignment

Performance metric	CLO	Able to complete the task over 80% (4-5)	Able to complete the task 50-80% (2-3)	Able to complete the task below 50% (0-1)	Marks
1. Realization of experiment	3	Executes without errors excellent user prompts, good use of symbols, spacing in output. Through testing has been completed.	Executes without errors, user prompts are understandable, minimum use of symbols or spacing in output. Some testing has been completed.	Does not execute due to syntax errors, runtime errors, user prompts are misleading or non- existent. No testing has been completed.	
2. Conducting experiment	2	Able to make changes and answered all questions.	Partially able to make changes and few incorrect answers.	Unable to make changes and answer all questions.	
3. Computer use	4	Document submission timely.	Document submission late.	Document submission not done.	
4. Teamwork	4	Actively engages and cooperates with other group member(s) in effective manner.	Cooperates with other group member(s) in a reasonable manner but conduct can be improved.	Distracts or discourages other group members from conducting the experiment	
5. Laboratory safety and disciplinary rules	2	Code comments are added and does help the reader to understand the code.	Code comments are added and does not help the reader to understand the code.	Code comments are not added.	
6. Data collection	2	Excellent use of white space, creatively organized work, excellent use of variables and constants, correct identifiers for constants, No line-wrap.	Includes name, and assignment, white space makes the program fairly easy to read. Title, organized work, good use of variables.	Poor use of white space (indentation, blank lines) making code hard to read, disorganized and messy.	
7. Data analysis	3	Solution is efficient, easy to understand, and maintain.	A logical solution that is easy to follow but it is not the most efficient.	A difficult and inefficient solution.	
Total (out of 35):					