Семинарское занятие № 11. Динамическое выделение памяти Перегрузка операций. Исключения

*Структуры и классы, поля и методы, public, private, конструкторы и деструкторы*

***Указание.*** Многофайловый проект.

***Задание 1.***

1. Доработайте класс Array (<https://github.com/avelana/cpp-examples/tree/master/07-structures-classes/DemoArray>), сделав его шаблонным и хранящим коллекцию чисел. Определите все конструкторы, следуя правилу пяти.
2. Реализуйте соответствующие функции для работы c массивом, которые вызывают перегруженные операции над массивами:

1) '+', выполняющей сложение массивов.

2) '+', выполняющей поэлементное сложение массива и значения соответствующего интегрального типа.

3) '-', выполняющей вычитание двух массивов.

4) '-', выполняющей поэлементное уменьшение массива на значение соответствующего интегрального типа.

5) '\*', выполняющей поэлементное умножение массива на значение соответствующего интегрального типа.

6) '/', выполняющей поэлементное деление массива на значение соответствующего интегрального типа.

7) '=', выполняющей присваивание массива другому массиву, либо присваивание значения соответствующего интегрального типа всем элементам массива.

8) '+=', выполняющей поэлементное увеличение элементов массива на значение соответствующего интегрального типа.

9) '-=', выполняющей поэлементное уменьшение элементов массива на значение соответствующего интегрального типа.

10) '\*=', выполняющей поэлементное умножение элементов массива на значение соответствующего интегрального типа.

11) '/=', выполняющей поэлементное делением элементов массива на значение соответствующего интегрального типа.

12) '==', выполняющей сравнение двух массивов, либо сравнение всех элементов массива со значением соответствующего интегрального типа.

13) '!=', выполняющей сравнение двух массивов, либо сравнение всех элементов массива со значением соответствующего интегрального типа.

14) '<', выполняющей сравнение всех элементов массива (лексикографическое сравнение, по младшему индексу младшая позиция).

15) '>', выполняющей сравнение всех элементов массива (лексикографическое сравнение, по младшему индексу младшая позиция).

16) '<=', выполняющей сравнение всех элементов массива (лексикографическое сравнение, по младшему индексу младшая позиция).

17) '>=', выполняющей сравнение всех элементов массива (лексикографическое сравнение, по младшему индексу младшая позиция).

18) увеличения '++', выполняющей увеличение на 1 значения каждого элемента массива.

19) уменьшения '--', выполняющей уменьшение на 1 значения каждого элемента массива.

20) '<<', ‘>> ’выполняющей вывод/ввод в.из текстовый(ого) поток(а) значений элементов массива

21) других операций, представляющихся полезными при работе с рациональными дробями.

1. Обработку ошибок реализовать с помощью исключений.
2. Доработать класс ArrayService.
3. Разработайте юнит-тесты для тестирования класса Array.

***Задание 2.***

1. Разработайте класс Matrix, сделав его шаблонным и хранящим коллекцию коллекций чисел – аналог двумерного массива. Определите все конструкторы, следуя правилу пяти. Добавьте конструктор для единичной матрицы, диагональной матрицы.
2. Реализуйте соответствующие функции для работы c матрицами, которые вызывают перегруженные операции над матрицами:

1) '+', выполняющей сложение матриц.

2) '-', выполняющей вычитание двух матриц.

3) '\*', выполняющей поэлементное умножение матрицы на значение соответствующего интегрального типа.

4) '/', выполняющей поэлементное деление матрицы на значение соответствующего интегрального типа.

5) '+=', выполняющей сложение матриц.

6) '-=', выполняющей вычитание матриц

7) '\*=', выполняющей поэлементное умножение элементов матрицы на значение соответствующего интегрального типа.

8) '/=', выполняющей поэлементное делением элементов матрицы на значение соответствующего интегрального типа.

9) '==', выполняющей сравнение двух матриц.

10) '!=', выполняющей сравнение двух матриц.

11) '<<', ‘>> ’выполняющей вывод/ввод в.из текстовый(ого) поток(а) значений элементов матрицы.

1. Обработку ошибок реализовать с помощью исключений.
2. Реализовать класс MatrixService, который кроме функций, аналогичных перегруженным операторам класса Matrix, реализует функции: транспонирование матрицы, вычисление обратной матрицы, вычислить определитель, решение уравнения AX=B, поворот матрицы на 90, 180, 270, -90, -180, -270 градусов.
3. Разработайте юнит-тесты для тестирования класса Matrix.