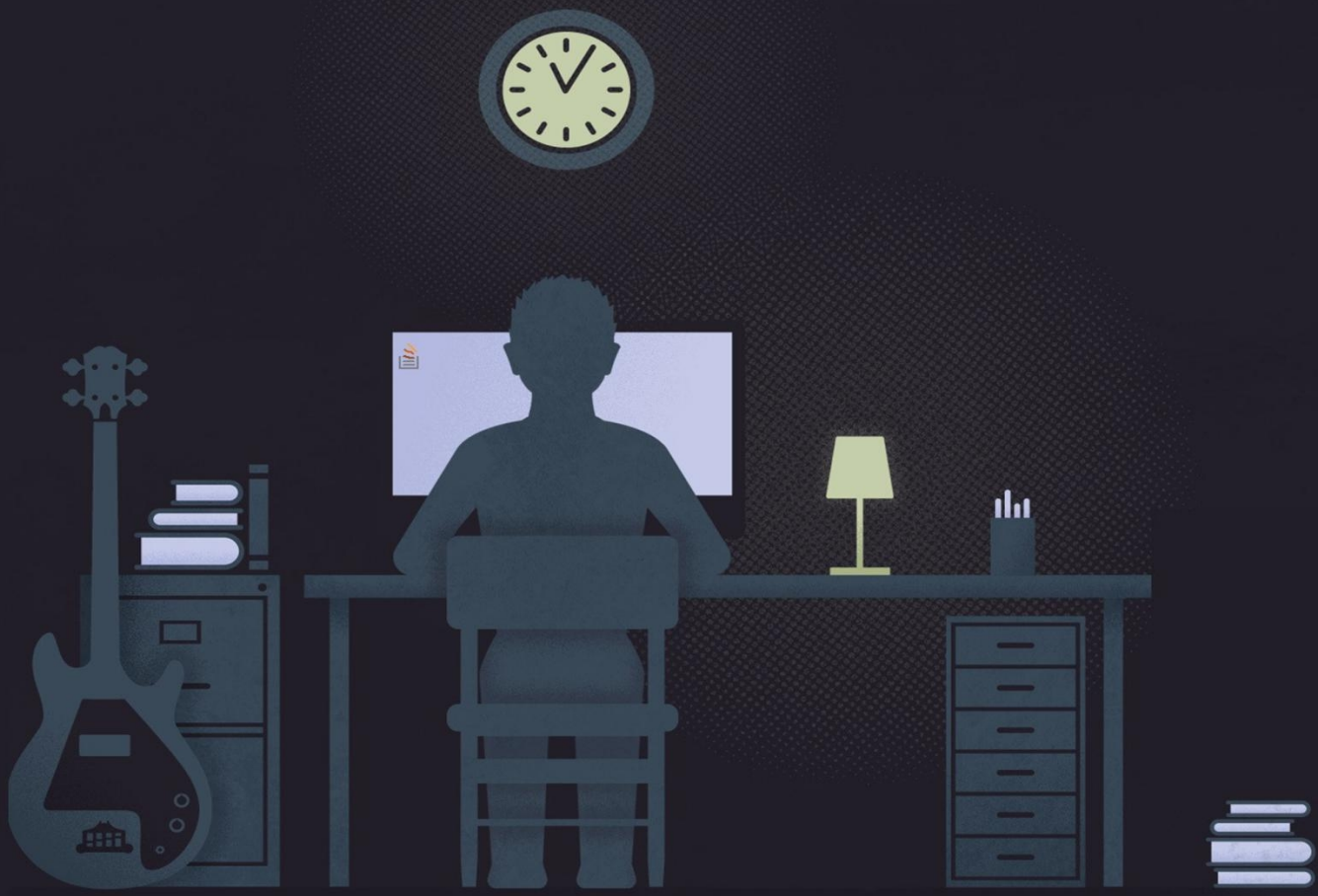
КОНСТАНТИН ШЕРЕМЕТЬЕВ

Как эффективно изучать программирование

ПУТЬ В ПРОГРАММИСТЫ



Содержание

Введение

В этой мини-книге мы поговорим о возможности самостоятельно изучить программирование, с чего начать и в каком направлении двигаться, чтобы стать профессиональным программистом.

Пару слов о себе. Я семь лет работал программистом в космической отрасли, потом семь лет преподавал программирование в ВУЗе. Защитил кандидатскую диссертацию и стал кандидатом технических наук в области программирования. Параллельно с этим я писал много коммерческих программ на заказ. Поэтому программирование – это моя профессиональная тема.

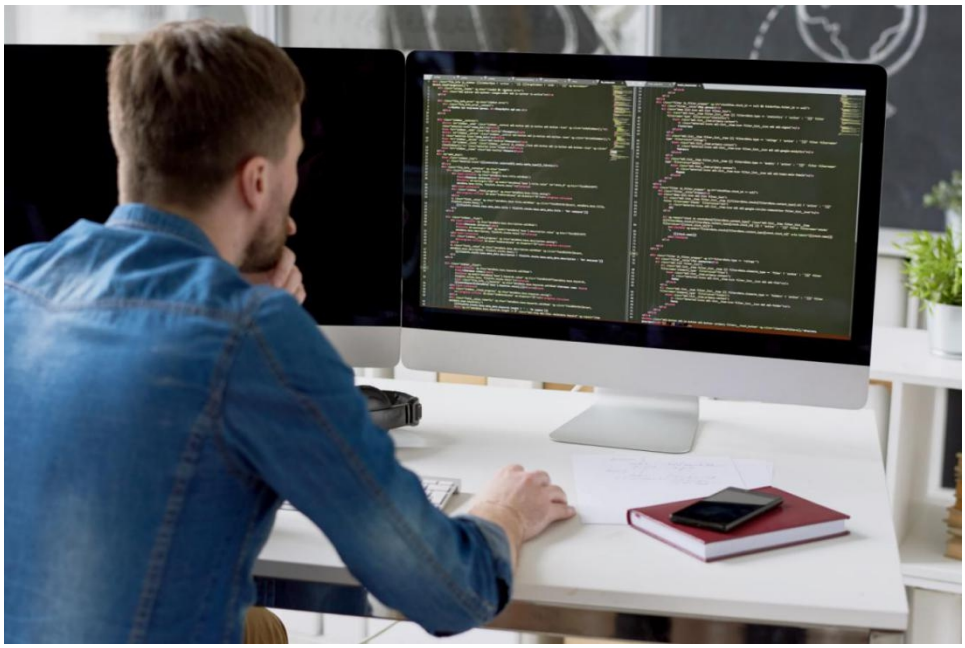
Кроме того, так как я обучал программированию сотни студентов, то я прекрасно знаю все тонкости обучения. В этой мини-книге я расскажу и отвечу на самые часто задаваемые вопросы:

- Как становятся программистами?
- Кто может научиться программированию?
- Каковы требования к будущему программисту?
- Прочие вопросы, связанные с обучением программированию.

Глава 1. Зачем становиться программистом?

Одна из самых востребованных на сегодняшний день профессий – это профессия программиста. Причём есть странный парадокс, который состоит в том, что программистов все время не хватает.

Обычно, если какая-то профессия становится дефицитом, то люди сразу стремятся ее приобрести. Открываются курсы, и очень быстро спрос закрывается. Но с программистами этого нет. Дефицит специалистов в области IT настолько высок, что даже не очень квалифицированный программист может легко найти себе место. И это при том, что программисты имеют очень высокие оклады: в Москве – от 100 тысяч рублей и выше.



Почему так происходит? Я отвечу на этот вопрос в главе 4, а сейчас поговорим о преимуществах профессии программиста, которые следуют из этого парадокса.

Несмотря на кризис, зарплаты программистов совершенно не уменьшаются, и поэтому самый лучший способ обеспечить себя востребованной профессией – хорошо оплачиваемой, уважаемой и перспективной – это стать программистом. А если вы собираетесь работать за рубежом, то профессия программиста – это тот редкий случай, когда вы можете реально устроиться по специальности, даже не очень ориентируясь в другой стране, потому что по другим профессиям там нужно переучиваться. В программировании этого нет.

Расскажу одну историю, которая меня в своё время очень поразила. Один мой знакомый, который уже вышел на пенсию, работал программистом. Его дети давно эмигрировали в Америку и позвали его к себе. И он поехал в Америку. При этом он совершенно не знал английский. И какое же у меня было удивление, когда я получил от него письмо, в котором он с изумлением сообщает, что очень быстро и легко нашёл работу, не зная языка. А какую работу он нашёл? Он зашёл в вакансии, указал, что знает Basic, и его тут же приняли на работу в ближайший супермаркет. В чём же состояла его работа? Для этого супермаркета написана большая база данных, работает всё хорошо, но начальству нужен иногда какой-нибудь нестандартный отчёт или какая-то выборка. Ему это поручают, и то, что он так слабо знает английский язык, в общем-то, не проблема, потому что они говорят: «Мы никак не можем найти программиста на такую работу». Таким образом, он поехал за рубеж и так хорошо устроился, потому что знал программирование.

Поэтому, если вы хотите иметь такую перспективную профессию, то вам прямая дорога в программисты. Но возникает вопрос: а получится ли у вас? Об этом в следующей главе.

Глава 2. Можете ли вы стать программистом?

Начну с очень показательной истории. Когда я работал доцентом и преподавал программирование, то в одном из курсов я обучал программированию сайтов.

В любой группе всегда бывают студенты-лентяи. И в одной группе было три таких лентяя, которые на лекциях болтали между собой и на практике тоже ничего не хотели делать. К ним я применил простой педагогический приём. Когда на практике все сидят перед компьютерами, они сидят как раз по трое, и я именно этим трем лентяям тихо делал подсказки. В результате они, естественно, выполняли задания первыми. Я их хвалил и говорил: «Какие молодцы!».

Это было достаточно весело. Конечно же, учиться они не хотели и, когда закончился семестр, я им поставил по тройке и думал, что на этом все и кончится.

Но через некоторое время, буквально через неделю после окончания семестра, вдруг они ко мне заходят на кафедру. Сели, говорят:

- Константин Петрович, у нас к вам такое предложение…
Мне стало интересно, говорю:
- Заинтриговали. Какое предложение?
Они говорят:
- Понимаете, это же пятый курс, и нам уже сейчас надо устраиваться на работу, а мы ничего не знаем, ничего не умеем. Но вы знаете, у нас стало что-то даже получаться в программировании сайтов. Мы сейчас поговорили – такие деньги там платят! Давайте, вы с нами позанимаетесь отдельно?
На что я им сказал:
- Нет, ребята, вы своё профукали. Надо было заниматься в семестре, а сейчас у меня на вас времени уже нет.

Почему я рассказал эту историю, почему она для вас важна? Потому что даже если хулиганы и лентяи, которые буквально раз в неделю приходили на практику и как-то одной ногой что-то делали, и то у них что-то получилось, то, если вы подойдёте к вопросу серьёзно, то, конечно же, научитесь.

По своему опыту я знаю, что и подростки, и пенсионеры, и мужчины, и женщины, и серьёзные руководители, и молодые мамы с детьми – все прекрасно изучали программирование.

Почему? Потому что программирование не требует ни какого-то супер-мышления, ни какого-то супер-таланта, ни какой-то супер-памяти. Всё, что оно требует – это некоторой усидчивости и определённого желания изучить программирование. Мои лекции по программированию были популярны, и на лекцию приходили другие преподаватели – одному было 30 лет, другому – 60, которые до этого с программированием вообще не сталкивались. И что интересно: они записывали, они пытались что-то делать, и к своему удивлению видели, что всё получается. И один из них так увлёкся, что стал вставлять примеры, связанные с программированием, в свои лекции по высшей математике, потому что есть пересечение высшей математики и программирования. И поэтому не нужно переживать о том, есть ли у вас способности.

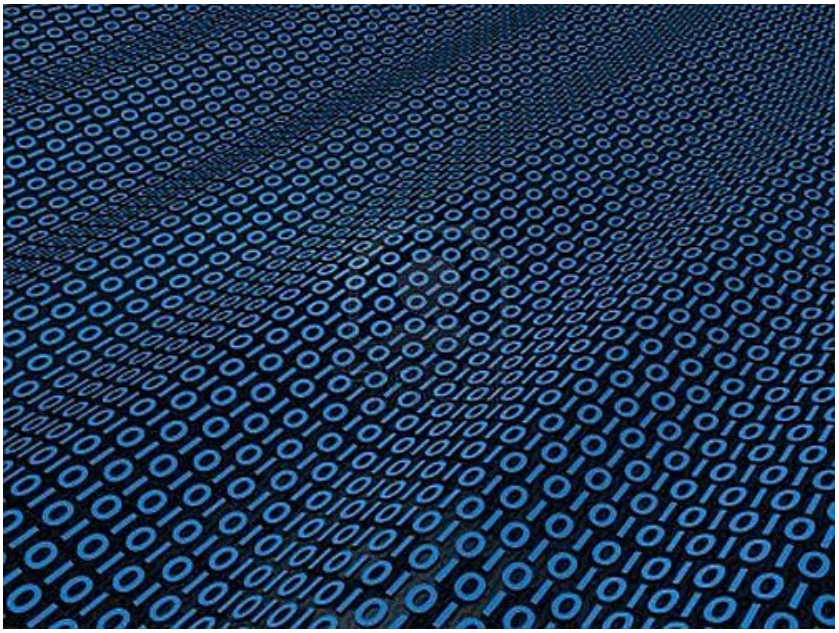
Для того, чтобы стать программистом, никаких выдающихся способностей не нужно. Если есть желание и готовность учиться, то вы гарантированно научитесь программировать.

Естественно, возникает вопрос: а к самому процессу обучения есть какие-то особые требования? Об этом в следующей главе.

Глава 3. Есть ли особые требования к изучению программирования?

Здесь у меня опять хорошие новости. Обучение программированию – это не рулетка. Не бывает так, что десять человек начали учиться, но только одному повезло.

Для овладения профессией программиста не требуется железной воли или полной самоотдачи. В основе компьютера лежит примитивная двоичная логика. То есть, «да-нет» или «единица-ноль». Любой вменяемый человек, начиная с подростка и заканчивая пенсионером, может совершенно элементарно разобраться в азах программирования, потому что это полный примитив. Поэтому, если вы серьёзно настроились и начинаете идти в сторону обучения программированию, то вы абсолютно гарантированно через некоторое время станете программистом.



Единственное требование к программисту – у программиста должно быть развито так называемое алгоритмическое мышление.

Алгоритмическое мышление – это способность решать задачи путём точного алгоритма действий. С алгоритмическим мышлением вы сталкивались, когда видели рецепт любого блюда, в котором написано: пожарить морковку, пожарить лук, порезать мясо, положить вот это, положить вот это, и вы получите в конце готовое блюдо. Рецепт – это и есть алгоритм.

Но один барьер для обучения всё-таки есть. О нем мы поговорим в следующей главе. Этот барьер, к счастью, не имеет отношения ни к компьютеру, ни к программированию. Это барьер психологического плана.

Главное требование к изучению программирования – это упорядоченность процесса. То есть для того, чтобы эффективно изучать программирование, нужно это делать каждый день. Пусть полчаса, пусть даже 15 минут, но каждый день. В отличие от каких-то других вещей, например, если вы учитесь играть на гитаре, вы можете немножко поиграть, потом сделать большую паузу, потом опять начать играть.

Если случае с гитарой вы не теряете навык, то в программировании все наоборот. Чтобы сохранять в себе алгоритмическое мышление, необходимо регулярно его практиковать. После перерыва нужно опять восстанавливать в себе навыки алгоритмического мышления. Если же перерыва нет, если вы каждый день хотя бы по чуть-чуть занимаетесь программированием, то вы начинаете двигаться вперёд достаточно быстро и эффективно. И теперь перейдём к большому барьеру, который может помешать вам стать программистом.

Глава 4. Большой барьер программиста

Это самая важная глава, которая как раз и определяет, получится ли из вас программист.

Есть определённая психологическая проблема, связанная с программированием, которую я называю:

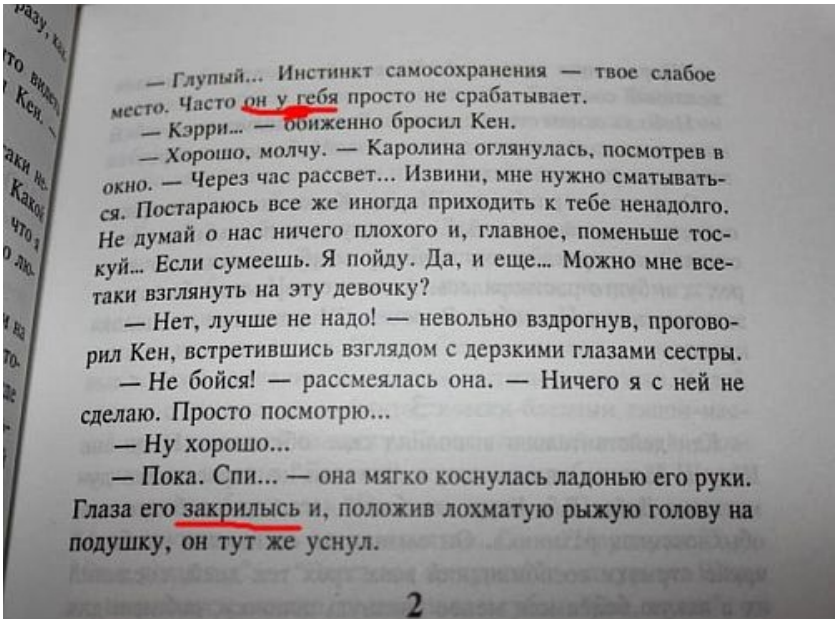
Большой Барьер Программиста

Я наблюдал много раз такую ситуацию: студенты заканчивают ВУЗ и дальше делятся на две части. Одна часть группы поступает на должность программиста с хорошим окладом, а другая часть группы идет в курьеры, охранники, официанты и так далее. Как же так? Ведь они учились пять лет! И затем идти охранником?

Да, виноват в этом именно Большой Барьер. Именно из-за него всегда будет дефицит программистов.

Давайте разберёмся, в чём состоит этот большой барьер программиста.

Представим себе, что писатель написал книгу, но в этой книге оказалось несколько орфографических ошибок. А теперь вопрос: критично ли это? Вообще не критично, 99% читателей вообще не обратят внимание на эти ошибки. Подумаешь, орфографическая ошибка!



— Глупый... Инстинкт самосохранения — твое слабое место. Часто он у тебя просто не срабатывает.

— Кэрри... — обиженно бросил Кен.

— Хорошо, молчу. — Каролина оглянулась, посмотрев в окно. — Через час рассвет... Извини, мне нужно сматываться. Постараюсь все же иногда приходить к тебе ненадолго. Не думай о нас ничего плохого и, главное, поменьше тоскуй... Если сумеешь. Я пойду. Да, и еще... Можно мне все-таки взглянуть на эту девочку?

— Нет, лучше не надо! — невольно вздрогнув, проговорил Кен, встретившись взглядом с дерзкими глазами сестры.

— Не бойся! — рассмеялась она. — Ничего я с ней не сделаю. Просто посмотрю...

— Ну хорошо...

— Пока. Спи... — она мягко коснулась ладонью его руки. Глаза его закрылись и, положив лохматую рыжую голову на подушку, он тут же уснул.

2

Точно также, в любой другой профессии. Небольшие ошибки, опечатки, шероховатости, нестыковки никак не влияют на основную работу. Но с

компьютером все не так. Программа может работать только при условии, что никаких орфографических ошибок в ней нет.

Компьютер – абсолютный перфекционист. Приведу пример из своей практики.

Когда я уже был программистом и работал на больших ЭВМ, которые занимают целый этаж, я впервые увидел персональный компьютер. Он тогда безумно дорого стоил. Я был допущен, чтобы с ним разобраться. У меня в руках учебник. Я ввожу первую команду из учебника. Эта команда выполнилась, всё хорошо. Я беру следующую команду, она не работает. Я возвращаюсь к предыдущей команде, которая только что сработала, она тоже не работает. И я начинаю вводить все команды подряд. На все вводимые команды компьютер пишет: «Error». И я и так, и сяк пытаюсь. И вообще не понимаю, что происходит. Первая мысль: «Я его сломал, что ли?» Пришлось позвать человека, который отвечал за компьютер. Он подошёл и ткнул пальцем в мою ошибку. Но смотрите, какая в этот момент была для меня ситуация: что бы я ни делал, ничего не получается, компьютер всё время говорит: «Ошибка, ошибка, ошибка»… Но я был спокоен, потому что знал, что так или иначе, я найду, где ошибка.

Большинство же людей подобная ситуация может просто взбесить. Именно это и есть Большой Барьер. Вы вроде все делаете правильно, а программа не работает.

Посмотрим, как преодолевается Большой Барьер. Начнем с бытового примера – испечь блины. Представьте себе: домохозяйка печет блины, вдруг срочный звонок, она увлеклась разговором и забыла про блин на сковороде. Блин начал гореть. Она врывается на кухню, там дым коромыслом. И вот теперь вопрос: что произойдёт с этой домохозяйкой, она бросит печь блины или нет? Вы скажете: «Нет, конечно. Она выбросит остатки этого сгоревшего блина, почистит сковороду, проветрит кухню и продолжит печь блины. Всё нормально, бывают ошибки».

Как вы понимаете, на кухне такое бывает, но редко. А теперь представьте, что с каждым блином что-то не так. Этот блин сгорел, тот недопекся, другой упал на пол. Много ли домохозяек выдержит? Вот именно так и происходит с программой. Все время находятся ошибки.

А ведь в программе никакие ошибки недопустимы. Каждая программа должна быть на 100% правильной и точной. Если это не так, то компьютер всё время будет сообщать об ошибке. Людям, далеким от программирования, кажется, что работа программиста в основном заключается в том, чтоб писать программы. Нет. Писать программы – это примерно десятая часть работы программиста, а 90% его труда заключаются в том, чтобы искать ошибки в программах. Это так называемый процесс отладки программы. В профессии программиста ошибки – это нормальная часть процесса.

Поэтому когда обычный человек сталкивается с тем, что каждые пять минут новая проблема и никак нельзя двинуться дальше, потому что:

• ничего не получается,
• и опять не получается,
• и снова не получается…, то людей это злит, бесит, выводит из себя. Они говорят: «Всё, я больше этим заниматься не буду!» И я это наблюдал много раз.

Какие же люди проходят этот Большой Барьер? У этих людей есть боевой настрой. Они относятся к ошибке, как детектив относится к тому, что нужно найти преступника. Да, непонятно, как его найти, но в этом и состоит

интрига. Ситуация, когда ничего не получается, их только заводит. И в том примере с персональным компьютером это меня лишь раззадорило, потому что был единственный вопрос: подождите, а так же не бывает! Только что всё работало, и вдруг перестало работать. Так не бывает. Конечно, поэтому я стал спрашивать у другого человека. Он бы не ответил, я бы взял другую книжку. Но проблему все равно решил бы.

В коллективе программистов часто бывают ситуации, когда кто-то из коллег зовет: «Ребята, помогите! Я вообще ничего не понимаю, что происходит». Все собираются вокруг него, потому что всем интересно, все понимают: раз человек позвал, то дело серьезное.

Все подошли к компьютеру, начинают смотреть в экран и говорить: «Вот это попробуй. Вот это». Вдруг кто-то говорит:

– Да вот же ошибка!
– А, точно, я не увидел!
Все радостные расходятся по рабочим местам.

Особенность тех людей, которые могут стать программистами, заключается в том, что когда они один на один с компьютером и у них ничего не получается, у них всё равно азарт, у них всё равно желание победить и боевой настрой. При таком настрое путь программиста – это легко и интересно. Потому что компьютер – это устройство очень простое, которое готово выполнить любую команду. Если вы эту команду написали правильно, то она будет выполнена. Поэтому успех программиста закономерен: все правильно написал – все заработает.

Поэтому для того, чтобы преодолеть Большой Барьер, нужно воспитывать в себе этот боевой настрой и говорить себе: «Да, у меня всё получится. Да, я буду сидеть час, я буду сидеть 10 часов, я буду сидеть месяц, но я всё равно с этой проблемой разберусь, я её решу». И, соответственно, если у вас такой настрой есть, то вы можете стать программистом.

Но чтобы вы себе не льстили, я предлагаю вам такой экспресс-тест: пройдёте ли вы Большой Барьер программиста или нет.

Вспомните те ситуации, когда у вас или с компьютером, или с мобильным телефоном что-то не получалось.

Как вы себя вели? Всё-таки старались разобраться, в чём дело? Пробовали разобраться или злились и говорили: «А, ну его!»? Если вы привыкли к тому, что нужно подумать и разобраться, то у вас есть все шансы пройти Большой Барьер.

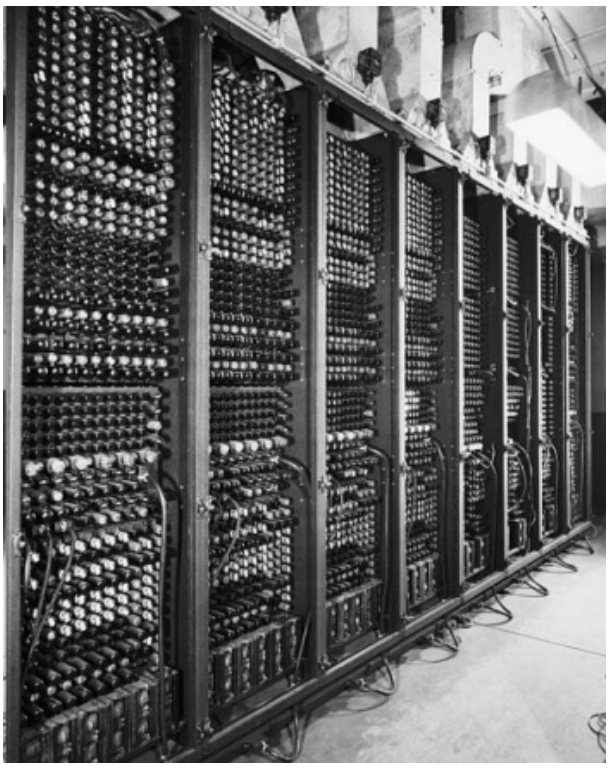
Глава 5. Нужно ли официальное образование?

Нужно ли вам образование для того, чтобы стать программистом?

Многие считают, что для овладения программированием нужно поступить в ВУЗ, отучиться пять лет, получить диплом – и вот только тогда вы стали программистом. У меня для вас хорошая новость: это совершенно не обязательно. Скажу больше, это может даже помешать вашему пути программиста. И сейчас расскажу свой опыт.

Я намеренно шел в ВУЗ, именно на факультет электроники, именно на программиста, это был сознательный выбор, мне очень хотелось, хотя я был в этом плане совершенным нулём.

Когда я поступал в ВУЗ, там был вычислительный центр, и было в этом вычислительном центре два компьютера: один компьютер занимал весь первый этаж, второй компьютер занимал весь второй этаж.



Я тогда ничего не понимал в компьютерах, но мне очень хотелось научиться. И мой опыт по изучению программирования был сначала безумно неудачным.

Почему? Потому что уже шла середина семестра, у меня по всем другим предметам все хорошо и все замечательно, кроме программирования. То есть в программировании я не понимаю ничего. Я не мог решить ни одно

домашнее задание, ни одной задачи. И мне приходилось списывать у соседей для того, чтобы это всё хоть как-то сдать. А понимать – я ничего не понимал. Я был в состоянии близком к панике, потому что я ведь поступил на специальность по программированию, а сам ничего не понимаю в программировании. И я делал разные попытки понять программирование, но книги по программированию были большими и непонятными.

Я нашел свой путь в программисты, когда устроился подрабатывать техником на вычислительный центр (ВЦ). Они набирали студентов для вечерних и ночных дежурств. Работа заключалась в простых действиях: менять жёсткие диски, запускать процессы тестирования, следить за журналом выполнения и прочее.

Иногда на ВЦ приходили студенты со старших курсов, которые запускали большие программы. В свободное время я брал распечатки этих программ и смотрел их. Пытался что-то понять. И вдруг, в какой-то момент мне приходит озарение: «Да всё же просто, всё же примитивно!»

Почему была у меня проблема именно с программированием? Из-за неверного преподавания. Женщина, которая вела предмет, давала кучу лишней информации: системы счисления, форматы данных, характеристики компиляторов…

Это к программированию имеет косвенное отношение. Суть программирования в алгоритмизации процесса решения задач. То, что нам преподавали, может показаться интересным, но эта информация мне потом в жизни вообще никогда не пригодилась.

А вот практика на вычислительном центре была очень полезна. Разбор работающей программы, понимание того, какой алгоритм был выбран – это и есть главный ключ к программированию.

Как только я это понял – с этого момента для меня программирование, которое нам преподавали – это было «тьфу», и я даже сделал такой эксперимент: я не стал готовиться к экзамену по программированию. Перед экзаменом я специально записался на ночное дежурство на ВЦ. Всю ночь продежурил и утром, совершенно не спав, пошёл на экзамен по программированию. Пошёл отвечать без подготовки, сразу же, первым, тут же ответил на все вопросы, получил «пятёрку», гордо ушёл и пошёл спать. Почему? Да потому что, когда я уже это понял, я в вычислительном центре сам стал писать совершенно простенькие тестовые программки: проверял так, проверял эдак и очень быстро набил руку. Я случайно открыл для себя кратчайший путь к изучению программирования и потом с успехом использовал его, когда сам стал учить студентов.

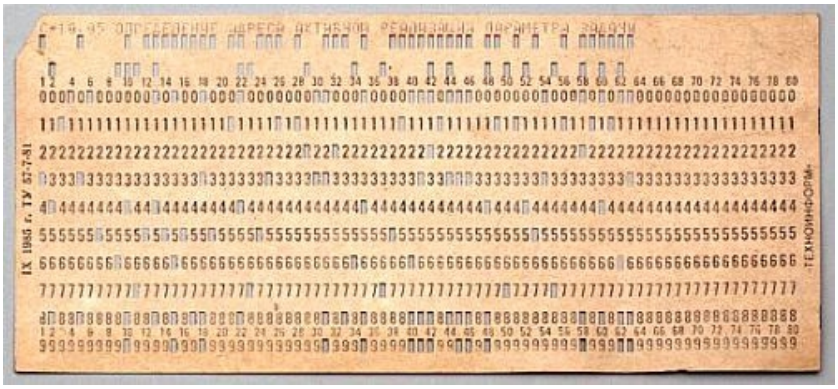
Суть метода в том, что нужно сразу начинать писать программы по принципу: «Делай, как я». Вот задачка, вот решение. Повтори решение и всё поймешь.

Лично в моем случае ключевые знания по программированию мне дал не институт, а практика на ВЦ. Причем на самом ВЦ меня тоже никто не учил, более того, мне даже не разрешали запускать свои программы, но я учился по чужим. А пробовал я в общем порядке, отдавал перфокарты в очередь и получал на следующий день. Но я постоянно что-то пробовал. В этом и состоит главный секрет быстрого обучения программированию.

В следующей главе мы будем говорить об организации занятий. Но сейчас вы должны понять и запомнить, что программисту для обучения нужны только две вещи: компьютер и хороший курс по программированию. Программист работает только с компьютером. И любой язык программирования, любую технологию программирования вы можете в совершенстве освоить полностью самостоятельно.

Есть ещё такая запугивающая тема: на форумах, когда новичок спрашивает: «А можно научиться программированию?», ему начинают предлагать безумное количество каких-то технологий, которые надо выучить. При этом всячески преувеличивают трудности.

Сразу скажу, мне в своей жизни пришлось писать программы приблизительно на 20-ти языках программирования. Самый первый – это был Фортран – я его учил год, прежде чем смог написать более-менее нормальную программу. Но с каждым новым языком программирования это время уменьшалось, и уже последние языки я учил так. Брал задачу и сразу начинал программировать на этом языке, изучая его в процессе разработки: «Ага, отлично, вот этот новый язык, понятно. Начинаем писать программу, начинаем разрабатывать алгоритм. А вот это как? А вот это?»



Потому что если уже знаешь несколько языков программирования, то начинаешь понимать общие принципы. Они во всех языках программирования абсолютно одинаковые. Каждый язык должен решить

задачу и просто решает её своими средствами.

С высоты многолетнего опыта бывает забавно читать о каком-то новом языке программирования, про который пишут: «оригинальный, революционный», а потом начинаешь знакомиться с ним и видишь, что все эти «революционные» идеи уже десятки раз применялись в других языках.

Как только вы поняли алгоритм, то остается лишь выразить его языковыми средствами . Все это вы можете пройти совершенно самостоятельно: никаких корочек, никакого официального образования вам не нужно.

Поэтому ничего страшного в изучении программирования нет и можете смело начинать его учить.

Глава 6. Организация занятий

В этой главе поговорим о том, как правильно организовать самообучение программированию. Сначала я перечислю ложные пути, по которым многие люди идут, в том числе и те, которые учатся в официальных учебных заведениях. Это все пути ложные, они очень трудоёмкие, они ни к чему не ведут.

Ложный путь №1. Читать книги и справочники подряд

Самый распространённый и неверный путь изучения программирования – это читать какие-то толстенные справочники, книжки по программированию, в которых описывается вообще всё, что можно. Если зайдёте в книжный магазин и подойдёте к полке с программированием, то вы увидите толстенные книжки, в которых по 500-700 страниц. Я даже видел книжку, в которой было больше тысячи страниц. Эти книги битком набиты информацией, которая вам никогда не понадобится. И люди пытаются это читать и запоминать.

Например, есть многотомный труд Дональда Кнута «Искусство программирования». Эта книга впервые была издана еще в 1968 году. Тома толстенные. Зачем-то на всех форумах рекомендуют эту книжку. Сразу скажу: я эту книжку открыл один раз, полистал и поставил на полку.

Я её читать не стал и не собираюсь, и никогда в жизни мне эта книжка не пригодилась. Почему? Потому что на самом деле, читать такие толстенные справочники – то же самое, что читать Большую Советскую Энциклопедию или всю Википедию. Это бессмысленно. Там вам для реальной работы потребуется, может быть, 1%. Вы пользуетесь энциклопедией в тот момент, когда у вас появляется конкретный вопрос. Вы открываете энциклопедию, ищете соответствующую статью и читаете. Вот точно так же нужно пользоваться этими толстенными справочниками и книжками. То есть вы, как изюм из булки, выковыриваете только те знания, которые вам нужны сейчас для практической работы.

Ложный путь №2. Изучать теорию без практики

Этот ложный путь часто предлагается в ВУЗах. К сожалению, я много раз это наблюдал. Курсы построены так, что студентам даётся колоссальное количество информации, которое никак не подтверждается практикой. А поэтому быстро забывается и практического значения не имеет.

Вы слушаете множество замечательных историй про теорию программирования, но практики нет. Вы учите абстрактные знания, которые могут пригодиться, а могут и не пригодиться.

Если вы решили не идти в ВУЗ, а пойти на курсы программирования, то у вас также есть шанс получить гору совершенно ненужной информации, которая вам потом нигде не пригодится. Поэтому я всё-таки рекомендую самообразование.

Ложный путь №3. Учить программирование, но не программировать

Даже если практика есть, то нужно как можно быстрее переходить к программированию на заказ. Иначе все усилия пропадут зря.

Есть очень большая ошибка – именно в программировании она достаточно фатальная – это когда люди начинают получать информацию о программировании «про запас». То есть вы не собираетесь сейчас писать программы, а вы говорите себе: «Вот, я сейчас изучу этот язык, потом когда-нибудь пригодится». Не пригодится. Вы всё забудете. Машинные инструкции очень быстро вылетают из головы. Это связано с тем, что компьютер функционирует совсем не так, как устроено человеческое мышление. Как только вы какую-то технологию не используете, она

забывается моментально.

И еще есть проблема. Информационные технологии развиваются быстро. Вы можете досконально выучить определенную технологию программирования, отложить работу на пару лет, а потом обнаружить, что эта технология устарела и никому не нужна. Ниже я приведу несколько примеров, как устаревали технологии, и тысячам программистов приходилось переучиваться.

Правильный путь

Намного лучше взять книгу самому и выбрать то, что нужно для работы именно сейчас. Я много раз видел картину, когда новичок вдруг начинал писать очень хорошие программы. На мой вопрос:

- Где ты учился?, - часто отвечали:
- Нигде не учился. Нужна была программа, сел и разобрался.

Меня всё время спрашивают: а можно ли устроиться на работу без диплома? Ответ прост: для других профессий это проблематично, но только не в IT-сфере. Поскольку постоянно существует и не уменьшается огромный дефицит программистов, то всё, что вам нужно – это иметь своё программистское портфолио. Если вы можете продемонстрировать то, что вы сделали, то вас возьмут. Никто и не поинтересуется, есть ли у вас какая-то корочка. Поэтому диплом в сфере IT не имеет значения. Кстати, у Билла Гейтса корочек тоже нет. Через два года обучения Билл Гейтс был отчислен из Гарвардского университета за пропуски занятий.

А в чем же состояло обучение Билла Гейтса?

Вот что пишет он сам: *«Я помешался на компьютерах. Пропускал физкультуру. Сидел в компьютерном классе до ночи. Программировал по*

выходным. Каждую неделю мы проводили там по двадцать-тридцать часов. Был период, когда нам запретили работать, потому что мы с Полом Алленом украли пароли и взломали систему. Я остался без компьютера на целое лето. Тогда мне было пятнадцать-шестнадцать лет…»

То есть выбирайте курс по программированию, садитесь за компьютер и начинайте учиться. Есть хороший момент, связанный с Большим Барьером: если вы ошиблись, у вас что-то не получается, то это абсолютно нормально для ученика, но если вы чувствуете, что вы злитесь, вас всё это бесит, и вы не можете идти дальше, потому что ничего не получается, то, скорее всего, вы Большой Барьер не проходите.

В случае, если вы уже поступили в ВУЗ, вы уже потратили колоссальное количество усилий, а может быть, и денег, вам, конечно, хочется корочку получить. В случае самообразования вы всегда можете или выбрать другую область программирования, либо прекратить заниматься вообще в любой момент.

Для обучения программированию не столь важны ваши усилия, а важна регулярность занятий. То есть, полчаса каждый день в программировании – это намного больше, чем целые сутки раз в неделю. Каждое выпадение из процесса очень сильно замедляет вашу работу.

Меня часто спрашивают: если, допустим, сейчас программирование не идёт, что делать? Я рекомендую: всё равно, возьмите, запустите среду программирования, поизучайте ее, посмотрите библиотеки, попробуйте новые фишки, почитайте технологии, которые рядом. То есть всё равно вы хоть как-то находитесь в атмосфере программирования, и в этом случае вы совершенно спокойно будете двигаться.

И еще одна ошибка – люди ставят себе какие-то совершенно недостижимые цели, когда приходят и заявляют: «А я сделаю второй Microsoft Word», или: «Я напишу свою Windows». Эти программы – разработка колоссального количества программистов, сисадминов, тестировщиков, там были вложены миллионы долларов и годы работы. Повторить это одному человеку совершенно немыслимо.

Ставить подобные цели – это путь к провалу. Нужно ставить практические цели, а именно -- писать простые коммерческие программы, за которые вам заплатят деньги. Вот это реальные цели. И, конечно же, когда вы уже получили первые деньги за программирование, это очень сильно мотивирует, и, соответственно, вы можете продолжать дальше.

Глава 7. Секрет быстрого продвижения

Сейчас я открою вам главный секрет именно тех студентов, которые успешно устраивались на хорошую работу сразу после ВУЗа. Обычно в начале семестра я замечал, что два-три студента из группы имеют навыки программирования. Я вызывал их на кафедру и говорил: «Вам не нужно ходить на мои лекции и посещать практики. Я даю вам конкретное задание. Когда вы его сделаете, я вам поставлю зачет». И раскрывал главный секрет быстрого продвижения.

Им это очень нравилось, они с удовольствием брали задание, работали, получали зачет и потом меня благодарили, потому что их брали на работу уже как опытных программистов.

Сейчас я расскажу этот секрет быстрого продвижения, который я открывал лучшим студентам. И он по своей эффективности намного превосходит любые курсы, учебники, вузовские программы и так далее.

Главный секрет быстрого продвижения в изучении программирования заключается в следующем: вам нужно взять некоторую программу, которую вы сами будете использовать в повседневной жизни, и начать ее разрабатывать. Эта программа должна быть вам действительно полезна. Например, вы любите фотографировать. Сделайте такую программу, в которую можно загрузить фотографии и сразу для нескольких фотографий написать, где она была сделана, например, Сочи или Париж. Программа сразу всем фото в папке проставит нужную надпись.

Конечно же, идей может быть бесконечное множество. Кто-то может создать программу для проигрывания мелодий, кто-то особый будильник.

Единственное требование к ней – вы реально будете ею пользоваться.

Что происходит дальше? Сначала человек быстренько набросал эту программу. Она появляется такая слабенькая, косенькая, но человек уже ею пользуется. И каждый раз, когда он ее запускает, видит ее недостатки. И вот в этот момент появляется мотивация прямо сейчас сесть, запрограммировать и получить удобство для себя. Но как раз в этот момент вы еще не знаете, как это сделать, и вы начинаете рыть, искать какие-то форумы, книжки, учебники, искать информацию, как сделать эту функцию. И дальше, как только вы ее сделали, получаете, что называется автоматическое вознаграждение, то есть вы только что нечто сделали, запрограммировали и вот уже этим пользуетесь.

И это настолько захватывает! Это сильный мотивирующий момент!

А еще часто эту программу человек начинает раздавать друзьям, те говорят, что программа очень удобная и они начинают просить: а ты можешь эту функцию добавить или ту? Этот принцип помогает программисту все время быть мотивированным, все время иметь желание двигаться. И в этом случае вы начинаете быстро входить в процесс программирования.

Очень много программ и сервисов мирового значения были сделаны именно таким способом. Кто-то начинал писать программу для себя, но сделал это настолько хорошо, что ею начинали пользоваться все.

В этом процессе нет такого, что вы себя заставляете. Вам хочется в программе иметь определенную функцию – вы ее делаете. Через какое-то время вы уже начинаете становиться мастером своего дела, потому что вы сами себя побудили идти вперед. Студенты, которым я подсказывал этот секрет быстрого продвижения, достигали больших успехов.

Например, один студент через полгода зашел ко мне на кафедру, показал ноутбук и сказал:
- Этот ноутбук я купил на деньги, которые заработал программированием.

Глава 8. Сколько времени это займет?

Сколько времени займет обучение программированию? Это важный вопрос.

Дойти до первой коммерческой программы за ОДИН год с того момента, как вы начали изучать программирование, вполне реально.

Возможно, вы удивились, но это так. Однако при том, что вы занимаетесь каждый день и используете секрет быстрого продвижения. Потому что если вы просто посещаете лекции по программированию, то и за пять лет вы можете ничего не добиться. Я это наблюдал много раз.

Сразу скажу: самая большая трудность - в самом начале, когда вы не знаете ничего, когда у вас ничего не получается. Здесь потребуется волевое усилие. Не надо строить иллюзий. В самом начале программирование – это действительно сложное дело. Никаких хитрых приемов здесь не существует. В самом начале будет тяжело.

Но вот когда у вас дело пошло, когда начинает получаться, и вы втянулись, дальше дело идет гораздо веселее, дальше это у вас становится приключением: а вот это попробовать, а вот это, это я умею, это могу. Соответственно, через какое-то время вы будете по-настоящему готовы к профессиональной работе. Чуть позже мы поговорим, как правильно приступать к профессиональному программированию.

Обычный курс обучения программированию для студентов-первокурсников занимает год, то есть два семестра. По многолетнему опыту могу сказать, что если студент в течение года действительно занимался, делал практические задания, то через год он может писать программы. И часто такие студенты начинают искать подработку программистами. Я всячески поддерживал их в этом, давал контакты фирм, которым требуются программы на заказ.

И наоборот, когда я работал в ВУЗе, то часто ко мне обращались руководители отделов программистов, которые просили присылать им лучших студентов. Мы устраивали встречи работодателей и студентов.

Спрос на программистов, пусть и начинающих, огромный. Поэтому даже студенты, которые могут только подрабатывать в свободное от учебы время, все равно очень ценились.

Часто ко мне приходили лучшие студенты и просили помочь в выборе, потому что они получали приглашения сразу в несколько компаний.

Поэтому путь от новичка до профессионального программиста реально пройти за один год. Я этому видел много примеров. Но только при условии, что этот человек сумел преодолеть Большой Барьер. Таких людей было немного.

Глава 9. Увлечение технологиями

В этой главе я расскажу про ловушки, в которые часто попадают новички. Иногда в подобные ловушки попадают даже профессиональные программисты. Эти ловушки заманят вас, высосут много сил, но никуда вас не продвинут.

Как эти ловушки выглядят со стороны? Начинаются они с того, что появляется какая-то «новая, уникальная, революционная» технология, которая якобы отменит все, что было до этого.

После этого на форумах программистов начинаются споры о том, какая технология самая лучшая. В результате все начинают срочно изучать новую технологию, тратят на это время, а через несколько лет выясняется, что у новой технологии много недостатков.

После этого появляется еще более «новая, уникальная, революционная», и весь процесс повторяется.

Давайте рассмотрим несколько историй из мира программирования, потому что, к сожалению, этот мир устроен так, что все время приходят новые технологии. Каждые два-три года объявляют о том, что программирование будет совершенно другое, и нужно всем переучиваться. Все переучиваются, эта волна уходит, наступает следующая и так далее.

Приведу несколько выдающихся примеров, которые стоили колоссальных усилий тысячам программистов. Когда-то считалось, что самым важным и главным языком программирования будет Кобол. Этот язык был просто культовым. Статьи того времени говорили о том, что «Кобол – язык будущего», изучайте только его, кроме него ничего не будет.

Мне лично с Коболом повезло. Меня предупредили, что Кобол имеет много недостатков, поэтому я его не изучал. Через какое-то время он просто исчез, и сейчас это слово «Кобол» известно только историкам программирования.

Через несколько лет стал популярным язык Паскаль. Все обсуждения программирования крутились только вокруг него. Книжки, учебники, курсы – это все было о Паскале. В какой-то момент казалось, что скоро все будут писать на Паскале. Этот момент я застал. Практически все мои знакомые программисты стали срочно изучать Паскаль.

Затем было объявлено, что надо изучать не Паскаль, а Object Pascal, который намного лучше.

Вершиной развития Паскаля было создание интегрированной среды разработки для Object Pascal, которая получила название Delphi. Подразумевалось, что эта платформа должна была интегрировать все варианты программирования. Она должна была разрабатываться и под Windows, и под macOS, и под iOS, и под Android, и так далее. Все начиналось замечательно, но с каждой версией Delphi она становилась все более монстроподобной. В какой-то момент программисты стали испытывать все больше трудностей. В марте 2006 года компания Borland приняла решение о прекращении дальнейшего развития Delphi. Поддержка среды прекратилась, и Delphi просто исчезла из мира программирования. Десятки тысяч программистов потратили колоссальное количество времени и сил на изучение этой среды, но все было потрачено впустую. Многие мои знакомые фактически просто остались без работы, потому что их знания оказались ненужными.

Следующая «супер-новейшая и самая мощная» технология возникла в 2002 году. Компания Microsoft вдруг объявила, что она выпускает так называемый Microsoft .NET Framework, который может одинаково применяться как для создания обычной программки, так и для веб-приложения и будет работать вообще везде. Дальше началась шумная реклама технологии .NET, в которой постоянно говорилось, что эта технология совместима со всем, чем только возможно. Более того, в состав технологии .NET вошел язык C#, который был создан по принципам языка Object Pascal.

Скоро выяснилось, что библиотеки .NET не только несовместимы между собой, но они также несовместимы с разными версиями Windows.

Начались проблемы у разработчиков. Уже готовая программа переставала работать, когда на компьютере ставилась другая версия Windows. Конечно же, в коммерческом программировании это недопустимо, поэтому о технологии .NET как о средстве для создания Интернет-приложений уже

давным-давно никто не вспоминает. Эта технология применяется только в узкой нише.

Ну и последняя ситуация, которая разворачивается прямо сейчас.

Есть детище компании Sun Microsystems (сейчас эта компания является частью Oracle) – платформа Java. На заре своего появления эта платформа была заявлена как полностью универсальная. Лозунг Java: «Написано однажды, работает везде». В какой-то момент стали говорить о том, что Java - это «мега-технология». Она будет работать везде и вытеснит всех.

В отличие от .NET платформе Java действительно удалось быть популярной на самых разных устройствах. Но потом выяснилось, что если приложение на Java сравнить с приложением на языке Си, то приложение на Java будет работать в 7 раз медленнее и требовать в 30 раз больше памяти.

То есть программы на Java – прожорливые и медленные, которые кроме того имеют большие проблемы с безопасностью. Вплоть до того, что Apple в свое время несколько раз блокировал работу Java OS X как раз из-за проблем безопасности. Даже появилось такое выражение на платформе Apple: «Нет Java – нет проблем». И в данное время платформа Java заняла свою нишу – это разработка тех приложений, для которых не имеет большого значения скорость, потребление памяти и безопасность.

Глава 10. Что изучать в первую очередь?

Программирование – это широкая сфера. Существуют такие разделы:

- системное программирование,
- программирование игр,
- корпоративное программирование,
- программирование мобильных приложений,
- программирование микропроцессоров и т.д.

Можете выбирать всё, что угодно.

Но в 2020 году после пандемии коронавируса резко пошел вверх интерес к веб-разработке, так как веб-разработкой можно заниматься удаленно.

Веб-разработка – это программирование сайтов и интернет-сервисов. В этой теме популярны следующие технологии:

- Язык разметки HTML
- Язык описания стилей CSS
- Язык программирования JavaScript
- Язык программирования PHP
- Система управления базами данных MySQL

Поэтому сейчас я рекомендовал бы начинать с изучения веб-разработки. В ближайшие годы спрос на специалистов в этой области будет только расти.

Я регулярно провожу бесплатный вебинар: «**Как с нуля освоить профессию веб-разработчика**». Записывайтесь на него здесь: **https://sheremetev.aoserver.ru/?r=ac&id=9083&lg=ru**