Контрольные вопросы:

1. (8 б.) Что такое идиома RAII? Как расшифровывается эта аббревиатура?
2. (8 б.) Что такое конструктор, копирующий конструктор, деструктор, геттер, сеттер? Как определить конструктор и деструктор по умолчанию?
3. (8 б.) Почему поля класса принято делать приватными? Какой метод называется константным?
4. (8 б.) С какой целью проекты делают многофайловыми? Почему происходит разделение кода на .h файлы и .cpp файлы?
5. (8 б.) Какие три этапа проходит код при превращении в исполняемый файл? Расскажите, что в них происходит.

Упражнения:

1. (25 б.) Разработайте класс умного указателя на какую-либо переменную (например, double), соответствующий идиоме RAII, т.е. автоматически освобождающий память при выходе из области видимости. Класс должен:

    a. Иметь поле типа double*.
    b. Иметь два конструктора: один из них принимает уже имеющийся указатель double*, другой — переменную типа double и автоматически выделяет память.

    c. Иметь перегруженные операторы разыменования указателя (чтобы снаружи его можно было бы разыменовывать так же, как обычный указатель), вывода (чтобы выводился указатель — шестнадцатеричное число).

    d. Иметь копирующий конструктор, осуществляющий глубокое копирование, т.е. выделение новой переменной в памяти, которой присваивается копируемое значение, и создание нового указателя.

    e. Соответствовать идиоме RAII.
2. (25 б.) Замените структуру из первой задачи прошлого задания на класс. Реализуйте ее объявление в .h файле, а определение в .cpp файле. Постарайтесь учесть все аспекты проектирования классов, которые мы рассматривали на семинаре: публичный интерфейс, приватные данные, дружественные операторы и проч.

3. (25 б.) Разработайте класс, представляющий некоторую физическую величину и предоставляющий интерфейс для записи и чтения её значения в различных единицах измерения. Например, Вы можете разработать класс для энергии, хранить ее значение в джоулях, и реализовать функции-члены для записи и чтения значения с соответствующими преобразованиями в эргах и электрон-вольтах. Для хранения констант, используемых функциями-членами, рекомендуется использовать статические константы.

4. (25 б.) Смоделируйте ситуации, когда возникает ошибка линкера с обнаружением многократно определенного внешнего символа и ошибка линкера с неразрешенным внешним символом. Т.е. в первом случае это, например, функция, у которой после сборки проекта оказывается несколько определений, во втором случае — функция, у которой есть только объявление, но нигде нет определения. Должна получиться именно ошибка сборки, не ошибка компиляции.