Object-Oriented Programming in C++

Prof. Caterina Doglioni and Dr. Charanjit Kaur

University of Manchester

PHYS3072 Assignment 5

Semester 2, 2022-23

Assignment 5: A Matrix Class

Write a C++ class for matrices:

- The class should at least contain the following private data: the number of rows, the number of columns and a pointer to the matrix data (matrix data should be of type double).
- When constructing a matrix object, dynamic memory allocation should be used to store the matrix data.
- Store the data in a one-dimensional array. For an $m \times n$ matrix A, the location loc of the element A_{ij} is then given by $\text{loc} = (j-1) + (i-1) * n$
- The $(i-1) * n$ and $(j-1)$ terms are due to the fact that C++ arrays start at zero, while the indices of our matrices start at 1!
- The destructor should explicitly delete any dynamically allocated memory when called.
- The assignment operator and copy constructor functions should perform deep copies of the data.
- Challenge: a recursive calculation of a determinant.
- The submission deadline is 7 pm, 24th March 2023. The expected time to complete it is 5 hours (including the challenge).

Marking Criteria (Total 9 Marks)

- The minimum content of the class: it must correctly implement the following member functions:
  - Parameterized constructor (utilising new) and destructor (utilising delete) (1 mark).
  - Member function to overload assignment operator that performs a deep copy of data. It must also handle self-assignment (0.5 mark).
  - A copy constructor and copy assignment operator that both perform a deep copy of the class data (1 mark).
  - An efficient coded move constructor and move assignment operator (1 mark).
  - A friend function to overload << producing nicely formatted output for a matrix object; and a friend function to overload >> to input a matrix in a sensible form of your choice (0.5 mark).

Marking Criteria (Total 9 Marks)

- Operational functionality:
  - Functions overloading + and − for matrix addition and subtraction (functions must check both matrices have the same dimensions) (1 mark).
  - A function overloading * for matrix multiplication (must check operation is possible) (1 mark).
  - A member function to return a matrix with the i^{th} row and j^{th} column deleted (1 mark).
  - **Challenge:** A function to calculate recursively the determinant of any square matrix (using the expansion in minors[1]- see e.g., mathworld)(2 mark).
- All of these functions must be demonstrated successfully in the main() function, see the next page for a list of requirements. Don't forget copy and move constructors.

[1] The (i, j) minor is the determinant of the matrix with the i^{th} row and j^{th} deleted.

A Matrix Class: Test Data

- Use the following matrices

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 9 & 8 & 7 \\ 4 & 2 & 6 \end{pmatrix}, \qquad B = \begin{pmatrix} 5 & 5 & 4 \\ 1 & 2 & 3 \\ 6 & 9 & 8 \end{pmatrix}, \quad C = \begin{pmatrix} 3 & 4 & 1 \\ 2 & 5 & 6 \end{pmatrix},$$

  to demonstrate all of the operations ($A + B$, $A - B$, $A \,.\, B$, $C \,.\, B$ and the impossibility of $B \,.\, C$)

- Show that you can read in the matrix A, and calculate the determinant of A and B.
- Use your copy constructor on A, modify the original matrix and output both it and its copy.
- Demonstrate the move constructor using A, show that the original matrix is in a valid state, and the new matrix has the expected form.

Marking Criteria (Negative Marks)

You lose marks if:

- The code generates any errors and warnings messages on compilation (-1 mark).
- Does not adhere to house style (-1 mark).
- The code does not adequately demonstrate the use and implementation of the class (-1 mark).
- For not submitting the .cpp file(s) (and .h files if used) (-9 marks). You can use a zip file to pack everything.
- A further penalty for late submission (after 7 pm, 24th March 2023).

Rubric: A Matrix Class

Grid View | List View

	Not at all	Partially	Completely
class contains a parametrized constructor and destructor using new and delete	**0** (0.00%)	**0.5** (5.55555%)	**1** (11.11111%)
class contains:(2) operator= function for deep copy with self-assignment check	**0** (0.00%)	**0.25** (2.77777%)	**0.5** (5.55555%)
class contains:(3) copy constructor for deep copy;	**0** (0.00%)	**0.5** (5.55555%)	**1** (11.11111%)
Efficiently coded move constructor and move assignment	**0** (0.00%)	**0.5** (5.55555%)	**1** (11.11111%)
class contains: (4) friend functions to overload operator<< and operator>> to produce sensible input and output for a matrix	**0** (0.00%)	**0.25** (2.77777%)	**0.5** (5.55555%)
Functions overloading + and - for matrix addition and subtraction (functions must check both matrices have same dimensions)	**0** (0.00%)	**0.5** (5.55555%)	**1** (11.11111%)
a function overloading * for matrix multiplication (must check operation is possible)	**0** (0.00%)	**0.5** (5.55555%)	**1** (11.11111%)
A member function to return a matri x with the ith row and jth column deleted	**0** (0.00%)	**0.5** (5.55555%)	**1** (11.11111%)
Correct recursive calculation of determinant using the minors (demonstrated in code)	**0** (0.00%)	**1** (11.11111%)	**2** (22.22222%)
The code generates no error messages on compilation	**-1** (-11.11111%)	**-0.5** (-5.55555%)	**0** (0.00%)
The code layout, presentation, comments, variable names are of sufficient quality	**-1** (-11.11111%)	**-0.5** (-5.55555%)	**0** (0.00%)
The code adequately demonstrates the use and implementation of the class	**-1** (-11.11111%)	**-0.5** (-5.55555%)	**0** (0.00%)