

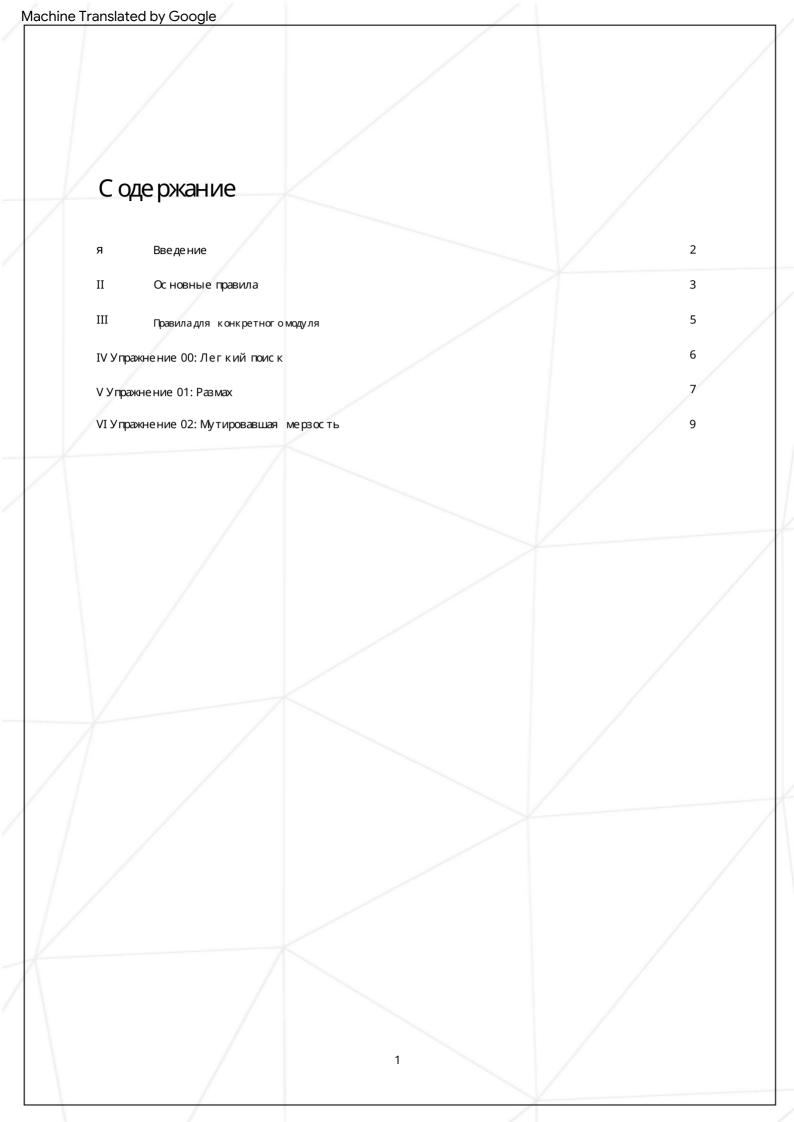


С++ - Модуль 08

Шаблонные контейнеры, итераторы, алг оритмы

Резниме: Этот документ с одержит у пражнения Модуля 08 из модулей C++.

Версия:7



Глава I

Введение

C++- э то я зык прог раммирования общего назначения, с озданный Бьерном Страу стру пом как рас ширение я зыка прог раммирования С или «С с клас с ами» (источник: Википедия).

Цель э тих модулей — познакомить вас с объектно-ориентированным программированием. Это будет отправной точкой вашего путешествия по С++. Многие я зыки рекомендуются для изучения ООП. Мы решили выбрать С++, так как он я вля ется производным от вашего старого знакомого С. Поскольку э то сложный я зык, и для простоты вашкод будет соответствовать стандарту С++98.

Мы знаем, что с овременный С++ с ильно отличается во мног их ас пектах. Так что, ес ли вы х отите с тать опытным разработчик ом С++, вам решать, идти ли дальше пос ле 42 Common Core!

Глава II

Ос новные правила

Компиля ция

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

С ог лашения о форматировании и именовании

• Каталог и у пражнений бу дут называтьс я с леду ющим образом: ex00, ex01, ...,

exn

- Назовите с вои файлы, клас с ы, функц ии, функц ии-члены и атрибуты, как требуется в рекомендац ии.
- Пишите имена клас с ов в формате UpperCamelCase. Файлы, с одержащие код клас с а, бу дут всег да называться в соответствии с именем клас с а. Например:

 СlassName.hpp/ClassName.h, ClassName.cpp или ClassName.tpp. Затем, если у вас есть заголовочный файл, с одержащий определение клас с а «BrickWall», обозначающего кирпичную с тену, его имя бу дет BrickWall.hpp.
- Если не указано иное, каждое вых одное сообщение должно заканчиваться с имволом новой строки. с имвол и отображается на стандартный вывод.
- Досвидания, Норминетт! В модуля х С++ не применя ется стиль кодирования. Вы можете следить за своим любимым. Но имей те в виду, что код, который не могут поня ть ваши коллеги-оценщики, э то код, который они не могут оценить. Старай тесь писать чистый и читаемый код.

Разрешено/Запрещено

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

- Вам разрешено ис пользовать почти все из стандартной библиотеки. Таким образом, вместо того, чтобы придерживаться того, что вы уже знаете, было бы разумно ис пользовать как можно больше С++-версий функций С, к которым вы привыкли.
- Однако вы не можете ис пользовать никаку юдруг у ювнешнююбиблиотеку. Это означает, что C++11 (и производные формы) и библиотеки Boost запрещены. Также запрещены с ледующие функц ии: *printf(), *alloc() и free(). Ес ли вы их ис пользуете, ваша оц енка будет 0 и вс е.

- Обратите внимание, что ес ли я вно не указано иное, ис пользуемое пространство имен <ns_name> и ключевые слова друзей запрещены. В противном случае ваша оценка будет -42.
- Вам разрешено ис пользовать STL только в Модуле 08. Это означает: никаких контей неров (вектор/с пис ок/карта/и т. д.) и никаких алг оритмов (все, что требует включения заголовка <algorithm>) до тех пор. В противном с лучае ваша оценка будет -42.

Несколько требований к дизайну

- Утечка памя ти проис х одит и в C++. Ког да вы выделя ете памя ть (ис пользуя новый ключевое с лово), вы должны избег ать утечек памя ти.
- От Модуля 02 до Модуля 08 ваши заня тия должны быть оформлены в православном стиле. Каноническая форма, за исключением случаев, ког да пря мо указано иное.
- Любая реализация функции, помещенная в заголовочный файл (кроме шаблонов функций), означает 0 для упражнения.
- Вы должны иметь возможность ис пользовать каждый из ваших заголовков независ имо от друг их. Таким образом, они должны включать все необх одимые им завис имости. Однако вы должны избежать проблемы двой ного включения, добавив защиту включения. В противном случае ваша оценка будет 0.

Прочтименя

- При необх одимости вы можете добавить нес колько дополнительных файлов (например, для разделения кода). Пос кольку э ти назначения не проверя югся программой, не стес ня йтесь делать э то, пока вы с даете обя зательные файлы.
- И ног да рекомендац ии к у пражнению кажутся короткими, но примеры могут показать требования, которые я вно не прописаны в инструкция x.
- Полностью прочитайте каждый модуль перед началом! Действительно, с делай э то.
- Клянусь Одином, клянусь Тором! Используйс вой мозг!!!



Вам придется реализовать много классов. Это может показаться утомительным, еслитольковы не умеете писать с ценарии в своем любимом текстовом редакторе.



Вам предоставля ется определенная свобода для выполнения упражнений. Однако соблюдайте обя зательные правила и не ленитесь. Ты бы упускаюмного полезной информации! Не стесня йтесь читать о теоретические концепции.

Глава III

Правила для конкретного модуля

Вы заметите, что в э том модуле у пражнения можно решать БЕЗ с тандартных контей не ров и БЕЗ с тандартных алг оритмов.

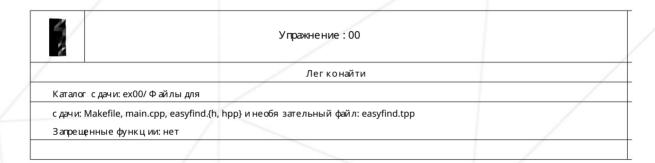
Однако именно их ис пользование и я вля етс я цельюданного Модуля. Вам разрешено ис пользовать STL. Да, вы можете ис пользовать Контейнеры (вектор/с пис ок/карту/ит. д.) и Алгоритмы (определенные в заголовке <algorithm>). Более того, вы должны ис пользовать их как можно чаще. Таким образом, с делайте все возможное, чтобы применить их везде, где э то уместно.

В противном с лучае вы получите очень плох уюоценку, даже если вашкод работает должным образом. Пожалуйста, не ленитесь.

Вы можете определить с вои шаблоны в файлах заголовков, как обычно. Или, если вы хотите, вы можете напис ать с вои объя вления шаблонов в файлах заголовков и напис ать их реализации в файлах .tpp. В любом случае файлы заголовков я вля югся обя зательными, а файлы .tpp — необя зательными.

Глава IV

Упражнение 00: Легкий поиск



Первое простое у пражнение — > то с пос об начать с правой ног и.

Напишите шаблон функции easyfind, который принимает тип Т. Он принимает два параметра. Первый имеет тип Т, а второй — целое чис ло.

Предполагая, что Тявляется контей нером целых чисел, эта функция должна найти первое вхождение второго параметра в первый параметр.

Ес ли вх ождение не най дено, вы можете либо с оздать ис ключение, либо вернуть значение ошибки. на ваш выбор. Ес ли вам ну жно вдох новение, проанализиру йте, как ведут с ебя с тандартные контей неры.

Конечно, внедрите и с дайте с вои с обственные тесты, чтобы у бедиться, что все работает так, как ожидалось.



Вам не нужно обрабатывать ас с оциативные контейнеры.

Глава V

Упражнение 01: размах



Упражнение: 01

Ov parupari

Каталог с дачи: ex01/

Ф айлы для с дачи: Makefile, main.cpp, Span.{h, hpp}, Span.cpp

Запрещенные функции: нет

Разработайте клас с Span, который может х ранить мак с иму м N ц елых чис ел. N — переменная типа int без знака и бу дет единс твенным параметром, передаваемым конструктору.

Этот клас с будет иметь функциючлен с именем addNumber() для добавления одного числа в Span. Он будет использоваться для его заполнения. Любая попытка добавить новый элемент, если уже с ох ранено N элементов, должна вызвать исключение.

Затем реализуйте две функц ии-члена: shortestSpan() и longestSpan().

Они с оответственно най дут к ратчай ший промежуток или с амый длинный промежуток (или рас стоя ние, если х отите) между всеми с ох раненными числами и вернутего. Если числа не с ох ранены или есть только одно, диапазон не может быть най ден. Так им образом, с генерируйте исключение.

Конечно, вы будете пис ать с вои с обственные тесты, и они будут намног о более тщательными, чем приведенные ниже. Проверьте с вой Span x отя бы с минимум 10 000 номеров. Больше было бы еще лучше.

Запускэ того кода:

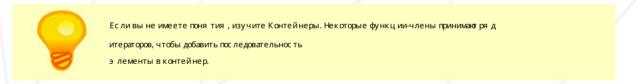
```
0 {
    Диапазон sp = Диапазон (5);
    sp.addNumber (6);
    sp.addNumber (3);
    sp.addNumber (17);
    sp.addNumber (9);
    sp.addNumber (11);
    std::cout << sp.shortestSpan() << std::endl; std::cout << sp.longestSpan() << std::endl;
    Be рнуть 0;
}
```

Должен выводиться:

```
$> ./ex01 2

14
$>
```

И последнее, но не менее важное: было бы замечательно заполнить ваш Span, ис пользуя ряд итераторов. Делать тыся чи вызовов addNumber() так раздражает. Реализуйте функциючлен, чтобы добавить много номеров в ваш Span за один вызов.



Глава VI

Упражнение 02: Мутировавшая мерзость



Теперь пора перейти к более с ерьезным вещам. Давайте разработаем что-нибу дь с транное.

Контейнер std::stack очень у добен. К с ожалению э то один из немног их контейнеров STL, который НЕ я вля ется итерируемым. Это очень плох о.

Но зачем нам э то принимать? Ос обенно, ес ли мы позволим себе зарезать исх одный стек для создания недостающих функций.

Чтобы ис править э ту нес праведливость, вы должны с делать контей нер std::stack итерируемым.

Напишите клас с MutantStack . Он бу дет реализован в терминах std::stack. Он бу дет предлаг ать вс е с вои функц ии-члены, а также дополнительну юфункц ию итераторы.

Конечно, вы будете пис ать и с давать с вои с обственные тесты, чтобы убедиться, что все работает как положено.

```
С++ - Модуль 08
```

Шаблонные контейнеры, итераторы, алг оритмы

Найдите тестовый примерниже.

```
Interpretation of the content of the
```

Ес ли вы запустите его в первый раз со с воим MutantStack и во второй раз замените MutantStack, например, на std::list, два вы вода должны быть одинак овыми. Конечно, при тестировании другого контей нера обновите приведенный ниже код с оответствующими функция ми-членами (push() может с тать push_back()).