Часто задаваемый начинающими программистами вопрос – какие материалы читать для повышения своего профессионального уровня? В таком, наиболее общем виде, ответить на данный вопрос невозможно. Это зависит от текущих знаний и умений человека, от его специализации, от задач, которые возникнут перед ним в будущем. Однако я попытаюсь ответить на данный вопрос для того направления программистской деятельности, в котором лучше всего разбираюсь – в создании кроссплатформенных сложных desktop-приложений. Приложений уровня и класса PhotoShop, VisualStudio, MS Word. Таких приложений, в которых с точки зрения программиста возникает больше количество разноплановых задач:

* Отрисовка классического «офисного» GUI типа кнопок, прогресс-баров, виджетов, анимации.
* Отрисовка любой нестандартной графики типа непрерывно движущихся графиков из миллионов точек.
* Работа с базами данных
* Работа с сетью
* Работа с реестром операционной системы
* Работа с принтером
* Быстрое (в том числе параллельное) выполнение сложных вычислительных задач.
* И многое, многое другое.

Наборов платформ и фреймворков, которые позволяют эффективно строить подобные приложения, не так уж много. В свое время, попробовав разные платформы, программисты в моей организации уверенно остановились на следующей связке: язык С++ и платформа Qt. Как показало время, этот выбор не подвел. Вот именно с оглядкой на этот выбор я начну излагать тот набор материалов, которые я рекомендую к изучению начинающими программистами, если они хотят стать настоящими гуру в этом направлении.  
  
Данный набор материалов в свое время изучался автором этого текста, его старшими и младшими коллегами и прошел несколько итераций, чтобы стать оптимальным. Опытным путем были выявлены материалы, изучение которых дает максимальный прирост эффективности работы при минимальном, хотя и не малом времени обучения. Конечно, раз уж мы говорим об оптимальности в сфере обучения, то налет субъективности здесь неизбежен, но я попытался его минимизировать.  
  
Итак, для кого предназначен этот текст. Если Вы – выпускник вуза, который решил связать свою жизнь с программированием, то этот текст – для Вас. Поверьте, Вы сэкономите очень много своего времени, если воспользуетесь моими советами на старте своей карьеры. Если Вы закончили вуз несколько лет назад, работаете программистом, но еще не решаете никакие серьезные задачи – то этот текст для Вас. Вы сможете вырасти до действительно серьезных задач, о которых не стыдно будет рассказать потомкам. Если Вы работаете программистом уже около 5 лет, но чувствуете, что у Вас почему-то карьера «не идет», а у сидящих рядом – «идет», то этот текст для Вас. Если Вы являетесь опытным программистом и начинаете идти по пути руководителя, то просто прочитайте этот текст и распространите его среди подчиненных. А если Вы являетесь гуру, который все испытал и которого сложно чем-то удивить, то можете просто прочитать этот текст, чтобы понастальгировать по пройденному пути, вспоминая ошибки или успехи прошлого.  
  
Итак, в чем же заключается этот, оптимальный с моей точки зрения, путь? Очень просто – в изучении в нужном порядке набора фундаментальных книг. Книг не очень много, но и не мало. Те, кто рассчитывают на халяву, могут этот текст сразу закрывать. Книги надо будет читать, сочетая это с практической деятельностью, то есть – с работой начинающего программиста. Как правило, у программистов даже на работе есть некоторые незаполненные работой промежутки времени, и читать в эти промежутки книги – дело святое. Если этих промежутков нет, то возьмите за правило читать вечером. Если на работе – работа, а иного времени нет, то закрывайте текст и не мучайте себя, но и не плачтесь потом о нерастущей зарплате.  
  
Почему именно книги? Почему упор на книги? Почему не интернет, там же все есть? Верно, интернетом Вам неизбежно придется пользоваться на пути обучения, и довольно часто. И все пробелы восполнять именно интернетом. Но предлагаемые книги — это та база, изучив которую, Вы научитесь ориентироваться в мире программирования. И будете способны далее уже полностью самостоятельно находить ответы на любые вопросы, как в еще не прочитанных книгах, так и в интернете. Набор книг мною подобран именно таким образом, чтобы сформировать эту базу. Уверенную, прочную базу, которую хватит на много лет. Не до конца жизни, но все же.  
  
Сколько времени понадобится, чтобы освоить все эти книги? Хороший вопрос, и самое ключевое слово здесь – освоить. Нужно не просто читать по диагонали, а именно осваивать. То есть применять эти знания на практике, воплощать в коде. Возвращаться к книге, перечитывать с новым понимаем текст. Вот тогда это будет освоение. Вот только тогда это будет накопление той самой базы, которая позволит решать любые задачи. Итак, сколько же времени? Мой ответ – 5 лет. Долго? Да, небыстро. Но опыт показывает, что быстрее нельзя. Хотите за полгода или за год? Нет, не получится, даже если Вы – Эйнштейн. Ни у кого не получалось. В этом особенность программирования в общем и языка C++ в частности. Как говорил мой более опытный коллега, язык С++ — не для криворуких, а создание больших систем – не для слабовольных. Так что если у Вас столько времени или воли нет, для того, чтобы стать опытным, высокооплачиваемым сотрудником, посмотрите куда-нибудь еще. Например, изготовление шаблонных сайтов – куда более простая вещь, чем создание сложных desktop-приложений. Верстальщики и эникейщики – привет!  
  
Итак, приступим к списку книг. Их суммарный объем составляет 5150 страниц. По возможности я буду давать комментарии, почему я рекомендую именно эту книгу, а не иную, или подсказывать альтернативы.

1. Любая книга по языку С++ объемом не менее 600 страниц. Например, «Герберт Шилдт, C++: полное руководство». 800 стр. Или «Бьерн Страуструп: Язык программирования С++». Лично мне больше нравится Шилдт, но это очень субъективно, а в целом обе книги очень достойные. Главное в той книге, что Вы будете читать – объем! Она должна быть большая! Малая по объему книга хорошо подойдет для сдачи студенческого зачета, но не для профессиональной подготовки. Прочитать 2 малые книги вместо одной большой тоже никакого смысла нет – и структурированного осмысления не наберете, и время не сэкономите. Прочитать такую большую книгу будет непросто (по сравнению с художественной литературой), но это будет действительно самая главная книга в Вашей жизни. Спустя некоторое время после прочтения Вы вполне можете забыть некоторые частности, например, как называется функция, которая округляет double в бОльшую сторону. Но Вы точно должны будете помнить, что такая функция есть, и будете способны эту функцию быстро найти и применить! Пока не прочитали эту книгу, к следующим пунктам переходить большого смысла нет. Практика показывает, что период усвоения книги Шилдта объемом 800 страниц занимает от полугода до года. Эту книгу вполне можно читать еще будучи студентом и нигде не работая. А если Вы работаете программистом и не читали подобной книги – то как Вас вообще взяли на эту работу? Если Вы выбрали книгу, начали читать, а потом поняли, что в ней нет ничего про новые стандарты С++ (С++11, С++14, С++17, С++20), то знайте – на этом этапе Вашего развития это совершенно не критично. Это будет потом, в следующих книгах, а на этом этапе материала будет столько, что с непривычки голова будет идти кругом.  
     
   Что Вы будете знать и понимать, когда изучите эту книгу? Вы будете знать по сути только синтаксис языка C++ и его богатые возможности. Вы научитесь писать студенческий код, но не более того. Когда опытные программисты будут просматривать Ваш код в ознакомительных целях (например, на собеседовании), то они однозначно будут относить его к категории «непрофессиональный код» или, если жестче, «говнокод». Но это нормально! На этом этапе Вы получаете только базу, а все профессиональные тонкости – далее.
2. «Скотт Мэйерс. Эффективное использование C++. 55 верных советов улучшить структуру и код ваших программ». 300 стр. У этого же автора есть еще несколько похожих, альтернативных книг: «Эффективное использование C++. 50 рекомендаций по улучшению ваших программ и проектов», «Наиболее эффективное использование C++. 35 новых рекомендаций по улучшению ваших программ и проектов», «Эффективное использование С++. 35 новых способов улучшить стиль программирования». Книги этой серии – примерно про одно и то же, поэтому можно прочитать только «55 верных советов». Эти книги легко читаются, имеют небольшой объем при хорошей плотности материала. Имея за спиной этот материал, Вы сможете наконец начать понимать, что такое плохой код, из какого незнания он берется и сможете писать его значительно меньше. Вот именно на этом этапе Вы сможете по праву ощутить себя программистом. Пусть начинающим и неопытным, но – Программистом.
3. «Qt 5.10. Профессиональное программирование на С++. М. Шлее». 1072 стр. Альтернатива – «Qt 4. Программирование GUI на C++. Саммерфилд Марк, Бланшет Жасмин». Альтернатива, на мой взгляд, сильно хуже: материал выглядит разрозненным и неструктурированным плюс погрешности перевода, а вот Шлее – русскоговорящий, там перевода вообще не требуется (такая вот редкая и приятная для нас ситуация в мире программирования). Изучив книгу Шлее, Вы научитесь самостоятельно создавать небольшие GUI-приложения, объемом 5 000-30 000 строк кода. Но сделать что-то более серьезное на данном этапе не получится — нехватка опыта и знаний накроет снежным комом.  
     
   Пункты 2 и 3 можно при желании поменять местами или вообще читать параллельно, но я бы рекомендовал, если крайней необходимости нет и Вы – не Цезарь, этого не делать.
4. «Стефан Дьюхэрст: Скользкие места С++. Как избежать проблем при проектировании и компиляции ваших программ». 263 стр. Книга классическая для погружения в глубины С++, к сожалению, она несколько устаревает. Воспринимайте её как базу для следующих книг.
5. «Чистая архитектура. Искусство разработки программного обеспечения. Мартин Роберт». 350 стр. Эту книгу очень приятно читать – она написана простым языком, безупречно переведена, она маленькая, но дает много ценного материала. По сути именно в эту книгу автор вложил весь свой опыт, который он излагал в других книгах на протяжении всей своей жизни. Эта книга – просто жемчужина. Освоив эту книгу, Вы должны будете окончательно расстаться с последними остатками говнокода, что до сих пор иногда выходили из-под Вашего пера. Это еще далеко не Мастерство, и даже не Искусство, но просто годный профессиональный уровень.
6. Пришло время детально изучить шаблоны проектирования. 3 прошлые книги их немного касались, но лишь чуть-чуть. К моменту, когда Вы дорастете до этого пункта, в Вашей работе точно должны были встречаться шаблоны. Они там были, даже если Вы этого не заметили. Но на этом этапе надо обязательно почувствовать себя архитектором, научиться видеть шире, подняться с уровня «просто кодера» и никогда туда уже не опускаться.  
     
   Итак, книгу в студию! «Погружение в Паттерны Проектирования, Александр Швец». 406 стр. формата А6. Эта книга написана совсем недавно, даже издать еще не успели (ищите в интернете pdf), написана русскоязычным автором. Обычно рекомендуют не эту книгу, а книгу так называемой «банды четырех»: «Гамма, Хелм, Джонсон: Приемы объектно-ориентированного проектирования. Паттерны проектирования». Я сознательно этого делать не буду. Книга Швеца по объему материала в точности соответствует «банде четырех», но при этом написана намного более простым языком; ее объем в несколько раз меньше; нет устаревших примеров на SmallTalk, как у «банды четырех»; и в целом тот формат pdf, который можно найти в сети, очень удобен для чтения на маленьком экране, что неоценимо в метро (проверено – читается хорошо даже в метро, это редкость для профессиональной литературы!)  
     
   Также стоит сказать, что есть еще хорошая классическая книжка «Фримен, Фримен, Сьерра: Паттерны проектирования». Причина, по которой я ее тоже не могу порекомендовать – она содержит примеры на Java вместо С++.
7. «Интерфейс. Основы проектирования взаимодействия. Купер Алан, Рейманн Роберт М.». 720 стр. Книга здорово учит понимать, что такое – хороший GUI, а что такое – плохой. Альтернатива – книга «Интерфейс: новые направления в проектировании компьютерных систем, Джеф Раскин». Тоже хорошая книга, но она последний раз издавалась в 2005, в то время как Купер обновил свое классическое издание в 2018.
8. «Чакон Скотт, Штрауб Бен. Git для профессионального программиста». 493 стр. К моменту, когда Вы доберетесь до этой книги, работа с системой контроля версий Git, конечно, будет уже привычным рутинным занятием. Но может случиться, что рабочие процессы, основанные на использовании Git, будут в вашей команде выстроены неоптимально. Это может произойти в случае недостаточной компетенции лидера команды, или устаревших представлений, или просто незнания некоторых возможностей Git. Вот именно для целей оптимизации я и могу порекомендовать эту книгу. Даже несмотря на то, что никакие вопросы оптимизации там вообще не обсуждаются. В книге подробно описываются возможности Git, дается базовое представление о возможных подходах к работе. Вот именно эта база и нужна, чтобы разобраться в том, какие рецепты именно для Вашей команды или компании являются оптимальными. Если же у Вас все хорошо, и есть полная уверенность, что все выстроено и так оптимально, то специально читать книгу по Git я бы не порекомендовал, а вместо этого предложил бы ограничиться статьями в интернете.
9. Самое время познать параллельное программирование! Это действительно очень сложная тема. Умение создавать хорошие параллельно работающие программы промышленного уровня – это Мастерство. Большинство книг по этой теме либо довольно сложные, либо имеют очень малый охват материала. А коммерческие задачи обычно намного более сложные, чем те академические и во многом искусственные и узкие примеры, что приводятся в подобных книгах. Поэтому в этой области программирования очень сложно порекомендовать одну книгу, которая принесла бы максимальную пользу. Но я попытаюсь.  
     
   Итак, «Прасид Пай, Питер Абрахам: Реактивное программирование на С++». 323 стр. Непосредственно реактивное программирование обсуждается только во второй части книги, а первая часть – это подводящие главы. Вот в них, на мой взгляд, и есть основная ценность данной книги для новичка. В них довольно просто излагаются проблемы, возникающие при параллельном программировании, идет знакомство с простыми путями решения. Бонусом идут описанные относительно простым языком нововведения C++17, C++14, C++11. В англоязычном варианте довольно много нелепостей, но в русском переводе все исправлено, объяснено и расставлено по полочкам. Редкий случай, когда переводчик разобрался в вопросе лучше, чем автор! В общем, книга получилась довольно простой и малой в изложении.  
     
   Альтернатива – книга «Энтони Уильямс: Параллельное программирование на С++ в действии. Практика разработки многопоточных программ». Эта книга большая и исключительно сложно читается. Некоторые главы (например, про модель памяти и неблокирующие структуры данных) понять с первого прочтения вообще невозможно. Правда, для объективности стоит сказать, что этот же материал про неблокирующие структуры данных и в книге Абрахама понять нельзя из-за сильной урезанности материала.
10. «Хамбл Джез, Фарли Дэвид. Непрерывное развертывание ПО. Автоматизация процессов сборки, тестирования и внедрения новых версий программ». 432 стр. Эта книга покажет, как самому создавать (или руководить созданием) системы непрерывной интеграции, включающие сборку, тестирование, развертывание программных продуктов. Никакой более-менее крупный и долгоживущий программный продукт без этого обойтись не сможет. Альтеранатива – книга «Непрерывная интеграция. Улучшение качества программного обеспечения и снижение риска. Дюваль Поль М., Матиас Стивен». Книга Хамбла, на мой взгляд, немного лучше книги Дюваля, поскольку в последней довольно много «воды» и практически нет конкретики.

Вот, собственно и все! Новички, дерзайте!  
  
Чтобы окончательно расставить точки над i, можно оглянуться назад и спросить – а почему в списке книг нет таких известных авторов, как Макконнелл, Ларман, Буч, Фаулер и другие? Просто потому, что изучение книг этих авторов я не посчитал оптимальным путем развития новичка в мире С++/Qt. И так получилось 10 книг с суммарным объемом 5150 страниц, и добавлять туда не столь удачные книги мне показалось перебором.