Порядок выполнения работы

1. Разработать класс в соответствии с заданием. Проектирование класса рекомендуется начать с представления состояния класса, учитывающего заданные операции, а затем реализации конструкторов и метода вывода.

2. Для отладки и исчерпывающего тестирования других методов разработанного класса реализовать диалоговую программу, которая позволяет вводить параметры отлаживаемых методов. Для обработки ошибочных ситуаций использовать механизм исключительных ситуаций.

3. Повторить разработку класса, реализовав отдельные методы (там, где это оправданно), перегруженными операторами. Состав перегруженных операторов целесообразно согласовать с преподавателем.

4. Еще раз повторить разработку класса при условии, что память под массив необходимой длины выделяется динамически, во время выполнения программы (с помощью оператора new). Для хранения данных в экземпляре класса не должна использоваться лишняя память.

Дополнить интерфейс класса следующими возможностями:

· создание экземпляра класса с его инициализацией другим экземпляром класса (копирующий конструктор);

· переопределение экземпляра класса (с помощью перегруженного оператора присваивания).

5*. Разработать и реализовать прикладную программу, использующую класс, разработанный другим студентом. Задание для прикладной программы разработать самостоятельно и согласовать с преподавателем.

Требования

1) Проект на git'е. (обязательно)

2) Класс проектируется в трех вариантах: статическое состояние класса без перегрузки операторов, статическое состояние с перегрузкой операторов, динамическое состояние с перегрузкой операторов. (обязательно).

3) Использование коллекций из STL, умных указателей и т.п. только для "профессионалов", остальным рекомендуется дождаться задачи №4.

4) Корректность состояния класса, отсутствие избыточности, наличие необходимых конструкторов и деструктора, корректность сигнатуры методов, сохранение семантики перегружаемых операторов и корректность их сигнатуры, сохранение семантики работы с потоками ввода/вывода для перегружаемых операторов сдвига (обязательно).

6) Взаимное рецензирование кода до проверки кода преподавателем - code review, рецензирование и исправление кода по его итогам отражается на github / bitbucket, вплоть до LGFM; проверяется и оценивается преподавателем (опционально).

7) Разработка приложения, использующего класс, который разработан другим студентом (опционально).

8) Стандарт языка C++17 (рекомендуется), C++20 (при наличии). Допустим C++11 или C++14 (если почему-то нет C++17).

9) Плюс все то, что применимо из задач №1 и №2.