Проект №2: Защищенный массив

Задание

Требования

Студенты должны владеть следующими знаниями, умениями и навыками для выполнения задания.

- *Шаблоны* — понимание основ метапрограммирования с использованием шаблонов.
- *Расширение классов* с использованием внешних «дружественных» функций.
- *Исключительные ситуации*.

Цели и задачи

Основная цель работы — изучение практических основ создание шаблонов классов в языке C++.

Задачи для выполнения:

- изучить предложенные файлы и классы `SafeArray` и `Subject`;
- изучить подходы по разделению частей шаблон на описание заготовки класса и описания заготовки описанных методов класса;
- реализовать описание недостающих методов классов `SafeArray` и `Subject`.

Ожидаемые результаты

Студенты, успешно (и самостоятельно) выполнившие задание, получат практические навыки разработки шаблонов классов и расширения семантики стандартных типов данных (на примере потоков) для пользовательских классов.

Предпосылки

С-массивы не осуществляют проверку за выход за диапазон допустимых операций. Программист может случайно (или умышленно) осуществить чтение или запись элементов массива за границами выделенной памяти. При этом программа может продолжать работу без принаков наличия ошибок, что затрудняет их обнаружение и устранение. Однако наличие таких ошибок в программах недопустимо, и одним из способов их устранения является использование специальных структур данных, осуществляющих проверку выхода за границы допустимого диапазона.

Защищенный массив (safe array) — вид массива, который предоставляет доступ к своим элементам с проверкой за невыход за границы диапазона значений. Для поддержки стандартной семантики обращения к элементам такого массива, в C++ часто перегружается оператор `operator[]`.

Описание задания

Класс SafeArray

Класс, представляющий собой защищенный массив, представлен типом `xi::SafeArray`. Данный тип является шаблоном, параметр которого, `T`, представляет элементы, которые хранятся в массиве `SafeArray`.

Так как исполняемая часть класс `SafeArray` не может быть скомпилирована до момента конкретизации типа `T`, модуль, представляющий этот класс, содержит только заголовочные определения, которые разделены на два файла: *safearray.h* и *safearray.hpp*.

Файл *safearray.h* является «основным» заголовочным файлом, содержащим описание класса `SafeArray`. Этот файл включается модулем трансляции (cpp-файлом, прямо или косвенно), использующим этот тип данных. Файл *safearray.hpp* не является самостоятельным заголовочным файлом и не может быть включен в код где-либо, за исключением своего «родительского» файла *safearray.h* (см. директиву `#include` в конце файла).

Файл *safearray.hpp* включает дописание методов класса `SafeArray`, прототипы которых описаны в файле *safearray.h*. Обратите внимание, что так как дописание этих методов осуществляется вне исходного класса, заголовки методов включают информацию не только о «своем» классе, но и о параметрах шаблона.

Определение методов класса `xi::SafeArray` в файле *safearray.hpp* является относится к первой части данной работы.

Класс Subject

Класс `xi::Subject`, определяющий некоторую учебную дисциплину, является простой структурой данных, включающей информацию о записи об образовательном предмете со следующими атрибутами:

- название предмета;
- заголовок предмета;
- текстовое описание предмета (до 10 строк).

Информация о предметах дана в виде демонстрационного файла *res/raw/subjects.txt*. Тестирование соответствия программы спецификации также осуществляется с использованием этого файла.

Описание класса `xi::Subject` практически полностью представлено в заголовочном файле *subject.h* и не представляет затруднения. Внимания к себе требуют два перегруженных оператора `operator<<` и `operator>>`. Данные операторы получают первым операндом поток (вывода или ввода соответственно), а вторым — объект класса `Subject`. Операторы осуществляют вывод текстового представления объекта `Subject` в поток вывода или ввод этого объекта в виде текстовых строк из потока ввода. Использование базовых классов

потоков позволяет осуществлять ввода/вывод как с клавиатуры/в консоль, так и из файла/в файл.

Реализация оператор-функции `xi::Subject::operator<<()` предложено вместе с заданием, а реализация оператор-функции `xi::Subject::operator>>()` в соответствии со спецификацией относится ко второй части данной работы.

Указания по выполнению задания

Исходные файлы

- *safearray.h* — описание шаблона класса `xi::SafeArray`; файл можно при необходимости дополнять;
- *safearray.hpp* — дооописание методов класса `xi::SafeArray`; данный файл сдается в качестве результата работы;
- *subject.h* — описание класса `xi::Subject`; изменение этого файла заданием не предусматривается;
- *subject.cpp* — реализация оператор-функций для работа с потоками, описанными в классе `xi::Subject`; данный файл сдается в качестве результата работы;
- *main.cpp* — стартовый метод программы; изменение этого файла заданием не предусматривается.

Задачи для выполнения

1. Разобраться со структурой исходных кодов и предлагаемых сборочных проектов/решений.
2. Определить файлы, в которые необходимо внести рабочие изменения и внести их.
3. Сдать измененные файлы в виде zip-архива в систему LMS.

Представление результатов

Задание сдаётся в LMS в zip архиве, который содержит следующие файлы: *safearray.hpp*, *subject.cpp*, опционально — *safearray.h*.