Семинар 1. Класс «Точка»

С. А. Шершаков

11 сентября 2017 г.

Ред. 1.1 от 11.09.2017 г.

В документе представлен 1-й цикл заданий для самостоятельного выполнения студентами курса «Алгоритмы и структуры данных» и методические рекомендации по их выполнению. Задание посвящено изучению основных конструкция языка C++, а также особенностям построения проектов для разных платформ и разными средствами сборки на основе единой кодовой базы. Также рассматриваются инструмент автодокументирования кода Doxygen и вопросы unit-тестирования C++-программ с помощью библиотеки gtest.

1 Требования, цели и ожидаемые результаты

Чтобы перейти непосредственно к шагам, необходимым для выполнения задания, обратитесь к разделу 4.

1.1 Требования

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

- Основы разработки программ на языке C++.
- Основы IDE Microsoft Visual Studio 2013.
- Методические рекомендации к практическим занятиям по дисциплине «Алгоритмы и структуры данных».

1.2 Цели и задачи

Основная цель работы — овладевание основными языковыми и инструментальными подходами к написанию программ на языке C++; рассмотрение основных отличий разработки на языке C++ от языков C# и Java.

Основные темы: принципы создания C++-проектов с помощью Visual Studio 2013 и CMake 3; модули языка, заголовочные файлы и модули трансляции; базовая система ввода/вывода; основные конструкции языка; память и указатели.

Связанные задачи: автодокументирование кода с использование инструмента Doxygen[1], unit-тестирование с использованием библиотеки gtest[2].

[1] `http://www.doxygen.org`

[2] `https://github.com/google/googletest`

Задачи для выполнения в рамках самостоятельной работы:

- Изучение заголовочных файлов классов `Point` и `PointArray` (модуль `xipoint.h`).
- Реализация объявленных методов классов `Point` и `PointArray` в модуле `xipoint.cpp` на основе спецификации методов, ориентируясь на их спецификацию (см. Doxygen-документацию) и настоящее руководство.
- [*Опционально*] протестировать полученную реализацию на множестве предложенных unit-тестов[3].

[3] Студенты получают часть unit-тестов (30–40 % от полного набора).

1.3 Ожидаемые результаты

Студенты, успешно[4] выполнившие задание, получат навыки разработки C++-программ, включающих базовые управляющие структуры, массивы, разделение модулей на интерфейсную и реализационную части, использование препроцессора языка, тестирования разработанных методов на предложенном наборе unit-тестов, автодокументирования кода в формате Doxygen.

[4] И самостоятельно.

2 Описание задания

В рамках самостоятельной работы студентам предлагается разработать недостающие методы классов `Point` и `PointArray`, интерфейсное описание и спецификация которых представлены в файле `xipoint.h`. Разработанные методы помещаются в подготовленный файл `xipoint.cpp`.

Далее представлена краткая информация по основным модулям/типам программного кода. Дополнительная информация может быть получена из автосоздаваемой документации Doxygen[5] или из комментариев к файлам/типам.

[5] Предоставляется в виде архива с каталогом `/html`, из которого необходимо открыть файл `/html/index.html` в любом браузере.

2.1 Класс `Point`

Класс `Point` описывает точку с двумя целочисленными координатами, хранимыми в защищенных полях. Класс содержит конструктор инициализации обеих координат, получающий параметры по умолчанию, что позволяет его использовать только для одной координаты `x` либо вообще инициализировать без параметров, то есть использовать в качестве конструктора по умолчанию.

Класс содержит методы для установки и получения значений отдельных координат (с перегруженными константными вариантами).

2.2 Класс `PointArray`

Представляет собой массив точек, представленных объектами класса `Point`. Класс самостоятельно управляет памятью подлежащего С-массива, предоставляя пользователю интерфейс для добавления, удаления индивидуальных точек, получения доступа к ним в виде указателей. Размер массива — число хранимых объектов — доступен через вызов метода `getSize()`.

Класс содержит несколько конструкторов, позволяющих по-разному конструировать объект. При каждом изменении размера подлежащего массива (добавление и удаление элементов) вызывается служебный (защищенный) метод `resize()`[6]. Очищение массива осуществляется методом `clear()`. При удалении объекта деструктор освобождает память,

[6] Данный метод создает новый массив запрашиваемой длины, копирует в него элементы из предыдущего массива (максимально возможное их число), удаляется предыдущий массив и перезаписывает указатели на массив и его размер. Метод осуществляет возможно неоптимальное перераспределение памяти при каждом изменении массива, однако в рамках данной задачи оптимальность не является изучаемым критерием.

занимаемую массивом, осуществляя тем самым политику избежания «утечки памяти».

3 *Структура проекта*

Студенты получают для работы проект со следующей структурой:

- `/docs` — каталог с документацией, включающий настоящее руководство;
- `/docs/doxydoc` — каталог с автоматически генерируемой документацией Doxygen;
- `/docs/doxydoc/out/html` — каталог с выходными файлами автоматически генерируемой документации в формате html;
- `/docs/doxydoc/Doxyfile.example` — пример файла настроек системы документации Doxygen для данного проекта;
- `/docs/doxydoc/doxyit.cmd.example` — пример командного файла для ОС Windows запуска автоматического построения документации;
- `/docs/doxydoc/readme.md`[7] — файл с заметками к содержимому каталога;
- `/sol` — каталог с решениями/проектами для MS Visual Studio 2013 и системы сборки CMake[8];
- `/src` — исходные коды проекта;
- `/tests/gtest` — тесты в формате gtest;
- `/tests/gtest/src` — исходные файлы тестов;
- `/tests/gtest/sol` — проекты тестового окружения для MS Visual Studio 2013 и CMake;
- `/tests/gtest/readme.md` — дополнительная полезная информация;
- `/readme.md` — самая общая и самая полезная дополнительная информация по проекту.

Проект поставляется в виде одного или нескольких (возможно заархивированных) файлов, что определяется конкретным способом представления задачи с использованием автоматизированной системы обучения (LMS).

[7] Файлы с расширением `.md` легко читаются в обычном текстовом редакторе. Многие современные агрегаторы кода, такие как Bitbucket и GitHub транслируют его на лету в HTML и в целом используют как основной формат «быстрой документации». При необходимости форматированного просмотра на локальном компьютере имеется возможность установить, например, плагин для Firefox (для Хрома, говорят, тоже что-то есть) и использовать браузер для просмотра.

[8] `https://cmake.org/`

4 *Указания по выполнению задания*

Выполнение задачи сводится к следующей последовательности[9].

- Ознакомиться с настоящим руководством-заданием.
- Получить исходные файлы для работы[10].
- Ознакомиться с предлагаемыми для работы исходными файлами и автодоксидокументацией к ним.
- Выполнить реализацию недостающих методов классов `Point` и `PointArray` в файле `/src/xipoint.cpp`.
- Протестировать (по желанию) реализацию на предложенном наборе тестов (файлы `/tests/gtest/src/point_test.cpp` и `point_array_test.cpp`).

[9] Рекомендованная последовательность.

[10] С использованием системы LMS: `http://lms.hse.ru`

- Загрузить результаты работы в виде единого файла в ассоциированный проект системы LMS до крайнего срока, указанного в сводке проекта.

При выполнении задачи не подразумевается изменение каких-либо файлов, не перечисленных выше. В случае, если возникает указанная необходимость, при сдаче такого файла необходимо сопроводить его комментариями в виде файла `student_notes.md`, который следует поместить в тот же каталог, что и «нормальнонесдаваемый» файл. В таком случае все файлы помещаются в единый архив формата zip, который подлежит сдаче в установленном порядке. В `student_notes.md` необходимо аргументировать потребность к изменению файла[11].

4.1 Сдаваемые файлы

- `/src/xipoint.cpp` — реализация основной задачи.

[11] Примером такой необходимости является выявленная ошибка в исходном задании/файлах, не позволяющая решить задачу. В случае, если будет признано, что такой необходимости в действительности не было, это может повлечь за собой снижение оценки.