Как стать программистом с нуля

Александр Ваньков

Предисловие. О чем эта книга

Как же все-таки им стать? Если говорить в двух словах – то … в двух словах не объяснить. За этим и пришлось написать книгу, требуется чуть больше слов. Итак, давайте по порядку.

Сегодня в книжных магазинах и в интернете Вы встретите огромное количество книг по отдельным языкам

программирования, IT-технологиям, различным методам организации работы, и так далее.

Однако почти нет или очень мало таких книг, которые подскажут в ясном и простом виде – что же именно делать, то есть за что, как говорится, «хвататься» для обозначенной цели. Ведь объем информации по теме, количество видов языков программирования, фреймворков (наборов готовых решений для разработки), просто зашкаливает. Не говоря уж о том, что среди моря этой информации полно откровенно устаревшего материала и обычного мусора. Вы знали, что некоторые языки программирования созданы лишь ради прикола? Сейчас не будем вдаваться в примеры. Давайте лучше перейдем к тому, что Вы сможете узнать из этой книги.

Эта книга не учит отдельному языку программирования или освоению какой-то конкретной IT-специальности. Она показывает путь и конкретные инструменты для его прохождения. А цель этого пути – стать разработчиком, или IT-профессионалом в той области, в которой Вы пожелаете.

Хотите ли Вы создавать мобильные приложения и стать автором какого-нибудь популярного нового сервиса или мессенджера? Или, может быть, Вам интереснее создавать веб-сайты, интернет-порталы, социальные сети? А может, хотите создавать игры или программы для обычных настольных компьютеров? Мы определимся с этим позже, дело сейчас не в этом. А в том, что изучив книгу, Вы поймете что именно нужно делать.

Книга написана *простым и понятным для непрофессионала языком*. Один их минусов попыток узнать методы у Ваших друзей и знакомых программистов, с которым Вы могли столкнуться (но не обязательно) – это то, что они не готовы внятно что-то объяснять. Они оценивают все уже с их точки зрения, мало кто из них переобучался, вероятно они были программистами «изначально», то есть с вуза.

При этом, конечно, программисты хорошо зарабатывают. Сегодня это одна из самых престижных и привилегированных профессий. Опытные программисты в какой-то мере «диктуют» условия работодателю, а не он им. Работодателю приходится делать для них удаленку, комнаты отдыха и фитнеса в офисах, свободный график, и прочие «плюшки».

Все это возможно с полного нуля, даже если у вас нет ни копейки денег, и вы совсем не понимаете в разработке. Было бы желание.

Как же так, спросите вы? Если я не математик, например, а гуманитарий, совсем не понимаю в этих фреймворках и технологиях, как я могу стать программистом?

И здесь расскажу Вам одну занимательную историю, которую когда-то нашел на просторах интернета.

В одном городе жил бомж, который сидел на улице, а мимо него регулярно проходил разработчик. И в один прекрасный день, разработчику пришла мысль: поставить эксперимент. Что если дать бомжу ноутбук, объяснить ему где находятся уроки для изучения программирования, и даже платить ему маленькую сумму денег или давать еды, чтобы этим мотивировать его обучаться. Подумано – сделано. Программист договорился с бомжом, что даст ему ноутбук, а тот будет учиться программировать, за что будет получать некоторое минимальное содержание на жизнь. И учеба пошла! Бомж научился программировать и даже разместил

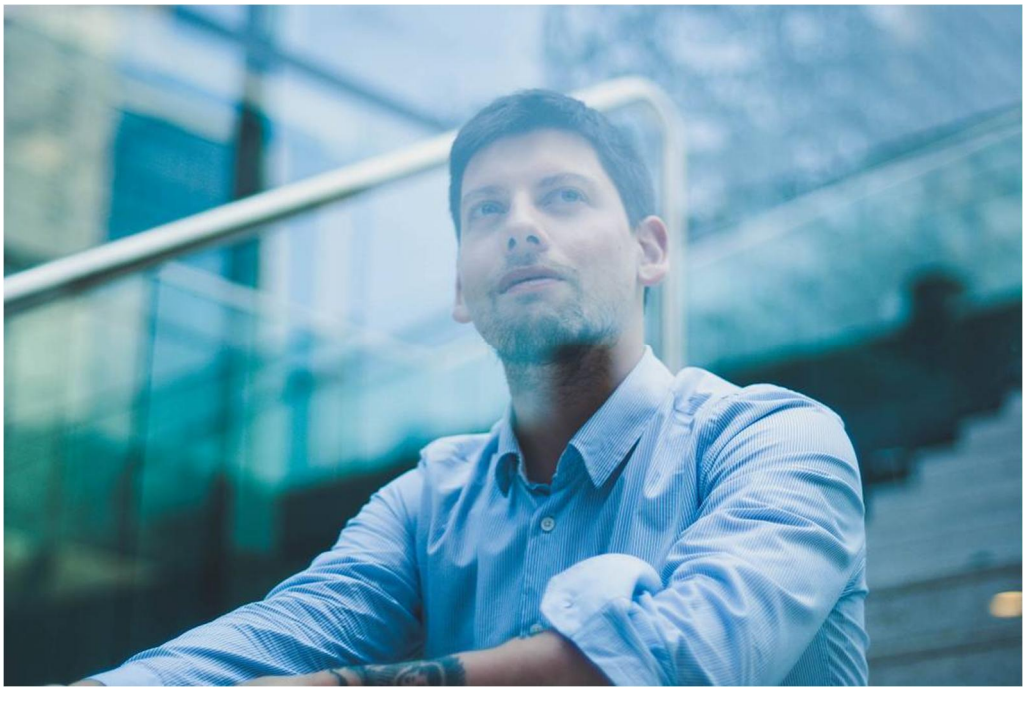
разработанное им приложение, если не ошибаюсь, в Google Play – магазине приложений для Android, и у этого приложения даже были пользователи. То есть, эксперимент был успешным.

Кто-то может сказать, что кем возомнил себя этот разработчик, проводивший эксперимент, как он мог определять судьбу другого человека? Но с этим вряд ли можно согласиться. Он сделал хорошее дело, попытался помочь бомжу.

Надо сказать, что карьера у бомжа не пошла, но не потому что он не захотел, а из-за его ареста за какие-то прошлые преступления. А так, эксперимент показал успех.

То есть даже бомж смог начать программировать. Что уж говорить о том человеке, который читает эту книгу. Вы точно сможете!

Мы еще вернемся к вопросу о возможностях и требованиях в главе 2. А пока давайте вспомним еще одну интересную вещь.



Представьте, что **вы точно знаете что нужно делать** для успешного достижения той или иной цели. При чем вы этого реально хотите и понимаете какой успех достижение этой цели принесет. Будете ли вы это делать? Конечно, что за вопрос.

В фильме «Области тьмы» (*английское название – “Limitless”*) главный герой вдруг, однажды, съев некую таблетку, стал обладать выдающимися сверхспособностями: его мозг работал с невероятной скоростью и он принимал верные решения для достижения успеха. Так и здесь, только конечно, с нашими обычными способностями. У вас есть мозг – и это все что вам для начала надо. А если серьезно, то поговорим об этом в главе 2.

Какова структура данной книги?

Первая глава рассказывает о том, что дает профессия разработчика. Возможно, Вы уже это прекрасно понимаете, тогда имеете полное право переходить к следующей главе. Если представление еще не ясное, то, пожалуйста, прочтите и первую главу.

Вторая глава уже серьезно рассказывает о «системных требованиях» к построению Вашего пути разработчика. Здесь говорится о том, что надо минимально иметь (в материальном, да и в нематериальном плане), чтобы начать обучение, а потом и работу.

Третья глава содержит общий набросок действий и их вариантов, пока без детального их рассмотрения. Здесь же дается обзор востребованных языков программирования.

Главы с четвертой по пятую дают уже детальный обзор отдельных вариантов, со ссылками на конкретные источники знаний и способов начала карьеры. От совсем экстремальных до таких, которые позволят совмещать текущую деятельность с новым для себя направлением – IT (*айти – информационные технологии*).

Остальная часть книги – это как-бы взгляд в будущее. Если Вам уже хочется понять, что же делать после получения нужных навыков, если Вы хотите заглянуть в это будущее и увидеть там себя в роли разработчика, то, пожалуйста.

В какой последовательности читать книгу? Рекомендую прочитать эту книгу *от корки до корки*, но как читать – это Ваш выбор и тут Вы полностью свободны. Это лишь рекомендация.

На всякий случай привожу также **e-mail для связи** со мной (не могу обещать отвечать на каждое письмо, но пишите, если есть важные вопросы): aleksandervankov@gmail.com

Глава 1. Зачем это нужно

Как уже было сказано, возможно, это уже очевидно для Вас. Сегодня везде и всюду говорят о цифровизации, нужности внедрения информационных технологий, и так далее. Но что это (программирование) дает лично Вам? Разобьем этот вопрос на несколько составляющих.

Во-первых, если Вам это интересно, то Вы можете обрести в этом основное занятие в жизни. Если программирование будет Вам нравиться, то Вы без сомнения, добьетесь высокого профессионального уровня, и будете заняты **интересным и увлекательным** делом.

Очень важно, чтобы в жизни было какое-то дело, занятие. Многие пытаются «найти себя», понять что им действительно нужно в жизни. Но данная книга не об этом. Предполагается, что Вам понятно чего Вы хотите – быть разработчиком. Иначе, зачем Вы держите в руках эту книгу (или электронную книгу, или еще какой цифровой девайс, на котором читаете

данную книгу).

Во-вторых, программисты – это, обычно высокоинтеллектуальные люди. Они умеют критически мыслить, их обычно не обмануть, они умеют прекрасно все анализировать. Мышление развивается вместе с освоением навыков разработчика и развитием карьеры. У меня в результате продолжительного общения с разработчиками сложилось четкое представление: это очень продвинутые люди, они могут эффективно управлять своей жизнью. Кому такое не надо? Думаю надо всем. Таким образом, это **высокий интеллект и саморазвитие**.

В-третьих, программисты востребованы на рынке труда и **хорошо зарабатывают**. Высокие зарплаты имеют, конечно, уже опытные разработчики. Начинающие – имеют обычные зарплаты. Однако здесь уже все зависит от Вас. Вам нужно будет постоянно совершенствовать свои навыки, осваивать новые технологии постоянно, новые языки программирования (если необходимо). Можно даже создать и свой бизнес на услугах по разработке приложений или веб-сервисов. Но это тема для отдельной книги.

В-четвертых, из востребованности на рынке труда вытекают и **другие плюсы помимо высокой зарплаты**. Это различные удобства, которые создают IT-компании для разработчиков.

Например, в офисах крупных компаний есть спортзалы, комнаты для отдыха, кухни, иногда небольшие сады. Мне довелось посетить множество офисов подобных компаний, и на момент написания книги моя работа происходит в одном из известных IT-холдингов. Поэтому все это так и есть.

А кто хочет работать дома, или в путешествии, или еще откуда угодно, где есть связь, то и этот вариант возможен. Называется – удаленка (*удаленная работа*). Конечно, сидеть на солнцепеке на пляже с ноутбуком не получится (это фантастика:), потому что не удобно – такое бывает только в картинках из интернета. Но на балконе в номере комфортабельного отеля – вполне. Или в своем загородном

доме. А может быть и прямо на ходу в каком-нибудь аэропорту.

Надо сказать, что не везде доступна такая удаленка. Некоторые работодатели в целях безопасности (чтобы не было утечек исходного кода) позволяют работать лишь в офисах корпораций. Но выбор есть всегда – здесь уже Ваш выбор работодателя, который для Вас более комфортен по условиям или соотношению зарплаты и этих условий.

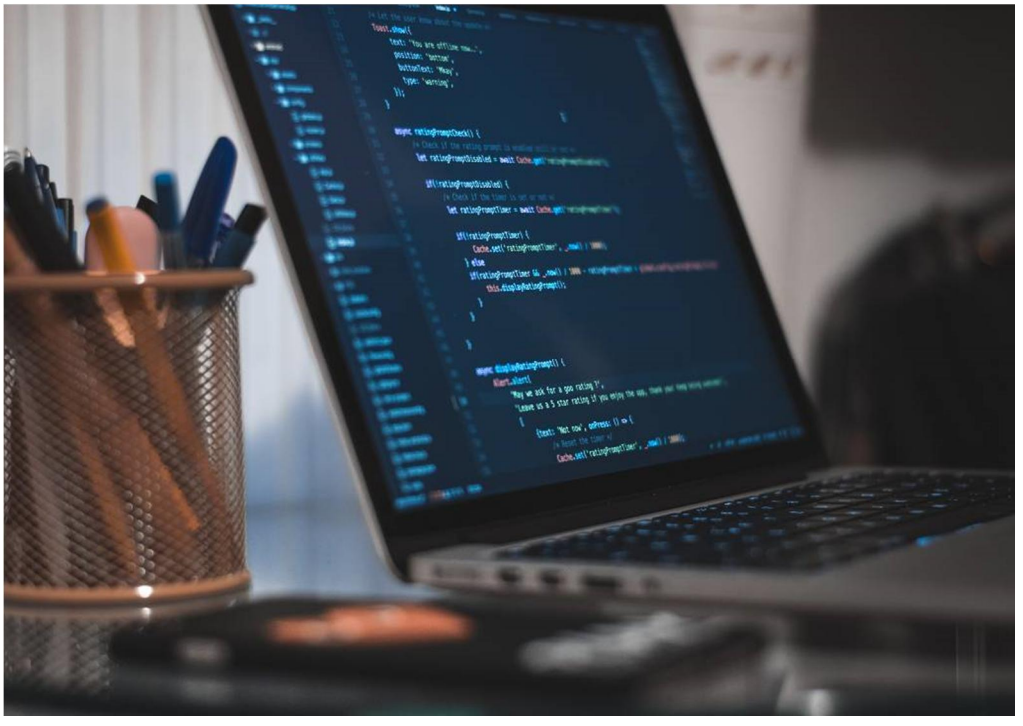
В-пятых, наконец, только **начав программировать**, Вы поймете «Ваше» это или «не Ваше». Важно попробовать, испытать подходит ли это Вам. И если подходит, если захватывает, то все предыдущие четыре пункта реализуются сами собой.

Глава 2. Реально ли мне стать разработчиком

Надеюсь, Вы прочитали историю о бомже в предисловии. По моим данным история имела место в реальной жизни, и у меня нет оснований не верить источнику. Понятно, что нужна мотивация. Однако прежде рассмотрим технический инструментарий.

Что же надо иметь **в материальном плане** для обучения программированию и начала работы.

Самая нужная и, по сути, единственная вещь, которая требуется – это машина :). Имеется в виду электронно-вычислительная машина, которой является персональный компьютер или ноутбук. Планшет или смартфон для этой цели не подойдут. Хотя, какую-то элементарную учебу можно начинать на планшете, полноценно работать на нем не удобно. Обязательно нужна физическая клавиатура, полноценная операционная система, такая как Windows, macOS, Linux. Для разработки не требуется самый современный ноутбук или компьютер. Однако он не должен быть еле живым десятилетней давности. Возьмите для себя нормальный рабочий инструмент – Вам должно быть комфортно работать за Вашим цифровым устройством (ноутбуком или компьютером). Если у Вас его нет, то придется купить. Такой средней мощности ноутбук или компьютер достаточен для начала. Важно чтобы у него был нормальный центральный процессор, достаточный объем памяти для повседневной работы.



Однако если Вы собрались заниматься разработкой игр или приложений с 3D-графикой, виртуальной или дополненной реальности, то без мощной видеокарты не обойтись. А это уже довольно дорогое устройство (хотя смотря для кого, и с чем сравнивать…).

Наконец, в отношении компьютерного оборудования сделаем еще одну оговорку. Возвращаясь в прошлое, да и к опыту других людей, мной было замечено, что лучшие свои IT-проекты удавалось и удается делать с какими-то малопригодными средствами: далеко не новая компьютерная техника, в гаражах, и тому подобное. Как начинались компании-гиганты современной IT-индустрии, Apple, Amazon? Именно так. Потому что «железо», то есть техника, это не самое важное. Что же надо еще? Читаем далее.

Что касается программного обеспечения для разработки приложений или веб-разработки, то практически все оно на сегодняшний день бесплатно. Вы можете начать что-то разрабатывать пользуясь бесплатными продуктами, при этом

они доступны в таком виде совершенно официально, не нужно искать какие-то «паленые» версии и тому подобное.

Лишь при переходе в какие-то сложные проекты, где требуются весьма специфические продвинутые программные инструменты и возможности, Вам понадобятся профессиональные версии продуктов для разработки или отдельные платные фреймворки. Но, скорее всего, их будет оплачивать работодатель. Либо Ваша зарплата будет такой, что их стоимость не будет для Вас иметь существенного значения. Да и к тому времени Вы уже купите себе самый «наворочанный» MacBook Pro.

Далее, нужен доступ в интернет. Желательно нормальный скоростной канал доступа, то есть проводной интернет, безлимитный. 50 мегабит в секунду будет вполне достаточно, или можете выбрать более скоростной тариф. Можно работать и в сети 4G, но тогда пакет трафика должен быть реально большим или безлимитным, а скорость доступа в месте Вашего нахождения комфортной для скачивания сред разработки, обучающих материалов, видеоконтента и прочего. Такие пакеты стоят значительно дороже проводного интернета.

В остальном Вы оборудуете свое рабочее место как Вам удобнее. Мы вернемся к материальным инструментам и к рабочему месту в главе 7, а пока уже хватит заниматься обсуждением этих материальных вещей.

Теперь коротко обсудим гораздо более важную составляющую вашего становления как разработчика – **сознательную, или психологическую**. Ноутбук может у Вас быть, и все остальное в техническом плане. А вот станете ли Вы программистом – зависит в основном от Ваших действий, которые, в свою очередь, контролируются сознанием. Бывает, сложно начать такую длительную учебу,

кто-то может бесконечно прокрастинировать, откладывать занятия или «забивать» на них временами или надолго.

Эта книга не претендует на то, чтобы научить Вас организовывать себя, свою эффективность и развивать энергичность. По этой теме существует невероятное множество книг. Скажем лишь в двух словах: надо заниматься физкультурой, правильно питаться, нужен здоровый сон, крепкие социальные связи и другие подобные факторы. Если они еще отсутствуют в Вашей жизни, самое время их внедрять. Точнее внедрять их – всегда самое время, если их нет. Так что, начинаем все менять и перестраивать.

Что же об образовании? Нужно знать математику в совершенстве? Здесь скажем, что вначале достаточно знать простые арифметические действия – это знают все. А остальная математика подтянется вместе с началом изучения, по ходу изучения, с началом работы и по ходу работы. Более того, все что преподают в обычных учебных заведениях (прежде всего – школах), это настолько неудобно и непонятно, что усвоить это реально сложно. На самом деле все становится понятно лишь когда понимаешь цель: для чего нужны те или иные вычисления и уравнения, формулы и прочее. В программировании как раз есть понимание этих целей, а значит, сознание не ощущает какой-то бессмысленности и оторванности, и даже не воспринимает расчеты как нечто абстрактное. Все становится логичным и понятным.

Таким образом, какое-то особое образование не нужно. Опять здесь напомню историю с бомжом из введения. Если Вы читаете эту книгу, значит у Вас есть все что нужно.

В чем же тогда проблема, почему все не идут учиться и не становятся программистами если все так просто, такая высокая зарплата и другие плюсы?

Здесь есть несколько нюансов.

Несмотря на всю доступность и ясность пути, доступность ресурсов для освоения специальности, требуется немало времени и сил, усидчивости и дисциплины.

Кроме того, многие просто не знают о том, что сегодня куда перспективнее быть программистом, чем каким-нибудь юристом или экономистом. Несмотря на то, что государство делает для развития IT немало. Возможно от отсутствия опыта, возможно от непонимания профессии, люди массово не становятся программистами.

Но хотелось бы. На сегодня это одно из самых перспективных направлений профессионального развития и карьеры.

Глава 3. Что делать: варианты и первые шаги

Теперь начнем разбирать самый важный вопрос: что делать для становления разработчиком?

Ответ зависит от Вашего возраста, образования, места нахождения, готовности или неготовности перемещаться в другие места, объема свободного времени и денег, знания английского языка и так далее. Много есть обстоятельств.

Стоп, не ужели надо обязательно куда-то перемещаться или платить за обучение? Вовсе не обязательно! Перечисленные условия не создают непреодолимых препятствий для изучения программирования. Есть бесплатные способы, есть платные. В зависимости от исходных условий всего лишь будет обозначен подходящий Вам путь. Но прежде следует сделать две оговорки.

Во-первых, все-таки надо **изучить английский язык** на уровне, достаточном для чтения и понимания технической литературы. Это не так уж сложно, как кажется. Опять же, в данной книге затруднительно описать методы обучения иностранному языку, все же у нее другая цель. Существует

множество как платных, так и бесплатных методов освоения английского языка. Это и местные клубы или курсы, которые могут быть бесплатными либо платными, это и онлайн-сервисы для обучения, и занятия с репетиторами, и просто общение на разных онлайн-площадках с носителями языка. Конечно же, сюда входит чтение книг на английском языке, как художественных, так и нехудожественных, просмотр фильмов, обучающих видео, прослушивание обучающих аудио наподобие «английский 100% за рулем» либо “Learn Real English” и так далее. Работа эта длительная и систематичная. Но уже вскоре после серьезного изучения языка в течение нескольких месяцев, чтение «программистской» литературы, справочных руководств и англоязычных форумов станет значительно легче.

Английский язык – это основной язык документации, общения разработчиков, доступных и бесплатных учебников и курсов по программированию. И если говорить о профессиональной карьере, то там он просто необходим для нормального развития.

Во-вторых, **скорость обучения** тоже будет определяться тем, как Вы управляете обстоятельствами, изложенными в начале главы. Если Вы можете их менять, это одна история, а если нет – то другая.

При этом обстоятельств в нашей жизни бесчисленное множество, жизнь сложнее каких-то строгих разновидностей этих обстоятельств. Что же делать, как вообще определить путь?

Просто – мы все упростим, мы сгруппируем (а может, разделим) эти виды ситуаций на несколько типичных, создав своего рода модель тех или иных стартовых позиций. В следующих параграфах этой главы будут перечисляться эти модели и рекомендуемые варианты Ваших действий. Вы можете выбирать готовую модель или комбинировать их как считаете нужным.

Прежде, чем переходить к вариантам учебы, посмотрим какие **языки программирования** сегодня наиболее востребованы.

В интернете существует специальный, так называемый, рейтинг или индекс популярности языков программирования. Ведет его компания TIOBE Software BV. Приведем «топ 20» этого индекса по состоянию на середину 2020 года:

1-е место: **язык C**; рейтинг 17.19%;

2-е место: **Java**; рейтинг 16.10%;

3-е место: **Python**; рейтинг 8.36%;

4-е место: **C++**; рейтинг 5.95%

5-е место: **C#**; рейтинг 4.73%;

6-е место: **Visual Basic**; рейтинг 4.69%;

7-е место: **JavaScript**; рейтинг 2.27%;

8-е место: **PHP**; рейтинг 2.26%;

9-е место: **R**; рейтинг 2.19%;

10-е место: **SQL**; рейтинг 1.73%;

11-е место: **Swift**; рейтинг 1.46%;

12-е место: **Go**; рейтинг 1.02%;

13-е место: **Ruby**; рейтинг 0.98%;

14-е место: **Ассемблер**; рейтинг 0.97%;

15-е место: **MATLAB**; рейтинг 0.90%;

16-е место: **Perl**; рейтинг 0.82%;

17-е место: **PL/SQL**; рейтинг 0.74%;

18-е место: **Scratch**; рейтинг 0.73%;

19-е место: **Classic Visual Basic**; рейтинг 0.65%;

20-е место: **Rust**; рейтинг 0.64%.

Сразу скажем, что не надо ориентироваться в выборе того, какой язык программирования изучать, исключительно на этот рейтинг, дело в том, что рейтинг подвергается критике, не все считают его объективно отражающим реальность.

Например, много хороших языков программирования идет уже не в «топе 20», но это не значит, что не следует их изучать.

Например, в этом же индексе есть такие языки, как:

21-е место: **Objective-C**; 0.61%;

22-е место: **Delphi/Object Pascal**; 0.59%;

30-е место: **Kotlin**; 0.45%;

50-е место: **Bash**; 0.17%.

Вы можете ознакомиться с полным индексом здесь: https://www.tiobe.com/tiobe-index/

Нужно ориентироваться *НЕ на популярность языка*, а **на то, что ближе Вам, и какие задачи Вы собрались решать**. Пожалуй, – это самое важное в выборе того, чему учиться.

Так как же выбрать языки программирования для изучения:

Совет 1: начните с простого, например, с изучения Python.

Совет 2: вам все равно придется изучать несколько языков программирования, так что берите тот, который Вам кажется проще или подходит под Вашу идею (если у Вас есть идея какого-то приложения или сервиса).

Совет 3: если Ваша цель исключительно высокая заработная плата, то выбирайте языки для разработки мобильных приложений.

Не случайно я говорю об изучении языков программирования, а не одного языка.

Дело в том, что Вам лучше определиться *НЕ с конкретным языком*, а с **областью деятельности или профессией**. Что это значит?

Приведем примеры:

Веб-разработчик фулл-стек (full-stack) – это такой разработчик, который умеет создавать полностью веб-порталы, интернет-сервисы. Применимые языки, технологии: PHP, Python, JavaScript, HTML/CSS, SQL и другие базы данных, Linux.

Бэк-енд (back-end) разработчик – создает невидимую пользователю, внутреннюю программную часть интернет-сервисов, скрипты, работа с базами данных. Применимые языки, технологии: PHP, Python, Haskell, Perl, Ruby, SQL, NoSQL, Linux и другие.

Фронт-енд (front-end) разработчик – создает веб-приложения, внешнюю их часть, программирует пользовательский интерфейс. Применимые языки, технологии: HTML/CSS, JavaScript и его многочисленные фреймворки, SQL и другое по необходимости.

Разработчик мобильных приложений (для iOS, **Android)** – использует языки: Objective-C, Swift (для iOS); Java, Kotlin (для Android).

Разработчик десктоп приложений (desktop – рабочий стол) – использует C++, C#, VisualBasic, Delphi и другие языки, создает обычные программы для Windows или иных операционных систем, которые работают на персональных компьютерах и ноутбуках.

Разработчик игр (game developer) – применяет C++, C#, другие языки программирования, а также различные игровые движки.

DevOps инженер – знает работу операционных систем, Unix/Linux/BSD/Windows, администрирование серверов, облачные технологии, базы данных, скриптовые языки программирования (Python, Perl, Bash) и многое другое. Занимается тоже очень многими вещами.

Также, как правило, все разработчики знакомы с Git, алгоритмами и структурами данных, всевозможными IDE (integrated desktop environment – это среды разработки) и прочими различными технологиями.

Это не полный перечень. Возможно, Вам хочется изучать **Big Data**, **AR/VR** (дополненная реальность или виртуальная реальность), **AI** (искусственный интеллект), или что-нибудь еще. Везде будет свой набор знаний.

Тем не менее, вот из этого (что перечислено выше или что Вы подберете для себя исходя из подходящих критериев) и надо выбирать. Так что вначале определитесь именно с этой областью, а не с конкретным языком.

Существует также различная аналитика по **зарплате** в общем и в зависимости от языка программирования. Возьмем для примера данные «Хабр Карьеры» за второе полугодие

2019
года: https://habr.com/ru/company/habr_career/blog/500786/

В среднем (для любых языков) зарплаты выглядят так:

Стажер (**Intern**): 31 000 рублей.

Джуниор (**Junior** – начальный уровень): 50 000 рублей.

Миддл (**Middle** – средний уровень): 100 000 рублей.

Сеньор (**Senior** – опытный разработчик): 160 000 рублей.

Лид (**Lead** – высококвалифицированный руководитель команды разработчиков): 180 000 рублей.

Однако это средние значения. Мне известны разработчики с зарплатами и более 200 000 рублей, и более 400 000 рублей.

Это мы всего лишь говорим о России. За рубежом зарплата измеряется десятками и сотнями тысяч долларов в год для опытных разработчиков. Однако для того, чтобы добиться такого уровня, Вы должны понимать какой огромный и кропотливый путь придется пройти. Возможно, Вы остановитесь на каком-то из средних уровней. Не призываю Вас к этому, а наоборот предлагаю никогда не останавливаться. Но не все имеют такую дисциплину и целеустремленность. Кто знает, может Вы, все-таки, окажетесь в числе этих выдающихся творцов цифровых технологий.

Не будем сейчас вдаваться в цифры по конкретным языкам программирования, потому что они меняются, да и не следует привязываться к этому. Почти в любой области программирования можно добиться высокого уровня дохода, если выйти на уровень сеньора или тимлида, а также если это действительно «ваше дело».

Далее перечислим модели возможного освоения профессии в общем виде.

§ 3.1. Если у Вас гуманитарное или иное подобное образование, и Вы уже давно работаете в этой области и не можете все бросить

При наличии гуманитарного образования и работы по гуманитарной специальности, возможно, учебу придется начинать не только с нуля, но и с готовностью в будущем, когда освоитесь, оставить то, чем Вы сейчас занимаетесь. Это – если Вы хотите постепенно прекратить Вашу текущую деятельность, работу, уйти когда-либо из профессии.

Но Вашей целью может быть и изучение программирования для того, чтобы использовать какие-то инструменты в своей профессиональной деятельности, ускорить, автоматизировать свою работу, стать эффективнее на своем месте. Для этого у Вас должны быть хотя бы примерные идеи как это сделать.

Также виды решений здесь зависят от свободного времени, объема обязательств перед кем-либо (семья, кредиты и тому подобное).

Давайте перейдем непосредственно к решениям. Начнем с самых простых, а далее перейдем к более сложным.

Учеба на бесплатных или платных онлайн-курсах – первое, что Вы можете попробовать. Рекомендую начинать с этого. Почему? Такая учеба даст Вам быстрое понимание что представляет из себя программирование, Вы также сразу включитесь в среду других учеников, разработчиков, наставников, преподавателей. Но эффективность этих курсов весьма различна, а часто может быть и низкой. И просто прослушать лекции не достаточно! Надо читать дополнительную информацию, книги, руководства, в том числе, о которых будет сообщаться на курсах. То есть прохождение этих курсов – это «верхняя часть айсберга» всего образовательного процесса.

О конкретных курсах со ссылками на веб-сайты рассказывается в главе 4.

Такие курсы подходят Вам если у Вас есть занятость на основной работе, есть домашние или семейные дела и обязанности.

По ходу изучения этих курсов все равно нужно что-то начинать изучать самостоятельно, желательно сделать какой-то свой проект (например, разработать приложение или

веб-сайт).

Самостоятельное изучение основ программирования, отдельных языков программирования и вообще computing и IT – второй, более сложный вариант.

Есть бесплатные видеокурсы, есть книги, есть онлайн-учебники и тренажеры. Материалов в сети так много, что никакой разумного объема книги не хватит, чтобы все их охватить. В главе 6 рассматриваются конкретные известные примеры интернет-платформ, книг, и прочего, предназначенного для такой учебы.

Здесь очень важна дисциплина. Как правило, не будет дедлайнов для выполнения заданий, никто не будет проверять и контролировать процесс. Такая учеба требует явного энтузиазма.

Здесь Вам также требуется высокий уровень английского языка, ну или хотя бы достаточный для понимания документации, а желательно и видеоуроков. Бесплатные онлайн-курсы на русском языке, как правило, очень ограниченны.

Если Вас интересует дальнейшее **трудоустройство**, то следует начать формировать свое портфолио – то есть наглядную демонстрацию Ваших разработок. Далее все зависит от Вашей цели – если это трудоустройство, то надо изучать смежные языки программирования, пробовать устроиться куда-либо с самой минимальной заработной платой. Фриланс, работа с заказчиками без работодателя – это тоже вариант, но рекомендовал бы его уже после того, как попробуете себя в обычной работе по найму. Однако тема поиска работы – это отдельный большой вопрос, который мы разберем в главе 7.

Если Вам нужно освоить программирование «для себя», для **запуска своего проекта**, для создания бизнеса или управления уже существующим, для совершенствования своей профессиональной деятельности, то следует начинать развивать свой проект (приложение, веб-сервис, какие-то

решения для автоматизации работы).

Возможно, у Вас возникнет множество целей: и попробовать работу, и фриланс, и создать свой проект. Все можно попробовать, и выбрать то, что Вам интереснее всего.

§ 3.2. Если Вы уже не молоды, и Вам нечего терять: нет работы и перспектив ее найти, обязательств

Если Ваша текущая профессия (специальность) не востребована на рынке труда, и Вы готовы с нуля освоить разработку программного обеспечения, то это один из лучших вариантов! Вы можете целиком погрузиться в изучение новых навыков.

При таком варианте Вы можете уже пойти и на **очную учебу**. Какую учебу выбрать зависит от наличия денег, места нахождения, готовности его менять.

Если с деньгами плохо, то есть бесплатная учеба.

Другой вариант – получение **второго высшего** образования. Вариант требует финансовых затрат и подходит только если Вы можете себе это позволить. Если Вы не в Москве, то лучше переехать туда.

В главе 5 разбираются варианты как платного, так и бесплатного очного обучения.

Не обойтись в любом случае без **самостоятельной учебы**. Она требует хороших знаний английского языка, а в большом количестве онлайн-материалов поможет разобраться глава 6.

О трудоустройстве или работе «на себя» смотрите два последних абзаца предыдущего параграфа. Они применимы и здесь.

§ 3.3. Если Вы еще только заканчиваете школу и выбираете вуз, профессию

Это самый лучший вариант, чтобы начать успешную карьеру программиста. Если Вы читаете эту книгу и думаете пойти ли учиться на какую-то гуманитарную специальность или все-таки на связанную с IT или другими высокими технологиями, сегодня однозначно надо выбирать – IT или высокие технологии.

У нас огромное количество никому не нужных бухгалтеров, юристов, экономистов. Высшие учебные заведения продолжают принимать на эти специальности. Не понятно только зачем и кто туда поступает. Ладно, скажем, МГУ, МГИМО, несколько иных подобных университетов, выпускают хороших специалистов в области этих наук (специальностей). Зачем нужны различные академии и факультеты по гуманитарным специальностям в технических вузах и огромном количестве непонятных региональных негосударственных вузов. Они выпускают армию людей, образование которых бесполезно. Но есть спрос – значит, имеется и предложение.

Давайте будем менять этот спрос. Поступить на техническую специальность, на мой взгляд, проще, там больше бюджетных мест. А перспективы карьеры выше. Об этом очевидно сказано в главе 1.

Даже если Вам захочется управленческой работы в будущем, Вы без труда освоите навыки коммуникаций с людьми, так называемые “soft skills” – то есть навыки, не связанные с основной профессией, направленные на коммуникацию, умение выстраивать отношения с людьми. А вот освоить высшую математику, технические науки, “hard skills” будет потом намного сложнее. Ведь не зря же пришлось писать эту книгу, не зря такой спрос на учебу на программиста – но всем хочется, чтобы это было попроще и побыстрее.

Отсюда следует вывод: для Вас оптимальным вариантом является поступление в **университет** на специальность,

связанную с информатикой, разработкой программного обеспечения, или электроникой, или другую подобную.

Не следует думать, что там научат всему и не придется учиться самостоятельно. Скорее там предоставят определенную базу, позволяющую сформировать соответствующий склад ума.

Самостоятельное изучение разработки программ должно в этом случае все равно идти параллельно учебе в вузе. Как организовать самостоятельную учебу рассказывает глава 6.

Вам также требуется высокий или как минимум достаточный уровень английского языка. Возможно в вузе и будут его преподавать, но что-то подсказывает, что не всегда то преподавание может быть качественным. Английский придется изучать и самостоятельно. Это не сложно.

Конечно, можно учиться и без университета, или в бесплатной школе разработчиков. Но особого смысла в таком возрасте я в этом не вижу.

В завершение хотелось бы сказать, что если Вы читаете эту книгу еще только решая какую специальность, университет выбрать, то это прекрасно, думаю, что из предисловия и главы 1 логичным становится выбор в пользу IT.

Если же Вы думаете о том, как развивать Ваших детей, какую учебу рекомендовать им, то это тоже хороший шанс дать им востребованное образование. Главное, конечно, чтобы они сами хотели осваивать эту специальность. Но кто не хочет жить в тех условиях, в которых живут высокооплачиваемые разработчики? Думаю, таких мало.

§ 3.4. Если Вы уже разбираетесь в информационных (компьютерных) технологиях, но не программист

На первое место в рекомендациях в этом случае поставлю **самостоятельную учебу**.

В интернете полно материалов, связанных с программированием и кодингом: книги, бесплатные видеоуроки, техническая документация, учебники, тренажеры. В главе 6 рассматриваются конкретные примеры.

При самостоятельной учебе очень важна дисциплина. Никто не будет проверять и контролировать процесс. Такая учеба требует от Вас явного желания развиваться в этом направлении.

Вам также требуется высокий или как минимум достаточный уровень английского языка. Английский придется изучать самостоятельно или на курсах английского. Это не сложно.

Конечно, если есть желание изучать **онлайн курсы**, или поступить в **учебное заведение** по соответствующей специальности или направлению подготовки, то это вполне возможно, но надо ли Вам это? Решать, конечно, Вам. Эти курсы упростят Вам задачу, но рекомендация – оставить это как не главный метод учебы. Подробнее об онлайн-курсах или очной учебе можете почитать в предыдущих параграфах этой главы, а также в главах 4 и 5 книги.

Глава 4. Онлайн учеба на курсах разработчиков

В этой главе мы разберем некоторые, самые, на мой взгляд, популярные онлайн-курсы, как русскоязычные, так и англоязычные. Большее внимание уделим русскоязычным курсам – все-таки проще начать учебу с них, особенно если у Вас гуманитарное образование и не самый высокий уровень знаний английского языка.

Сразу надо сказать, что многие из русскоязычных курсов оцениваются профессиональными разработчиками достаточно низко. Тем не менее, если очень постараться, можно использовать их как стартовую площадку для начала изучения информационных технологий, а в отдельных случаях – для начала карьеры. Но все по порядку: итак давайте посмотрим, что у нас здесь имеется.

§ 4.1. GeekBrains

Веб-сайт курсов: https://geekbrains.ru

Это курсы от Mail.ru – самые раскрученные в рунете на момент написания данной книги.

На курсах представлен широкий спектр не только программистских специальностей, но и других IT-специальностей и отдельных курсов.

Сайт данной платформы располагает мощнейшим маркетингом. Чтобы понять какие курсы Вам подходят, у них организованы бесплатные вебинары, мастер-классы, профориентация, дни открытых дверей в офисе Mail.ru с собранием множества народа. Все это впечатляет.

Некоторые коротенькие курсы дают даже бесплатно при прохождении каких-либо опросов, или в порядке тестирования, а также при проведении иных акций. Часто дают скидки на учебу.

Сами курсы на данном портале можно разделить на две большие группы:

GeekUniversity – это первая группа. Это достаточно объемные курсы, называемые факультетами. Указано, что они дают обучение до уровня Middle с гарантированным трудоустройством. Опять же напомню, что без самостоятельной дополнительной работы над изучением соответствующего предмета и создания своего портфолио, успех маловероятен.

Среди факультетов представлены такие, как факультеты: искусственного интеллекта, разработки на языках Python, Go, Java, разработки для iOS и Android, веб-разработки, DevOps, Big Data, Data Engineering, системной и бизнес-аналитики, разработки игр, информационной безопасности, различных видов дизайна (графика, веб, продуктовый, интерьеры, игры, UX/UI), продакт- и проджект-менеджмента, интернет-маркетинга, SMM, и даже тестирования ПО, управления персоналом.

GeekBrains «просто» – это вторая группа. Представляет собой обучение с нуля до уровня Junior, трудоустройство не гарантируется. Эти курсы покороче, подешевле. Они подходят лишь для того, чтобы ознакомиться с конкретными областями знаний, а не получить в готовом виде комплексную учебу по новой специальности.

Перечислять все виды этих курсов здесь не будем. Они сейчас уже не особо популярны, портал в основном продвигает сейчас свои факультеты.

Есть еще GeekSchool для детей. Выбор там не большой, и уделять внимание этому блоку курсов здесь не будем.

Кроме собственно курсов, GeekBrains предоставляет платформу для построения карьеры (размещение резюме) или набора в свой проект выпускников курсов (размещение вакансий).

Итак, если хочется попробовать, то регистрируемся на сайте: https://geekbrains.ru

§ 4.2. Яндекс.Практикум

Веб-сайт курсов: https://praktikum.yandex.ru

Данные курсы, на мой взгляд, являются более качественными, но выбор вариантов курсов здесь значительно меньше. На сайте курсов указывается, что Яндекс.Практикум помогает людям расти – на работе и в жизни, освоить новую специальность и получать удовольствие от того, чем занимаетесь.

Курсы имеют бесплатную вводную часть, позволяющую попробовать: подойдет ли Вам тот или иной курс.

По результатам курсов обещается помощь в трудоустройстве (но не 100% его гарантия): помощь в составлении резюме, откликах на вакансии и тренировка в прохождении собеседований.

Каких-либо особых требований к поступающим не предъявляется, достаточно иметь среднее образование. Хотя некоторые темы, особенно на курсах по работе с

данными, связаны с математикой и могут быть в этой части сложны для гуманитариев. Видимо, все сделано так, что любой человек сможет в этом разобраться при желании.

Перечень курсов на момент написания данной книги следующий:

Python-разработчик – 9 месяцев обучения по 10 часов в неделю, по результатам предполагается создание портфолио из проектов: социальная сеть, бот, онлайн-турнир по шашкам или игре в го. Курс ориентирован на веб-разработку, а точнее бэкенд, работу с базами данных.

Веб-разработчик – 10 месяцев обучения по 10 часов в неделю, в основном ориентирован на фронтенд. На курсе изучаются верстка веб-сайтов, JavaScript, фреймворк React, Node.js, серверный JavaScript.

Инженер по тестированию – курс длится всего 4 месяца по 15 часов в неделю. Профессия тестировщика предполагает создание сценариев тестирования, прогнозирование сбоев и поиск ошибок в программном обеспечении.

Флоу – курс английского языка, длится полгода. Есть два уровня: Elementary и Intermediate.

Аналитик данных и **Специалист по Data Science** – это еще два курса по работе с данными. Первый длится 6 месяцев, второй – 8 месяцев. Оба по 10 часов в неделю. Курс по Data Science ориентирован на работу с большими данными, при этом оба этих курса похожи по набору изучаемых инструментов.

У Яндекса еще есть Академия, это нечто иное, но попасть туда реально сложно, учеба не с нуля. Принимают по конкурсу. Наиболее перспективная учеба очная, есть также онлайн-курсы. Если найдете там что-то для себя, а тем более пройдете по конкурсу туда на учебу, это будет очень хорошо.

§ 4.3. Skillbox

Веб-сайт курсов: https://skillbox.ru

Позиционирует себя как онлайн-университет. Чем-то напоминает GeekBrains по наполнению курсами. Платформа не такая раскрученная, как курсы от Mail.ru. Тем не менее, выбрать из чего есть.

Курсы здесь группируются (или классифицируются) по двум крупным блокам:

Профессии с трудоустройством – это программы достаточно длительные для освоения (от одного года). Здесь обещают освоение специальности с нуля, формирование портфолио по ходу учебы и гарантированное трудоустройство. Вот перечень таких профессий на момент написания книги: веб-разработчик (фулл-стек, фронт, бэк), геймдизайнер, разработчики игр на Unity, на Unreal Engine 4, Data Scientist, профессии для различных языков программирования: Java, Python, C#, C++, 1C, PHP, Go, тестировщик, разработчик под iOS, Android, специалист по кибербезопасности, разработчик VR&AR, DevOps-инженер, TeamLead, IT-рекрутёр, дизайнеры различных направлений, включая 2D и 3D, менеджеры проектов или продуктов, маркетологи всевозможных направлений и так далее.

Курсы «просто» – это более короткие курсы, продолжительностью от нескольких недель, но не более года. Включают курсы для освоения отдельных языков программирования, фреймворков или технологий.

Платформа содержит в себе также онлайн-вебинары для ознакомления, которые проводятся бесплатно. Также имеется центр карьеры, где помогут с составлением резюме, собеседованиями, предоставят карьерные консультации.

§ 4.4. Учебный Центр «Специалист» при МГТУ им. Н.Э.Баумана

Веб-сайт курсов: https://www.specialist.ru

Значительное количество курсов по различным направлениям и тематикам. Многие из них не относятся непосредственно к IT.

Группы курсов включают: программирование и DevOps (языки C, C#, C++, Go, Java, Python; Qt, SQL, Oracle, DevOps, и др.), сети и компьютерная безопасность, бухучет, веб-технологии (включая HTML, JavaScript и UX/UI), менеджмент, маркетинг, продажи, курсы по офисными программам (например, Excel), графика, дизайн, видеосъемка, изобразительное искусство, проектирование в САПР, управление проектами, кадровое дело, soft-skills, курсы для школьников, и даже офисных секретарей.

Качество этих курсов может быть от низкого до приемлемого: честно говоря, не пробовал эти курсы, но видел как отрицательные, так и положительные отзывы. Отсюда и такой вывод. Рекомендую пользоваться курсами «Специалиста» лишь когда аналогов нет где-то в другом месте.

§ 4.5. HTML academy

Веб-сайт курсов: https://htmlacademy.ru

Данная платформа ориентирована только на одно направление: верстка сайтов, HTML, CSS, JavaScript – в общем форнтенд. Кусок этой системы бесплатен, то есть начать заниматься можно бесплатно, но, в общем, для получения доступа ко всем обучающим заданиям придется оплатить учебу. Обучение здесь производится в онлайн-тренажерах и направлено на практику. Лекции тоже есть.

Также академия предоставляет две укрупненные программы длительностью в несколько месяцев: Фронтенд-разработчик и React-разработчик. Такие программы обещают гарантированное трудоустройство.

Академия имеет центр карьеры с вакансиями, а также так называемый «Акселератор», под которым понимается практика для верстальщиков и фронтенд-разработчиков с командой и горящими сроками, где Вы работаете над искусственно созданным проектом, а потом Вам дается обратная связь по результатам. Интересное решение.

§ 4.6. Другие русскоязычные курсы

Нетология: https://netology.ru

Достаточно большой и разнообразный перечень курсов, изначально проект был заточен на обучение маркетингу, теперь там много курсов по IT.

HackerU: https://hackeru.pro

Один из филиалов международной школы, со штаб-квартирой в Израиле. Курсы ориентированы на кибербезопасность.

Учебник и курсы по JavaScript: https://learn.javascript.ru

И множество других.

Но нам хватит перечисления и этих, иначе совсем можно запутаться в таком количестве курсов.

Если по прочтении предыдущих параграфов у Вас сложилось впечатление, что Вы не знаете куда пойти, выбирайте просто: GeekBrains или Яндекс.Практикум.

§ 4.7. Зарубежные платформы

Иностранные платформы мы не будем разбирать так детально, как русскоязычные.

Если Вы хорошо знаете английский язык, то Вы сможете прочитать всю информацию даже без записи на какие-то курсы. Однако, если Вам надо готовую, структурированную информацию, то приведем примеры популярных платформ (опять же, нижеследующий список не исчерпывающий).

Codecademy: https://www.codecademy.com

Сайт, на котором обучался бомж из предисловия. Весьма известный ресурс, с онлайн тренажерами. Позволяет изучать различные языки программирования, дизайн, некоторые другие специальности.

edX: https://www.edx.org

Это онлайн-платформа американских вузов. Существенная часть обучения бесплатна, проект некоммерческий. Платно некоторые фрагменты и сертификаты. Если знаете английский язык, возможности этой платформы неисчерпаемы. Этот портал – выбор автора данной книги.

Coursera: https://www.coursera.org

Широко известная платформа-маркетплейс различных курсов от многих поставщиков. Курсы там присутствуют по самым разным темам, а не только разработка или IT. Есть русскоязычные курсы.

Udemy: https://www.udemy.com

Обширный выбор курсов по множеству направлений, не только IT, но и различные виды творчества. Это маркетплейс множества поставщиков курсов. Там есть курсы на разных языках, включая русский.

Udacity: https://www.udacity.com

Платформа для обучения на курсах, связанных в основном с IT и другими высокими технологиями.

Глава 5. Очная учеба на курсах или в учебных заведениях

Если Вы еще только заканчиваете школу либо, давно ее закончив, собираетесь уделить очень серьезное внимание смене профессии, то очная учеба для Вас. Какие здесь варианты? Поступление в вуз впервые, в том числе на бюджетное место, получение второго высшего образования (платно), а также нечто совсем не обычное – бесплатная очная школа от Сбербанка, где в казанском филиале даже предоставляют бесплатное проживание. В предыдущей главе уже упоминалась Академия Яндекса. В ней есть очные курсы, но на начальном этапе попасть туда затруднительно.

Итак, начнем с вузов.

§ 5.1. Очная учеба в высших учебных заведениях

Если Вы можете учиться в каком-нибудь университете типа Массачусетского Технологического Института в США, то, конечно, надо выбирать этот вариант.

Однако в большинстве случаев речь будет идти о российских вузах. У нас есть много региональных университетов, как классических (где есть полный набор факультетов), так и технических (политехнических). Все их здесь не перечислить. Если в своем регионе Вы знаете отличный вуз, знаете, что образование там качественное, то можете поступать туда. Однако учеба в ведущих московских вузах будет все-таки лучшим вариантом.

Среди этих московских вузов остановимся на следующих.

МФТИ (Физтех) – предоставляет одно из лучших образований (а может и самое лучшее в России) по техническим специальностям. Как сообщает данный вуз на своем сайте, он входит в престижные рейтинги лучших университетов мира. В институте есть программы подготовки по направлениям: теоретическая физика, прикладная физика, математика, информатика, химия, биология, информационные технологии, другие науки. Поступить туда не просто, а платное обучение стоит недешево.

Сайт вуза: https://mipt.ru

МГТУ им. Н.Э. Баумана – в рейтингах российских вузов технического профиля МГТУ им. Н.Э. Баумана занимает одно из лидирующих мест. Стратегия университета, как сообщается на его сайте, направлена на подготовку кадров для передовых и высокотехнологичных отраслей науки и техники, приоритетных направлений развития экономики, в том числе: информационно-телекоммуникационные системы; индустрия наносистем и материалов; энергетика и энергосбережение; живые системы; безопасность; транспортные и авиационно-космические системы; военная техника.

Университет предоставляет программы второго высшего образования со сроком освоения 2.5-3 года. Среди них: информатика и вычислительная техника; информационные системы и технологии; программная инженерия.

Сайт вуза: https://bmstu.ru

НИЯУ МИФИ – тоже один из лучших российских университетов. Основные направления образовательной и научно-исследовательской работы включают:

– информационные технологии.

– ядерные исследования и технологии;

– лазерные, плазменные и пучковые технологии;

– СВЧ-наноэлектроника;

– нанобиотехнологии, биомедицина и медицинская физика;

Сайт вуза: https://mephi.ru

Какой бы университет Вы ни выбрали, выбирайте программу (факультет, специальность) для себя прежде всего: ту, в которой Вы, оценив перспективы, видите будущее. Если вести речь об информационных технологиях, то это такие программы или направления, как: прикладная математика и информатика, информатика и вычислительная техника, программная инженерия, и другие, связанные с IT.

Конечно, в эти передовые московские вузы не все могут поступить. А платное образование стоит не дешево. Но есть деньги и желание есть, то рекомендую. Есть и хорошие региональные вузы (Новосибирский госуниверситет, например), да и среди московских их еще много (НИУ ИТМО, НИТУ «МИСиС», например) но, как уже было сказано, из них уже выбирайте самостоятельно.

§ 5.2. Школа 21 от Сбербанка

Сайт школы: https://21-school.ru

Данная школа представляет собой последнее слово в образовательных технологиях, потому что процесс обучения

построен очень необычно. Школа не дает официальных документов об образовании государственного образца. Однако ее прохождение, как представляется, сильно меняет человека.

Посудите сами. Обучение в данной школе полностью бесплатно: финансирует школу Сбербанк. Данная школа основана на методике «школы будущего», используемой в инновационной французской школе программирования «Школа 42» (в названии нашей школы это число поделено на 2). В школе отсутствуют преподаватели, привычные оценки, лекции. Режим работы школы: круглосуточно, время учебы не регулируется. В основе методики – метод peer-to-peer («равный равному»), или коллективное обучение. В ходе обучения человек учится не просто решать практические задачи, а взаимодействовать с людьми.

Обучение проходит только очно. Оборудование в школе – компьютеры Apple. Начинается учеба с изучения языка C.

Школа на момент написания данной книги имеет два филиала: в Москве и в Казани. В Казани участникам из других городов предоставляется бесплатное проживание в общежитии.

Особых требований к поступающим нет, разбираться в компьютерных технологиях не обязательно. Документы об образовании для поступления не нужны. Возрастных ограничений, кроме минимального возраста 18 лет, нет. Для поступления нужно пройти какой-то игровой тест на сайте, собеседование и так называемый «бассейн». Самое сложное здесь – этот «бассейн», потому что он представляет собой весьма напряженное решение заданий по программированию. Тем не менее, любой может попробовать себя.

Рекомендую вариант учебы в этой школе, если у Вас нет денег, нет работы и профессия не актуальна, трудно заставить себя учиться самостоятельно, не знаете что делать. Однако Вы должны быть стрессоустойчивы в этом случае.

Глава 6. Самостоятельное освоение

Самостоятельная учеба предполагает, что Вы изучаете общедоступные курсы, видеоуроки, тренажеры, а также документацию, различные форумы, блоги, порталы, справочники. При этом без наставников, зачастую без проверки другим человеком Ваших результатов.

Приведем пару примеров как это сделать. Есть инструкция простая, а есть посложнее. Обе не влекут каких-либо существенных финансовых затрат, то есть обучение почти бесплатное.

Начнем с простой. Она предполагает начать с изучения языка программирования **Python**. Занимает 1-2 года.

Начало: Берите курс “Computing in Python” от Georgia Institute of Technology на портале https://www.edx.org

Не знаете английский в достаточной мере – вначале изучите английский.

Проходите его полностью. Изучайте SQL самостоятельно по документации, формам, видеоурокам.

Возьмите учебник по Python – только, чтобы версия там была не устаревшая. Смотрите видеоуроки из других источников (кроме edX).

Пройдите какой-нибудь платный курс по Python или бэкенд разработке, познакомьтесь с людьми, которые тоже его изучают.

Начинайте разрабатывать собственные проекты для портфолио.

Изучайте Django, JavaScript, HTML/CSS, различные фреймворки типа Angular, Node.js, React, Vue.js.

Завершите разработку своих проектов.

Поучаствуйте в каких-нибудь хакатонах.

Начинайте изучать какой-либо иной язык программирования помимо Python и JavaScript. У Вас уже будет понимание какой, *НЕ* думайте об этом в момент чтения данных строк.

Ищите работу, а потом и фриланс-проекты.

Итог: Вы становитесь джуниор фуллстек рзаработчиком.

Что-либо из этого непонятно? Значит, лучше заниматься на курсах, в вузе, школе разработчиков и так далее, а к самостоятельной учебе – переходить позднее.

Другая, более сложная. Здесь предлагается начать с изучения языка программирования **C** (Си). Занимает 1-3 года.

Начало: Берите книгу «Язык программирования С», авторы: Керниган Брайан У., Ритчи Деннис М.

Читайте ее полностью и выполняйте все упражнения.

Если испытываете затруднения при выполнении упражнений (а скорее всего они будут), берите книгу «Язык Си. Книга ответов», авторы: Тондо К., Гимпел С.

После изучения этих книг переходите ко книге «Язык программирования С++», автор: Бьерн Страуструп.

Изучите устройство операционной системы Linux, разработки в ней.

Вам надо будет здесь заняться изучением математики, если у вас не техническое образование и прокачать как всю алгебру, так и тригонометрию и геометрию, а также мат. анализ, мат. статистику, теории множеств и вероятностей, и так далее.

Далее изучайте алгоритмы и структуры данных: на курсах или по книгам. Здесь уже Вы сами поймете какие источники Вам надо.

Попробуйте написать какие-либо полезные приложения для Linux или Windows, возможно open-source. Начинайте формировать портфолио.

Начинайте изучать иной язык программирования помимо С и С++. Вам надо будет определяться чем Вы будете заниматься: разработка на C#, или Objective-C, например. А может, станете Java или Go разработчиком. Также

определяйтесь, что вы будете разрабатывать: программы для микроконтроллеров, мобильные приложения, десктоп-приложения, интернет-сервисы. Сразу скажу, что думать обо всем этом *НЕ надо* пока не выполните первые шаги, поэтому начало – это обозначенные в самом начале две книги.

Создайте мобильное или десктоп приложение, дополните им свое портфолио.

Ищите работу, а потом и фриланс-проекты.

Итог: Вы становитесь универсальным джуниор разработчиком, либо разработчиком мобильных приложений, либо десктоп-приложений, либо каких-то сетевых приложений.

Что-либо из этого непонятно? Значит, лучше заниматься на курсах, в вузе, школе разработчиков и так далее, а к самостоятельной учебе – переходить позднее.

Можно приводить и другие примеры. Здесь все зависит от Ваших целей, чего Вы хотите добиться, какое направление выбираете.

К примеру, вот отдельные ресурсы, которые пригодятся в обучении тем или иным языкам программирования, технологиям:

Python и другое: http://www.openbookproject.net/books/

PHP: https://www.php.net/manual/ru/index.php

JavaScript: https://developer.mozilla.org/ru/docs/Web/JavaScript

SQLite – документация: https://www.sqlite.org/docs.html

Разные языки программирования и технологи: https://www.w3schools.com

Язык Go: https://golang.org/doc

Kotlin: https://kotlinlang.org/docs/

Swift: https://swift.org/documentation/

Практически любая технология или язык программирования имеет официальную документацию. Перечислять эти ресурсы можно очень долго. Этот путь самый сложный, выбирайте его только если у Вас высокая самодисциплина:

например, вы не зависаете в смартфоне, в дзене, в YouTube и прочих подобных местах. Впрочем, бороться с цифровой зависимостью всегда полезно, но эта не тема данной книги.

Глава 7. Вы становитесь разработчиком

Что же делать, когда Вы уже чему-то научились? Делать то, что Вы хотели, для чего Вы учились. Это может быть работа по полученной специальности разработчика или иного IT-специалиста. Это может быть запуск собственного проекта (свое коммерческое приложение либо создание чего-то не для продажи). Это может быть фриланс, выполнение работ по заказам, что тоже имеет шанс перерасти в полноценный бизнес.

Еще одной интересной вещью является участие в хакатонах, бизнес-акселераторах, продолжение своей учебы и развития, основание своей компании. Об этом мы поговорим в следующей главе.

Здесь же проведем краткий обзор путей начала своей профессиональной деятельности: работа по найму, свой проект, работа по заказу.

В главе 3 мы уже упоминали, но повторим здесь для лучшего понимания, что по уровню опыта IT-профессионалов часто классифицируют следующим образом:

Стажер (**Intern**) – совсем нет опыта, еще только заканчивает учебу и практикуется где-то.

Джуниор (**Junior**) – учебную программу освоил и имеет весьма мало опыта или практически не имеет, однако вполне способен выполнять работу.

Миддл (**Middle**) – средний уровень, уже есть нормальный опыт работы, но пока еще не достиг вершины мастерства.

Сеньор (**Senior**) – опытный и высококвалифицированный, ему не нужны уже советы, он разбирается во многих вещах в своей области.

Лид (**Lead**) – высококвалифицированный руководитель команды разработчиков.

Если Вы работаете по найму, или фрилансер, то данная классификация будет применима к Вам. Если у Вас свой творческий проект или бизнес, то она может быть применима, но в общем-то в этом случае зачастую Вы сами себя будете оценивать по этой шкале.

§ 7.1. Трудоустройство

Работа по найму – это самый простой вариант, с него рекомендую начинать свою новую профессиональную деятельность. Поработать по найму также считаю нужным до того, как хотите заняться фрилансом или своим бизнесом.

Основные вопросы здесь: как найти эту работу и как сделать, чтобы меня на нее приняли? Найти работу не составит труда, вакансий по IT-специальностям огромное количество. Условия работы самые разнообразные. А вот для того, чтобы Вас взяли на эту работу без опыта, придется постараться.

Искать рекомендую на сайте hh: https://hh.ru

На сегодня это практически основной канал поиска работы в России. Конечно, есть и другие сайты, но я плохо представляю, кто и в каких случаях ими пользуется. Если Вы знаете таковые, то можете попытать счастья там, но вижу смысл обязательно размещать резюме на hh.

Итак, Вам нужно будет составить хорошее резюме. По этому поводу написано огромное количество литературы, статей в интернете. Приведу лишь несколько советов.

1. Описывайте наиболее релевантный опыт. Если у Вас не было опыта в IT, лучше указать Ваши собственные разработки каким-то образом, чем детально перечислять не относящееся к вакансии места и выполнявшуюся работу.

2. Не пишите эти избитые и никому не нужные фразы типа «стрессоустойчивый», «целеустремленный», сегодня это выглядит как-то несуразно. Лучше заострите внимание на каких-то ключевых, выдающихся способностях, навыках, знаниях. Да, навыки из области soft-skills (коммуникативные) тоже можно отразить, но только делать это надо, если Вы действительно ими обладаете, например, имеете навык и опыт ведения сложных переговоров с заказчиками разработки программных продуктов.

3. Описывайте свой опыт или профессиональные знания четко и ясно: у человека, который будет читать Ваше резюме есть от нескольких секунд, до нескольких минут, чтобы увидеть что-то внятное и цепляющее внимание. Поскольку мне самому приходилось подбирать персонал для разных позиций, я знаю, о чем говорю.

4. Загружайте адекватную, нормальную фотографию, не обязательно в деловом костюме. Резюме без фото, а равно с каким-то странным фото – это сегодня не годится.

5. Указывайте только самые важные способы связи с Вами – 1-3 способов связи достаточно. Очень желательно, чтобы предпочитаемым способом связи был мобильный телефон, однако e-mail тоже должен обязательно быть.

6. Рекомендации в тексте резюме сегодня указывать не целесообразно.

7. В резюме рекомендуется привести примеры работ, ссылки на портфолио.

Резюме – это только начало. Далее надо откликаться на вакансии. И, конечно, резюме должно быть видимым для всех зарегистрированных работодателей, не надо делать его скрытым или частично скрытым, часто работа находит требуемого специалиста сама.

Откликов должны быть **сотни**. Если в Вашем регионе нет соответствующих вакансий – ищите в других, в Москве и Санкт-Петербурге.

Рано или поздно Вы попадете на собеседование. Главное правило здесь: не обманывать, чувствовать себя уверенно.

Таким путем, в конце концов, Вам удастся найти первую работу. А далее Вы уже будете ее менять через какое-то время или переходить на фриланс. Все будет повторятся снова. При этом опытному разработчику будет устроиться проще, а зарплата будет расти со временем.

§ 7.2. Собственная разработка

Создание своего проекта – это не фриланс. Под этим подразумевается нечто совсем иное. Вы создаете, например, востребованное пользователями мобильное приложение, или веб-сервис, и «монетизируете» его, то есть или продаете приложение, или доступ к сервису, или предоставляете его бесплатно, но с рекламой и так далее. Способы монетизации могут быть различными.

А может Вам и вовсе не хочется монетизировать свой проект. Вы создаете какой-нибудь программный продукт с отрытым исходным кодом и делаете вклад в развитие информационных технологий человечества. Выбор за Вами.

Надо сказать, что создание своего проекта – мне лично нравится больше всего. Потому что это настоящее творчество. Работа по найму не является настолько творческой, там все-таки Вы реализуете придуманную кем-то идею. В своем проекте Вы реализуете свою идею.

Вариант своего проекта, на мой взгляд, это не бизнес – по крайней мере на начальной стадии. Если проект окажется невероятно популярным и, что называется, «выстрелит», то да, он имеет шансы перерасти в бизнес какого-то масштаба, может быть даже и крупного. Но таких проектов считанные единицы.

Свой проект – это, прежде всего, саморазвитие, творчество, работа на себя. Он может приносить доход напрямую (платное приложение, покупки в приложении, реклама, иные методы) или косвенно (например, посмотрев на Ваш проект, работодатели или заказчики захотят, чтобы Вы выполняли работу для них). Но главная цель в другом, о чем здесь уже было сказано: творчество и развитие Вашей личности, а также сделать что-то полезное для многих людей.

Илон Маск считает, что сделать что-то для пользы многих людей это лучшее, что человек может сделать, пусть это даже будет какая-то мелочь, маленькое приложение или компьютерная игра. Этот совет звучит как логичный и понятный, весьма рекомендуется к использованию в Вашей профессии.

Если Вы что-то делаете для других, на самом деле Вы делаете это для себя. И результат этой работы, к тому же, у Вас невозможно отнять. Может быть, Вам сложно это понять, но это так. Просто делайте, поймете потом (я надеюсь).

Важно делать, что Вы можете и как можете, используя те навыки и ресурсы, какими Вы сейчас располагаете. Не надо пытаться создать сразу какой-то невероятный проект, делайте MVP (минимально жизнеспособный продукт от англ. *minimum viable product*).

§ 7.3. Фриланс

Фриланс – это работа на заказчика без трудоустройства. Вы ищите заказчиков, а они Вас – на биржах фриланса, в соцсетях, иным способом.

Сегодня легко можно стать самозанятым или оформить ИП (то есть, стать индивидуальным предпринимателем). Чтобы получать нормальные заказы, нужно официально вести свою деятельность, так как компаниям выгодно, чтобы у Вас был соответствующий статус.

Если Ваш доход от фриланса не превышает 2.4 миллиона рублей в год, то регистрация самозянытым (а точнее, если называть это правильно – начать применять режим налога на профессиональный доход) является самым простым решением. Какой-то сложной отчетности или учета у этой системы нет, все легко делается в специальном приложении.

Если доход большой, то нужно оформлять ИП или ООО (регистрировать компанию). Но, думаю, если у Вас большой доход, то Вы и так уже знаете что делать.

Наверное, самые известные биржи фриланса:

Российская: https://www.fl.ru

Зарубежная: https://www.upwork.com

Не буду расписывать как работать на этих биржах. Скажу только, что это сложно. Там куча разных правил, ограничений, рейтинги, и так далее. Надо платить за свое продвижение. Поэтому-то я и рекомендую начинать с работы по найму, а если есть свои идеи – со своего проекта.

Эпилог. Не останавливаясь на достигнутом

Прочитав эту книгу, как думается, Вы теперь знаете что делать. Если Вы хотите стать разработчиком, Вам только осталось выбрать подходящие для себя методы освоения и выполнить рекомендации. Дело техники!

Все ли так просто? Да, выполняя определенные шаги, вы дойдете до своей цели. Это неизбежный успех, если будете выполнять шаги. Если же не будете, то программистом не станете. И одновременно нет, не все так просто. Вам придется много работать (я не имею в виду работу в узком смысле слова – трудоустройство). Вам придется много изучать, много уделять внимания своим разработкам, фокусироваться на этом. Без фокусировки в определенном направлении, вряд ли Вы добьетесь чего-то серьезного в нем.

В этой заключительной части книги, хотелось бы еще сказать об интересных вещах, которые можно делать помимо работы, фриланса и своих проектов. Одна из них – участие в хакатонах. Хакатоны – это сборища IT-специалистов, которые за пару дней должны решить какие-то задачи и представить публике свои проекты. Задача требует применения максимума своих возможностей, как разработчика или иного профессионала, так и как человека, способного ладить с людьми и работать в команде. Один из самых масштабных таких хакатонов, «Цифровой прорыв»: https://leadersofdigital.ru

По ходу своей карьеры не забывайте о своем здоровье и прочих аспектах жизни. Между прочим, Вы знаете что существуют стоячие и ходячие рабочие места? Что за

ходячие рабочие места – это те, у которых есть беговая дорожка, так что программист не сидит, а идет во время работы – дело в том, что это хорошо для здоровья. Возможно, в будущем Вы захотите сделать себе такое, и, наверное, это будет хорошо, но на начальном этапе не вдавайтесь в такие сложности.

На всякий случай: **e-mail для связи с автором**: aleksandervankov@gmail.com

Желаю Вам успехов в разработке ваших приложений и прочих проектов!

Благодарю за внимание к данной книге. Если Вы нашли пользу в этой книге, автор будет рад получить пожертвование, потому что цена книги мала, и большая ее часть расходуется на издание. Если же Вы реально воспользуетесь советами этой книги, то выгода для вас будет высокой. Если желаете, вот этими способами можете это сделать:

1. Перевести с карты:

https://money.yandex.ru/to/410014828500050

2. Яндекс.Деньги: 410014828500050


STARTUP
PLAYBOOK