



C++ - Модуль 07

Шаблоны C++

Резюме: Этот документ содержит упражнения Модуля 07 из модулей C++.

Версия : 6

Содержание

я	Введение	2
II	Основные правила	3
III	Упражнение 00: Начните с нескольких функций	5
IV	Упражнение 01: Итер	7
V	Упражнение 02: Мас с ив	8

Глава I

Введение

C++ — это язык программирования общего назначения, созданный Бьерном Страуструпом как расширение языка программирования C или «C с классами» (источник: [Википедия](#)).

Цель этих модулей — познакомить вас с объектно-ориентированным программированием. Это будет отправной точкой вашего путешествия по C++. Многие языки рекомендуются для изучения ООП. Мы решили выбрать C++, так как она является производным от вашего старого знакомого C. Поскольку это сложный язык, и для простоты ваш код будет соответствовать стандарту C++98.

Мы знаем, что современный C++ сильно отличается во многих аспектах. Так что, если вы хотите стать опытным разработчиком C++, вам решать, идти ли дальше после 42 Common Core!

Глава II

Основные правила

Компиляция

- Скомпилируйте свой код с помощью C++ и флагов `-Wall -Wextra -Werror`
- Ваш код все равно должен компилироваться, если вы добавите флаг `-std=c++98`.

Соглашения о форматировании и именованиях

- Каталог и упражнения будут называться следующим образом: `ex00`, `ex01`, ..., `exn`
- Назовите свои файлы, классы, функции, функции-члены и атрибуты, как требуется в рекомендациях.
- Пишите имена классов в формате `UpperCamelCase`. Файлы, содержащие код класса, будут всегда называться в соответствии с именем класса. Например: `ClassName.hpp/ClassName.h`, `ClassName.cpp` или `ClassName.tpp`. Затем, если у вас есть заголовочный файл, содержащий определение класса «BrickWall», обозначающего кирпичную стену, его имя будет `BrickWall.hpp`.
- Если не указано иное, каждое выходное сообщение должно заканчиваться символом новой строки. Символ отображается на стандартный вывод.
- Досвидания, Норминетт! В модулях C++ не применяется стиль кодирования. Вы можете следовать за своим любимым. Но имейте в виду, что код, который не могут понять ваши коллеги-оценщики, — это код, который они не могут оценить. Старайтесь писать чистый и читаемый код.

Разрешено/Запрещено

Вы больше не кодируете на C. Время C++! Следовательно:

- Вам разрешено использовать почти все из стандартной библиотеки. Таким образом, вместо того, чтобы придерживаться того, что вы уже знаете, было бы разумно использовать как можно больше C++-версий функций C, к которым вы привыкли.
- Однако вы не можете использовать никакую другую внешнюю библиотеку. Это означает, что C++11 (и производные формы) и библиотеки Boost запрещены. Также запрещены следующие функции: `*printf()`, `*alloc()` и `free()`. Если вы их используете, ваша оценка будет 0 и все.

- Обратите внимание, что если явно не указано иное, используемое пространство имен `<ns_name>` и ключевые слова друзей запрещены. В противном случае ваша оценка будет -42.
- Вам разрешено использовать STL только в Модуле 08. Это означает: никаких контейнеров (вектор/список/карта/и т. д.) и никаких алгоритмов (все, что требует включения заголовка `<algorithm>`) до тех пор. В противном случае ваша оценка будет -42.

Несколько требований к дизайну

- Утечка памяти происходит в C++. Когда вы выделяете память (используя новый/ключевое слово), вы должны избегать утечек памяти.
- От Модуля 02 до Модуля 08 ваши занятия должны быть оформлены в стандартном стиле. Каноническая форма, за исключением случаев, когда прямо указано иное.
- Любая реализация функции, помещенная в заголовочный файл (кроме шаблонов функций), означает 0 для упражнения.
- Вы должны иметь возможность использовать каждый из ваших заголовков независимо от других. Таким образом, они должны включать все необходимые зависимости. Однако вы должны избежать проблемы двойного включения, добавив защиту от повторного включения. В противном случае ваша оценка будет 0.

Прочитайте

- При необходимости вы можете добавить несколько дополнительных файлов (например, для разделения кода). Поскольку эти назначения не проверяются программой, не стесняйтесь делать это, пока вы не дадите окончательные файлы.
- Иногда рекомендации к упражнению кажутся короткими, но примеры могут показать требования, которые явно не прописаны в инструкции.
- Полностью прочитайте каждый модуль перед началом! Действительно, сделайте это.
- Клянусь Одном, клянусь Твором! Используйте свой мозг !!!




Вам придется реализовать много классов. Это может показаться утомительным, если только вы не умеете писать сценарии в своем любимом текстовом редакторе.



Вам предоставляется определенная свобода для выполнения упражнений. Однако соблюдайте обязательные правила и не ленитесь. Вы будете упускать много полезной информации! Не стесняйтесь читать о теоретических концепциях.

Глава III

Упражнение 00: Начните с нескольких функций

	Упражнение : 00
Начните с нескольких функций	
Каталог с дачи: ex00/ Файлы	
для с дачи: Makefile, main.cpp, что уг одно{h, hpp}	
Запрещенные функции: нет	

Реализуйте следующие шаблоны функций:

- `swap`: меняет местами значения двух заданных аргументов. Ничего не возвращает.
- `min`: сравнивает два значения, переданные в его аргументах, и возвращает наименьшее значение. Если два из них равны, то возвращается второй.
- `max`: сравнивает два значения, переданные в его аргументах, и возвращает наибольшее из них. Если два из них равны, то возвращается второй.

Эти функции можно вызывать с любым типом аргумента. Единственное требование состоит в том, что два аргумента должны иметь один и тот же тип и должны поддерживать все операторы сравнения.



Шаблоны должны быть определены в файлах заголовков.

Выполнение следующего кода:

```
ИНТ    основной (недействительный) {

    ИНТ а = 2;
    ИНТЕРВАЛ б = 3;

    // поменять местами (а, б);
    std::cout << "а = << а << ", б = << б << std::endl;
    std::cout << "min(а, б) = << ::min(а, б) << std::endl;
    std::cout << "max(а, б) = << ::max(а, б) << std::endl;

    std::string с = "цепочка1";
    std::string d = "цепочка2";

    // поменять местами (с, d);
    std::cout << "с = << с << ", d = << d << std::endl;
    std::cout << "min(с, d) = << ::min(с, d) << std::endl;
    std::cout << "max(с, d) = << ::max(с, d) << std::endl;


    вернуть 0;
}
```

Должен выводиться :

```
а = 3, б = 2
мин (а, б) = 2
макс (а, б) = 3
с = цепочка2, d = цепочка1
min(с, d) = chaine1
max(с, d) = chaine2
```

Глава IV

Упражнение 01: Итер

	Упражнение : 01
Итер	
Каталог с дачи: ex01/ Файлы	
для с дачи: Makefile, main.cpp, iter.{h, hpp}	
Запрещенные функции: нет	

Реализуйте `iter` шаблона функции, который принимает 3 параметра и ничего не возвращает.


- Первый параметр — это адрес массива.
- Второй — длина массива.
- Третий — это функция, которая будет вызываться для каждого элемента массива.

Включите файл `main.cpp`, содержащий ваши тесты. Предоставьте достаточно кода для создания тестового исполняемого файла.

Ваш шаблон функции `iter` должен работать с любым типом массива. Третий параметр может быть созданным шаблоном функции.

Глава V

Упражнение 02. Мас с ив

	Упражнение : 02
Множество	
Каталог с дачи: ex02/	
Файлы для с дачи: Makefile, main.cpp, Array.{h, hpp} и необязательный файл: Array.tpp	
Запрещенные функции: нет	

Разработайте шаблон класса `Array`, который содержит элементы типа `T` и реализует следующее поведение и функции:

- Конструктор без параметров: создает пустой массив.
- Конструктор с целым числом без знака в качестве параметра: создает массив из n элементов. Инициализируется по умолчанию.
Совет: Попробуйте скомпилировать `int *a = new int();` затем отобразите `*a`.
- Построение операторов копирования и присваивания. В обоих случаях изменение либо исходный массив или его копия после копирования не должны влиять на другой массив.
- Вы **ДОЛЖНЫ** использовать оператор `new[]` для выделения памяти. Превентивное выделение (предварительное выделение памяти) запрещено. Ваша программа никогда не должна обращаться к неопределенной памяти.
- Доступ к элементам можно получить с помощью оператора индекса `[]`.
- При доступе к элементу с помощью оператора `[]`, если его индекс выходит за пределы, генерируется исключение `std::exception`.
- Функция-член `size()`, которая возвращает количество элементов в массиве. Эта функция-член не принимает параметров и не должна изменять текущий экземпляр.

Как обычно, убедитесь, что все работает должным образом, и сдайте файл `main.cpp`, содержащий ваши тесты.