Задание 1.1

CLion

Оглавление:

Общая характеристика:

CLion — это многофункциональная IDE. В ней вы можете не только писать на C и C++, но и заниматься веб-разработкой на HTML, CSS, JavaScript, XML. Некоторые другие языки доступны в виде плагинов (например, Lua). IDE интегрирована с многими популярными системами контроля версий и баг-трекерами. В дальнейших версиях мы планируем добавлять поддержку фреймворков для юнит-тестирования. Помним о самом главном CLion сейчас поддерживает два компилятора: GCC и Clang. Для отладки приложения предполагается использовать GDB 7.8. Встроенный отладчик позволит пройти программу по шагам, остановившись на точках остановки, посмотреть значения переменных и даже изменить их на ходу:

Функции:

1. Авто дополнение

CLion помогает писать код быстрее с помощью функции авто дополнения. Начните вводить ключевое слово, имя переменной, функции или класса — CLion предложит список подходящих вариантов авто дополнения. Чтобы сэкономить еще больше времени, вы можете ввести только заглавные буквы имени или даже любую его часть. Кроме того, IDE предлагает варианты авто дополнения для символов из внешних библиотек и фреймворков, которые используются в вашем проекте. Постфиксное авто дополнение для C и C ++ подставляет нужный код с обеих сторон введенного выражения. Оно поможет заключить выражение в часто используемую языковую конструкцию или передать его в качестве первого аргумента свободной функции.

2. Стиль кода

CLion помогает поддерживать единый стиль оформления кода и соблюдать настройки форматирования. Настройки применяются автоматически, когда вы пишете код, а также с помощью действия Reformat Code (Ctrl+Alt+L). Стиль кода настраивается в меню Editor | Code Style. Вы можете настроить общую конфигурацию или опции для конкретного языка. Настройки стиля оформления кода определяют правила выравнивания, расстановки пробелов, табуляции и отступов, генерации кода и использования пустых строк. Если настройка стиля влияет на код, CLion предложит предварительный просмотр, выделив строку, затронутую изменением.

3. Расширенные настройки

CLion позволяет не применять некоторые настройки форматирования к коду в выбранных файлах. Если отступы в файле отличаются от действующих настроек форматирования, CLion предупредит об этом при открытии файла. Вы можете выбрать принудительное применение настроек стиля или оставить существующие отступы. Эту функцию можно включить/отключить в настройках Editor | Code Style... CLion позволяет изменить установленные настройки для отдельного фрагмента кода. Выделите нужный фрагмент кода и примените быстрое исправление Adjust code style settings (Alt+Enter).

4. Подсветка и конфигурации

Если в коде встречаются блоки, зависящие от флагов компиляции и переменных таргета CMake, CLion сумеет правильно подсветить эти части кода — просто выберите конфигурацию Run/Debug, и IDE автоматически переключит resolve-контекст. Вы также можете воспользоваться ручным переключением resolve-контекста в правом нижнем углу редактора.

5. Breadcrumbs

маленьких маркеров внизу редактора — вы можете легко перемещаться по коду. Такие маркеры отображаются для пространств имен, классов, структур, функций и лямбд.

6. Предопределенные стили оформления кода

Вы можете выбрать предопределенный стиль оформления кода и применить его к вашему проекту. IDE поддерживает стандартные стили Google, LLVM и др. Выберите подходящий стиль в меню Set from... | Predefined Style. Стили применяются к настройкам форматирования, правилам именования C/C++, стилям header guards.

7. Правила именования

CLion соблюдает заданную схему именования. Выбранные настройки будут использоваться во всех действиях IDE, включая:

a) Авто дополнение
b) Генерацию кода
c) Рефакторинг кода
d) Применение быстрых исправлений

На вкладке Naming Convention в настройках Settings | Editor | Code Style | C/C++ можно выбрать один из предопределенных стилей или вручную настроить свой.

8. Подсказки параметров

Благодаря подсказкам имен параметров вам не придется переключаться на сигнатуру функции во время изучения вызова функции. В свою очередь, это помогает улучшить читаемость вашего кода. Для вызовов функций, лямбда-выражений, конструкторов, списков инициализаторов и макросов CLion показывает имена параметров для переданных аргументов. Подсказки работают, если аргументом является литерал или выражение с более чем одним операндом.

9. Подсказки типов

Для улучшения читаемости кода в CLion показываются подсказки с выведенными типами. Эти подсказки полезны для переменных типа auto, в конструкциях структурного связывания и для типов значений, возвращаемых из лямбд. Включить или отключить подсказки тех или иных типов можно в меню Settings | Editor | Inlay Hints | C/C++ или прямо в контекстном меню подсказки.

10. ClangFormat

В качестве альтернативного средства форматирования кода CLion поддерживает инструмент ClangFormat. Вы можете включить его в настройках и использовать для форматирования кода как в текущем проекте, так и во всех проектах CLion. Если в проекте будет обнаружен конфигурационный файл. clang-format, CLion предложит переключиться на ClangFormat.

11. Выделение фрагментов кода и комментарии

Чтобы быстро выделить блок кода, поместите в него курсор и нажмите Ctrl+W необходимое количество раз, чтобы расширить выделение до конца выражения, строки, логического блока и так далее. Аналогичным образом, чтобы снять выделение с логической части, нажмите Ctrl+Shift+W столько раз, сколько нужно. Для выделенных фрагментов доступно блочное (Ctrl+Shift+/) и однострочное комментирование (Ctrl+/). А чтобы добавить комментарий к одной строке, ее даже не нужно выделять — просто

12. Многокурсорность

CLion позволяет делать несколько вещей одновременно. В режиме много курсорного редактирования вы можете вносить изменения в файл в нескольких местах одновременно. Выберите места для редактирования (нажмите Alt+Shift и поместите курсор в нужные места) или просто добавьте пару следующих совпадений в выделение с помощью Alt+J. Чтобы удалить совпадение из выделения, нажмите Alt+Shift+J. Завершив редактирование, нажмите Esc, чтобы вернуть курсор в обычный режим.

13. Быстрый просмотр определений

Посмотреть реализацию или объявление функции можно прямо в редакторе, не переходя на другой файл. Просто поместите курсор на нужный символ и нажмите Ctrl+Shift+I, чтобы открыть всплывающее окно Quick Definition.

14. Быстрый просмотр документации

В CLion есть специальное окно для быстрого просмотра документации (Quick Documentation popup, вызывается нажатием Ctrl+Q), в котором отображается документация для выбранного класса, функции, переменной, параметра или макроса. Окно Quick Documentation показывает результаты подстановки в макросах, выведенные типы для переменных, объявленных как auto, и указатели на ссылочные

типы. В нем также доступен предварительный просмотр документации в формате Doxygen.

15. Информация о параметрах

CLion придет на помощь, если вы не уверены, какие параметры принимает функция. Вызовите информацию о параметрах функции (Ctrl+P), чтобы увидеть все доступные сигнатуры и параметры функции. При редактировании параметров CLion скроет все неподходящие сигнатуры.

16. Реорганизация кода

В редакторе CLion можно перемещать вверх/вниз по коду целые строки и блоки кода. Чтобы переместить одну строку, не нужно выделять ее — просто нажмите Alt+Shift+Up/Down. Для перемещения выделенного блока кода используйте сочетание клавиш Ctrl+Shift+Up/Down.

17. Действие Unwrap/Remove

Чтобы безопасно удалять фрагменты в сложном коде со множеством вложенных операторов, используйте действие Unwrap/Remove… (Ctrl+Shift+Delete). IDE предлагает разные варианты преобразований в зависимости от того, где находится курсор, и умеет вынимать код из управляющих конструкций if, else, for, while, do..while и for.

18. Автоматическое импортирование

Когда вы используете символ, который еще не был импортирован, CLion выполнит поиск и предложит добавить нужную директиву #include или сделает это автоматически.

Необходимое программное и аппаратное обеспечение:

Windows:

1. 64-битная версия Windows 8, 10, 11.
2. Не менее 2 ГБ ОЗУ.
3. 2,5 ГБ свободного места на диске.
4. Разрешение экрана — не менее 1024x768 пикселей

MacOS:

1. macOS 10.14 или более поздняя версия
2. Не менее 2 ГБ свободной оперативной памяти, рекомендуется использовать устройства с 8 ГБ RAM
3. 2,5 ГБ свободного места на диске, рекомендуется использование SSD
4. Разрешение экрана — не менее 1024x768 пикселей

Linux:

1. Среда GNOME или KDE
2. Не менее 2 ГБ свободной оперативной памяти, рекомендуется использовать устройства с 8 ГБ RAM
3. 2,5 ГБ свободного места на диске, рекомендуется использование SSD
4. Разрешение экрана — не менее 1024x768 пикселей

Функции, связанные с разработкой ПО:

1. Создание проекта

a) Если в настоящее время в CLion нет открытого проекта, нажмите «Новый проект» на экране приветствия. В противном случае выберите Файл | Новый проект в главном меню.
b) В открывшемся диалоговом окне «Новый проект» выберите целевой тип вашего проекта (исполняемый файл или библиотека) и язык, который будет использоваться (чистый C или C++).

CLion сгенерирует файл CMakeLists.txt верхнего уровня на основе предоставленных настроек. Дополнительные сведения см. в шаблонах файлов CMakeLists.txt.

Создание проекта CMake из исходников

Для работы с исходными кодами, отличными от CMake, в CLion их можно преобразовать в структуру проекта CMake.

a) В главном меню выберите Файл | Откройте и выберите корневую папку проекта.
b) Откройте исходный файл в редакторе.
c) **Если файла CMakeLists.txt** верхнего уровня нет, CLion предложит его создать:
d) В диалоговом окне «Создать CMakeLists.txt» укажите следующее:

Выбрать файлы проекта — выберите файлы для импорта в качестве файлов проекта. Используйте флажки подкаталогов, чтобы импортировать все их содержимое, или снимите флажки, чтобы импортировать содержимое выборочно.

User Include Directorys — выберите каталоги, которые будут включены в проект и указаны в команде CMake include_directories. CLion включает каталог автоматически, если он содержит хотя бы один заголовочный файл, если он называется **include** или если в нем есть подкаталоги, содержащие только заголовочные файлы.

Обратите внимание, что каталоги, не выбранные на панели «Выбрать файлы проекта», не представлены в списке «Пользовательские включаемые каталоги» — сначала выберите их, а затем в списке появятся доступные включаемые каталоги.

2. Кодирование
Действие Create from usage.

В CLion вы можете создавать функции, переменные, члены класса или даже целые классы, прежде чем они будут объявлены. IDE подсветит элемент красным и предложит быстрое исправление. Нажмите Alt+Enter, чтобы добавить соответствующее объявление. Для функций CLion также определяет тип возвращаемого значения и типы параметров.

Меню Generate CLion

Позволяет сэкономить время на печати кода, предоставляя различные варианты код генерации в меню Generate (Alt+Insert). Сгенерировать функции get/set, конструкторы/деструкторы класса, операторы сравнения, равенства и печати (stream output) можно в один клик. CLion позволяет выбирать поля класса, которые необходимо использовать во время генерации, а также указывать различные опции: нужно ли создавать новые конструкции как члены класса, производить ли генерацию на месте, где стоит курсор, использовать ли std::tie в реализации и т.д. В случае если некоторые операторы уже есть в коде, CLion предложит добавить отсутствующие или полностью заменить существующие.

Рефакторинги CLion

Помогает поддерживать высокое качество кода, предоставляя набор надежных рефакторингов. Во время рефакторинга CLion безопасно производит преобразования по всей выбранной области. Чтобы увидеть все рефакторинги, доступные в текущем месте, используйте меню Refactor This... (вызывается нажатием Ctrl+Alt+Shift+T). Список рефакторингов включает:

a) Rename (Shift+F6) переименует символ и автоматически обновит все необходимые ссылки.
b) Change Signature (Ctrl+F6) поможет добавить/удалить параметры функции и поменять их порядок, а также изменить тип результата и имя функции, при этом все использования будут обновлены автоматически.
c) Inline Ctrl+Alt+N уберет лишнюю переменную или вызов функции, заменив их на непосредственное значение/код функции.
d) Extract (Function Ctrl+Alt+M, Typedef Ctrl+Alt+K, Variable Ctrl+Alt+V, Parameter Ctrl+Alt+P, Define Ctrl+Alt+D, Constant Ctrl+Alt+C, Lambda Parameter) — CLion проанализирует блок кода, для которого вы вызываете рефакторинг, обнаружит входные и выходные переменные и использования выбранного выражения в коде, чтобы заменить их на новую сущность по вашему выбору.
e) Pull Members Up/Push Members Down помогут с безопасной реорганизацией иерархии классов в вашем проекте.

Intention actions

Помогают применять автоматические изменения к коду, чтобы улучшить его и упростить выполнение рутинных задач. С их помощью можно добавлять поля класса и инициализаторы в конструкторы, применять законы де Моргана к логическим выражениям, вводить typedef, удалять определение из класса и многое другое. Значок лампочки на левом поле редактора указывает на доступность одного или нескольких intention actions. Просто нажмите Alt + Enter, чтобы применить одно из действий.

Live Templates и Surround with

Используйте функцию Live Templates (Code | Insert Live Template или Ctrl+J) для генерации готовых конструкций кода. Полный список доступных шаблонов можно посмотреть в настройках (Editor | Live Templates). При необходимости вы можете создавать свои шаблоны или изменять существующие. Чтобы применить готовый шаблон, просто введите его аббревиатуру и нажмите Tab. Для навигации по переменным шаблона используйте клавиши Enter или Tab. Используйте шаблоны Surround with, чтобы быстро обернуть выбранный код в конструкцию языка. Выберите пункт меню Code | Surround With или нажмите Ctrl+Alt+T, чтобы легко сгенерировать if, while, for, #ifdef и другие выражения в зависимости от контекста. Implement, Override и Generate Definitions**.** Почувствуйте всю мощь кодогенерации в CLion, воспользовавшись действиями для создания функций Implement (Ctrl+I), Override (Ctrl+O) и Generate Definitions (Shift+Ctrl+D). Настройки Generate in-place по умолчанию зависят от вашего кода, так как CLion адаптируется к шаблонам, которые вы используете в проекте, и поддерживает наиболее распространенные случаи (например, классы, полностью расположенные в заголовочных файлах или, наоборот, только в .cpp-файлах и т. д.).

3. Форматирование кода

CLion позволяет вам переформатировать ваш код в соответствии с требованиями, указанными в вашей текущей схеме стиля кода. Вы можете переформатировать часть кода, весь файл, группу файлов, каталог и модуль. Вы также можете исключить из переформатирования часть кода или некоторые файлы. Переформатировать фрагмент кода

a) В редакторе выберите фрагмент кода, который вы хотите переформатировать.

   Перед переформатированием вы можете взглянуть на настройки стиля кода, которые применяются к выбранному коду: нажмите Alt+Enter и щелкните изменить настройки стиля кода.

b) В главном меню выберите Код | Переформатируйте код или нажмите Ctrl+Alt+L.

Переформатировать файл

a) Откройте файл в редакторе и нажмите Ctrl+Alt+Shift+L или в окне инструмента «Проект» щелкните файл правой кнопкой мыши и выберите «Переформатировать код».
b) В открывшемся диалоговом окне «Переформатировать файл» при необходимости выберите следующие параметры переформатирования:
   Оптимизировать импорт: выберите этот параметр, если хотите удалить неиспользуемые импорты, добавить отсутствующие или упорядочить операторы импорта.
   Очистка кода: выберите этот параметр, чтобы запустить проверки очистки кода.

Не оставляйте разрывы строк: переформатируйте разрывы строк в соответствии с настройками стиля кода. Этот параметр переопределяет параметр сохранить при переформатировании | Установка разрывов строк.

c) Нажмите «Выполнить».
Если вы хотите увидеть точные изменения, внесенные в ваш код во время переформатирования, используйте функцию «Локальная история».

Переформатировать отступы строк

Отступы строк можно переформатировать на основе заданных параметров.

a) Находясь в редакторе, выделите нужный фрагмент кода и нажмите Ctrl+Alt+I.
b) Если вам нужно настроить параметры отступа, в диалоговом окне «Настройки/Настройки» (Ctrl+Alt+S) выберите «Редактор | Стиль кода.
c) На странице соответствующего языка на вкладке «Вкладки и отступы» укажите соответствующие параметры отступов и нажмите «ОК».

Автоматически переформатировать код при сохранении

Вы можете настроить IDE для автоматического переформатирования кода в измененных файлах при сохранении изменений.

a) Нажмите Ctrl+Alt+S, чтобы открыть настройки IDE, и выберите Инструменты | Действия при сохранении.
b) Включите параметр «Переформатировать код».
c) Кроме того, вы можете настроить способ, которым IDE будет переформатировать ваш код:

Щелкните настроить область, чтобы указать шаблоны имен файлов и каталогов, которые вы хотите исключить из переформатирования.

В списке Все типы файлов выберите типы файлов, в которых вы хотите переформатировать код.

Выберите «Весь файл» или «Измененные строки», если ваш проект находится под контролем версий.

Если выбрать Измененные строки, переформатирование будет применяться только к тем строкам кода, которые были изменены локально, но еще не возвращены в репозиторий.

Исключить файлы из переформатирования

Вы можете исключить группу файлов и каталогов из переформатирования и оптимизации импорта.

a) В диалоговом окне «Настройки/Настройки» (Ctrl+Alt+S) выберите «Редактор | Стиль кода.

b) Перейдите на вкладку Форматирование и в поле не форматировать введите файлы и каталоги, которые вы хотите исключить, используя шаблон глобуса.
c) Вы можете указать несколько шаблонов глобусов, разделенных точкой с запятой; Если нажать, поле расширится, и каждый паттерн будет показан на отдельной строке.
d) Примените изменения и закройте диалог.

Исключить фрагменты кода из переформатирования в редакторе

a) В диалоговом окне «Настройки/Настройки» (Ctrl+Alt+S) выберите «Редактор | Стиль кода.
b) Перейдите на вкладку Форматтер и включите параметр включить/выключить модуль форматирования с маркерами в комментариях к коду.
c) В редакторе в начале области, которую вы хотите исключить, создайте строчный комментарий Ctrl+/ и введите @formatter:off. В конце области создайте еще один комментарий и введите @formatter:on.

Код между маркерами не будет переформатирован.

Сохранить существующее форматирование
Вы можете выбрать правила форматирования, которые будут игнорироваться при переформатировании кода. Например, вы можете настроить IDE так, чтобы простые методы и функции размещались в одной строке, тогда как обычно после переформатирования кода они разбиваются на несколько строк.

a) Перейдите в Настройки/Настройки | Редактор | Code Style, выберите язык программирования и откройте вкладку Wrapping and Braces.
b) В разделе «Сохранить при переформатировании» выберите правила форматирования, которые вы хотите игнорировать, и отмените выбор тех, которые следует применять.
c) Переформатируйте код (Ctrl+Alt+L).

CLion переформатирует ваш код в соответствии с текущими настройками стиля, сохраняя существующее форматирование для выбранных вами правил.

4. Отладка

a) Настройте параметры отладчика.
b) Для проекта CMake выберите нужный профиль CMake.
c) При необходимости создайте или измените существующую конфигурацию запуска/отладки.
d) Разместите точки останова в своем коде.
e) Нажмите кнопку Debug <configuration_name> или используйте другие параметры, чтобы начать сеанс отладки.

Приостановите или возобновите сеанс по мере необходимости.

Во время сеанса отладки вы можете выполнять код пошагово, оценивать выражения, изменять значения на лету, устанавливать часы и точки наблюдения.
После запуска сеанса отладки значок, помечающий окно средства отладки, переключается на, указывая на то, что процесс отладки активен.

5. Запуск

Быстрый способ:

Запуск из редактора

Если вы не собираетесь передавать какие-либо параметры в свою программу, и ваша программа не требует выполнения каких-либо определенных действий перед запуском, вы можете запустить ее прямо из редактора. Щелкните в поле рядом с объявлением класса и выберите «Выполнить». Чтобы запустить скрипт, откройте его в редакторе или выберите в окне инструментов Проект, а затем выберите в контекстном меню пункт запустить <имя файла скрипта>.

Настраиваемый способ:

Если вы собираетесь передавать параметры своей программе или иным образом настраивать ее запуск, используйте конфигурацию запуска/отладки.

a) Создайте конфигурацию запуска/отладки.
b) 2. На главной панели инструментов выберите конфигурацию запуска/отладки, которую вы собираетесь использовать.
c) 3. Щелкните или нажмите Shift+F10.

Когда приложение запускается, вы можете просматривать его выходные данные и взаимодействовать с ним в окне инструмента «Выполнить». Каждая конфигурация запуска/отладки создает отдельную вкладку при ее запуске.

Повторно запускать приложения

На панели инструментов окна инструментов «Выполнить» щелкните или нажмитеShift+F10

Остановить программу

В окне инструмента «Выполнить» щелкните на панели инструментов. Либо нажмите Ctrl+F2и выберите процесс, который нужно остановить.

Приостановить программу

Щелкните правой кнопкой мыши в окне инструмента «Выполнить» и выберите «Приостановить вывод» в контекстном меню. Используйте тот же переключатель, чтобы возобновить программу.

6. Компиляция

Для текущего открытого файла выберите Build | Перекомпилируйте из главного меню (или нажмите Ctrl+Shift+F9):

Для файла в дереве проекта используйте параметр «Перекомпилировать» в контекстном меню (или используйте тот же Ctrl+Shift+F9ярлык):

Для нескольких файлов выберите их в дереве проекта и используйте параметр «Перекомпилировать выбранные файлы» в контекстном меню Ctrl+Shift+F9:

Обратите внимание, что при использовании нескольких файлов перекомпиляция останавливается при первом сбое компиляции.

7. Публикация в репозитории

Включите интеграцию с Git.

a) В главном меню выберите | Включить интеграцию с контролем версий.
b) В открывшемся диалоговом окне выберите Git из списка доступных систем контроля версий и нажмите ОК. Либо нажмите Alt+`и выберите «Создать репозиторий Git» (или нажмите 1). В открывшемся окне Finder укажите корневую папку для локального репозитория Git.

Вы получите уведомление о том, что для вашего проекта создан локальный репозиторий Git. На панели инструментов и в строке состояния появятся элементы управления, связанные с Git:

Теперь доступны специальные окна инструментов для работы с Git: Commit (Ctrl+K или View | Tool Windows | Commit) и Git (Alt+9или View | Tool Windows | Git).

В окне инструмента CommitAlt+0 вы можете просмотреть локальные изменения и зафиксировать их в локальном репозитории Git. Из окна инструмента Git можно работать с журналом Git, управлять запросами на вытягивание из GitHub и тд.

Добавьте файлы в. gitignore

На вкладке Local Changes окна инструмента CommitAlt+0 вы видите список файлов, принадлежащих вашему проекту. Эти файлы еще не добавлены в репозиторий Git — вам нужно выбрать, какими из них вы хотите поделиться, а какие должны игнорироваться системой контроля версий.

a) Сгруппируйте файлы по каталогам: нажмите Ctrl+Alt+P или щелкните на панели инструментов и выберите Каталог.
b) Выберите каталоги, которыми вы не хотите делиться. Например, следующие каталоги можно игнорировать без нарушения целостности проекта:

.idea : настройки вашей локальной установки CLion. Игнорируйте этот каталог, если вы не хотите делиться своими настройками с членами команды.

cmake-build-debug: автоматически созданный каталог для артефактов сборки CMake.

c) Щелкните правой кнопкой мыши выделение и выберите «Добавить в. gitignore | Добавьте в. gitignore
d) Вам будет предложено подтвердить создание нового файла. gitignore в корневом каталоге вашего проекта. Щелкните Создать.

e) В открывшемся диалоговом окне вы можете либо сразу добавить вновь созданный файл в Git, нажав кнопку «Добавить», либо отложить это, нажав кнопку «Отмена».

   Щелкните Добавить. Вы увидите, что файл. gitignore был добавлен в корневой каталог вашего проекта и помещен в область изменений. Каталоги, которые вы выбрали для игнорирования, больше не отображаются
   в списке неверсированных файлов:

Зафиксируйте свой проект в локальном репозитории Git.

Теперь, когда все ненужные директории исключены из списка неверсионированных файлов, нужно просто добавить все файлы в репозиторий и зафиксировать их для сохранения их текущего состояния.

a) В окне инструмента CommitAlt+0 переместите все файлы из списка Unversioned Files в Changes с помощью перетаскивания. Выберите все файлы, установив флажок корневой папки.

b) Введите сообщение для вашего первого коммита:
c) Щелкните Зафиксировать. Соответствующее уведомление появляется после выполнения фиксации:

Поделитесь своим проектом на GitHub

Чтобы сделать ваш проект доступным для других участников, вам необходимо опубликовать его в удаленном репозитории, например, на github.com. CLion обеспечивает интеграцию с GitHub, что позволяет вам управлять проектами, размещенными на GitHub, разветвлять внешние репозитории, управлять запросами на вытягивание и выполнять другие операции GitHub из IDE. Подробнее см. в разделе GitHub.

a) В главном меню выберите VCS | Поделиться проектом на GitHub.
b) В открывшемся диалоговом окне введите логин и пароль GitHub и
   нажмите войти: Если вы не зарегистрированы на GitHub,
   нажмите зарегистрироваться на GitHub, чтобы перейти на github.com и создать там новую учетную запись.
c) Если у вас включена двухфакторная аутент ификация для GitHub, появится следующее диалоговое окно: введите код и нажмите ОК.
d) В открывшемся диалоге вы можете изменить имя репозитория (по умолчанию оно совпадает с именем проекта), имя удаленного (по умолчанию origin), выбрать тип репозитория (публичный или частный) и добавить некоторое описание, если необходимо: Щелкните Поделиться. После успешной публикации проекта на GitHub появится следующее уведомление: Щелкните ссылку в уведомлении, чтобы открыть репозиторий на GitHub.