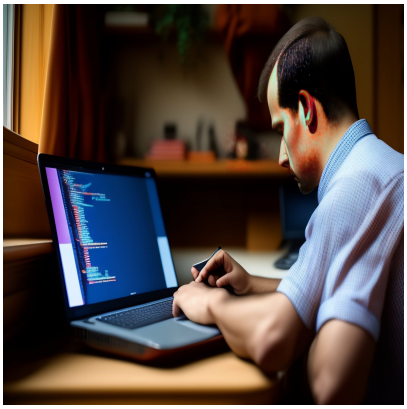


Как начать писать код: Полное руководство для начинающих программистов

Описание

Кодирование – один из самых ценных навыков, которые вы можете приобрести. Если вы ищете “как научиться кодировать”, возможно, вы хотите продвинуться по карьерной лестнице. Возможно, вы надеетесь создавать программное обеспечение или игры для своих друзей или лучше ориентироваться в технологическом пространстве. В любом случае, изучение компьютерного языка может стать важным подспорьем в вашем профессиональном и личном развитии.

При наличии времени, целеустремленности и доступа в Интернет любой человек может научиться писать код. Если вы читаете эту статью, то это касается и вас. Поэтому в этом руководстве мы расскажем обо всем, что вам нужно знать, чтобы начать писать код, в том числе:

- Зачем учиться кодить?
- Как начать писать код?
- Кодирование в сравнении с программированием
- Кодирование для начинающих

Что такое кодирование?

Кодирование – это процесс использования языков программирования для передачи инструкций компьютеру. Эти инструкции приводят в действие веб-сайты, программное обеспечение и приложения, которыми люди

пользуются каждый день.

Зачем учиться кодировать?

Прежде чем приступить к первому уроку, подумайте, почему вы хотите научиться кодировать. Это поможет вам определить, какой язык программирования вы решите изучать в первую очередь, какие проекты вы хотите выполнять и, в конечном счете, что вы хотите сделать из своих навыков.

Вот некоторые известные преимущества:

Кодирование помогает развивать профессиональные навыки.

Давайте уберем с дороги очевидное. Знание компьютерного программирования является ценным преимуществом при трудоустройстве. Поскольку технологии продолжают вплетаться в нашу повседневную жизнь, навыки кодирования будут становиться все более востребованными среди кандидатов – по данным сайта Indeed.com, несколько наиболее востребованных навыков относятся к области вычислительной техники. Если вы хотите сделать карьерный поворот в сторону технологий или перейти на более техническую должность в своей области, знание хотя бы одного соответствующего языка программирования является обязательным условием.

Однако это относится не только к разработчикам. Веб-дизайнеры должны знать HTML, CSS и JavaScript. Менеджеры проектов должны знать внутреннюю работу инструментов, которые они помогают создавать. Даже если у вас простой сайт на WordPress, знакомство с языками front-end и некоторыми PHP поможет вам в этом. Даже если вы не претендуете на сугубо техническую должность, опыт кодирования является преимуществом. Он демонстрирует технические знания, способность понимать абстрактные концепции и то, что вы можете решать сложные проблемы. Наконец, знание кодирования позволит вам заняться фрилансом или сделать карьеру на полный рабочий день.

Кодирование может помочь вам зарабатывать больше.

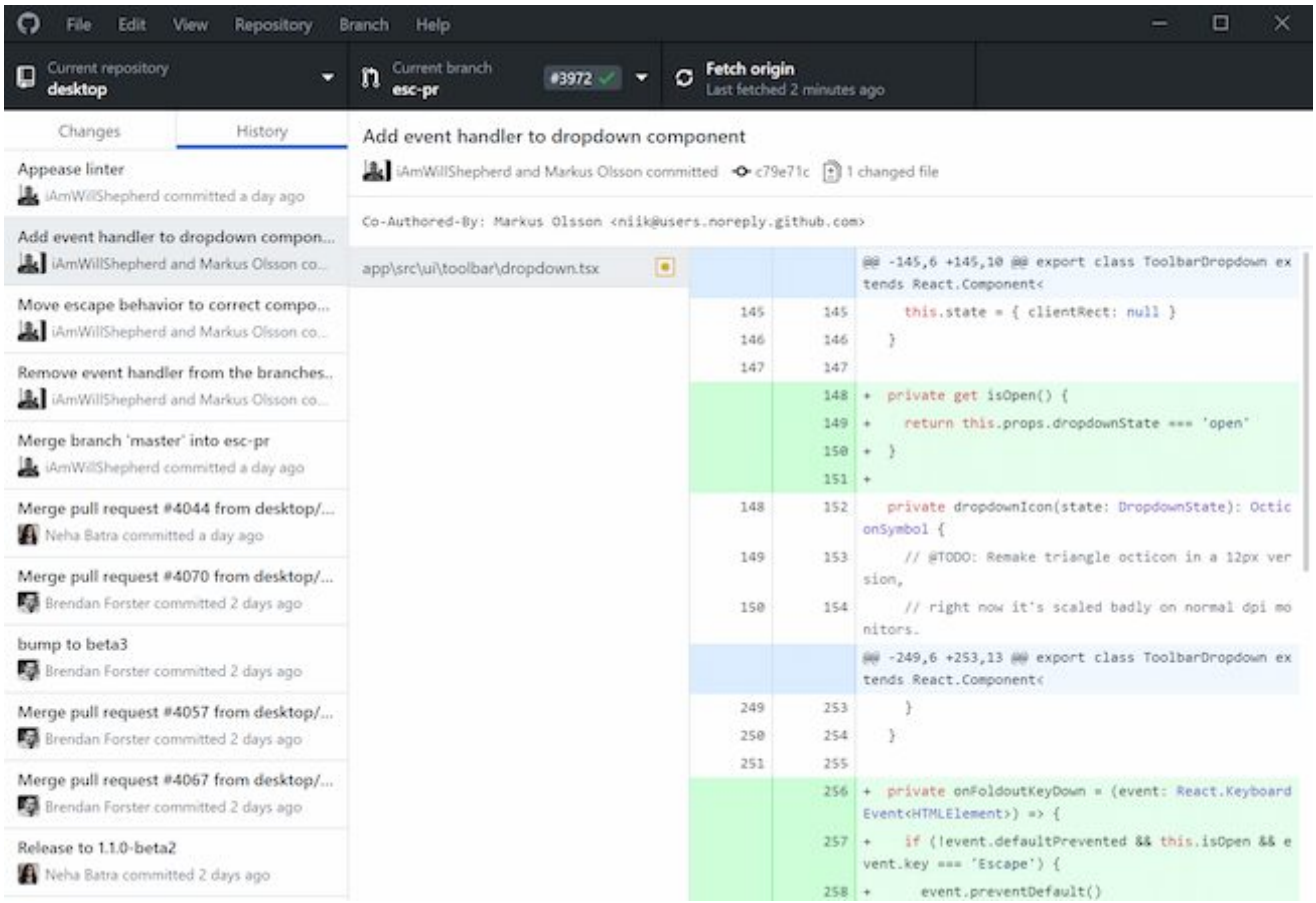
Средняя зарплата начального уровня в США в 2022 году составит 40 153 доллара. Но средняя стартовая зарплата программиста более чем в два раза выше – $85 293. Ранее мы упоминали, что между кодерами и программистами есть различия.

Продолжая изучать кодирование, вы можете начать специализироваться. Многие из самых высоких зарплат в области кодирования связаны с возможностью предложить свои навыки в определенном виде кодирования.

Например, продолжая изучать кодирование, вы можете решить стать разработчиком. Помимо написания кода, разработчики также занимаются отладкой программного обеспечения и работают с исходным кодом. Разработчики обычно специализируются на определенном языке программирования. Зарплата разработчиков часто выше, чем у программистов, и прогнозируется высокий рост числа рабочих мест. По данным Бюро статистики труда США, к 2030 году число рабочих мест для разработчиков должно вырасти на 22%.

Кодирование позволяет создавать вещи.

Одно из самых приятных преимуществ изучения кода – это возможность воплощать свои идеи в жизнь. У вас есть концепция веб-сайта, приложения или компьютерной игры? Теперь вы можете создать его именно так, как вам хочется, а затем поделиться им со всем миром. Хотите ли вы монетизировать свой проект, разместить его на платформе с открытым исходным кодом, такой как GitHub, или просто создавать что-то в качестве хобби, у вас будут знания и инструменты для этого.



Приятно осознавать, что вы можете создавать программы, которые до сих пор не были вам до конца понятны. Кроме того, проекты играют важную роль в процессе обучения и поиска работы.

Кодирование может помочь вам лучше понять окружающий мир.

Изучение даже основ компьютерного программирования поможет вам понять компоненты растущего технологического ландшафта. Вы сможете совершенно по-новому взглянуть на технологии в вашей жизни и понять, как все это работает вместе.

Кодирование – это весело.

Это банально, но верно – для многих изучение кода является полезным и приятным опытом. После того, как вы освоите основы и начнете работу над собственными оригинальными проектами, этот процесс будет меньше походить на обучение и больше – на отдых. В конце концов, если вы не получаете от этого удовольствия, зачем делать карьеру?

Как начать кодировать?

1. Определите, почему вы хотите научиться кодировать.
2. Выберите, какой язык кодирования вы хотите изучить в первую очередь.
3. Пройдите онлайн-курсы.
4. Смотрите видеоуроки.
5. Читайте книги и электронные книги.
6. Используйте инструменты, которые облегчают изучение кода.
7. Посмотрите, как кодируют другие люди.
8. Выполняйте проекты по кодированию.
9. Найдите наставника и сообщество.
10. Подумайте о том, чтобы записаться в буткемп по кодингу.

Благодаря Интернету, никогда еще не было лучшего времени для изучения кода. Но огромное количество вариантов может поставить некоторых начинающих кодеров в тупик еще до того, как они начнут. В этом разделе мы расскажем о том, как научиться кодированию для начинающих, и порекомендуем некоторые ресурсы для каждого этапа.

1. Определите, почему вы хотите научиться кодировать.

Очень заманчиво сразу же приступить к кодированию. Но если у вас нет конечной цели, вы можете разочароваться и прекратить обучение, не дойдя до самого интересного. Поэтому, прежде чем приступить к обучению, подумайте, почему вы хотите научиться кодировать. Подумайте о проектах, которые вы хотите завершить, о том, почему этот навык вас увлекает, и о том, какими ресурсами вы располагаете.

Например, предположим, вы хотите стать разработчиком, чтобы получать больший доход для своей семьи. Знаете ли вы, где вы хотите работать и для каких проектов им нужны разработчики? Готовы ли вы потратить время на то, чтобы научиться кодировать, изучить нужные языки программирования и создавать проекты, которые покажут, что вы способны на это? Постановка такой широкой цели может разочаровать. Вместо этого начните с более мелких и конкретных целей. Например, скажите, что вы хотите создать мобильное приложение для своего друга, который готовится к полумарафону через год.

Эта цель поможет:

- Поможет вам приобрести необходимые навыки.
- Даст вам представление о том, с какого языка программирования лучше начать.
- Определит крайний срок, чтобы вы могли управлять своим временем во время обучения.

2. Выберите, какой язык программирования вы хотите изучить в первую очередь.

Если вы пытаетесь решить, с какого языка программирования начать, подумайте о своих долгосрочных целях. Вы занимаетесь кодированием для развлечения или для продвижения на работе? У вас гибкие временные рамки или вам нужно закончить проект в спешке? Как начинающий, вы, возможно, захотите начать с языка, который не использует структуры данных или алгоритмы. Если это так, то HTML или CSS – отличные варианты для начала. Но такие языки, как Java и Python, также отлично подходят для начинающих, и они также имеют широкий спектр применения. Иногда на изучение языка программирования могут уйти месяцы,

поэтому уделите время принятию решения, чтобы убедиться, что ваше время потрачено не зря.

3. Пройдите онлайн-курсы.

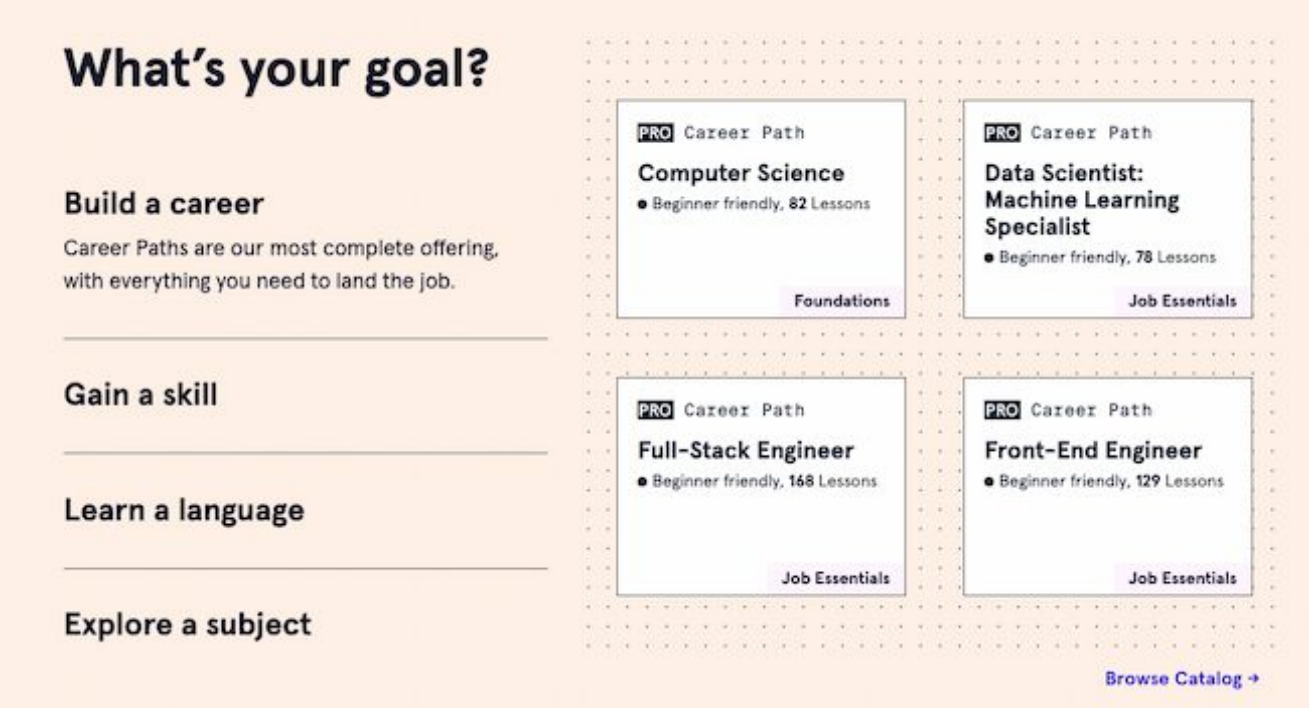
Для того чтобы научиться программировать, больше не обязательно учиться в классе. Сегодня существуют тысячи онлайн-курсов по программированию, которые охватывают все – от основ HTML до структур данных и сложных алгоритмов. Первый курс должен знакомить вас с основами языка и содержать интерактивные модули и задания, которые будут направлять ваше обучение. Курсы создают структуру обучения, что очень важно, поскольку понятия в информатике строятся друг на друге. Организованный курс позволяет все усвоить и гарантирует, что вы будете изучать предметы в правильном порядке.

Популярные поставщики бесплатных курсов включают:

- freeCodeCamp
- W3Schools
- Известный гарвардский курс “Введение в компьютерные науки” на edX и канал CS50 на YouTube.

Эти варианты отлично подходят для того, чтобы на раннем этапе определить, готовы ли вы потратить время на изучение конкретного языка. Кроме того, существует множество платных курсов, стоимость которых в разы меньше, чем

стоимость очных занятий.



Эти варианты охватывают широкий спектр тем по CS для начинающих, среднего и продвинутого уровня:

- Code Academy.
- Springboard.
- Введение в информатику и программирование с использованием Python в Массачусетском технологическом институте.

Некоторые платные сервисы предлагают бесплатные курсы или пробные версии, если вы хотите получить представление об их преподавании, прежде чем брать на себя обязательства.

4. Посмотрите видеоуроки.

Вы наверняка смотрели пару уроков на YouTube – почему бы не сделать то же самое с кодированием?

В то время как онлайн-курсы – это лучший вариант для получения практического опыта, онлайн-видео могут дополнить ваше обучение и удовлетворить ваше любопытство. Мои личные фавориты – это:

- Краткий курс информатики

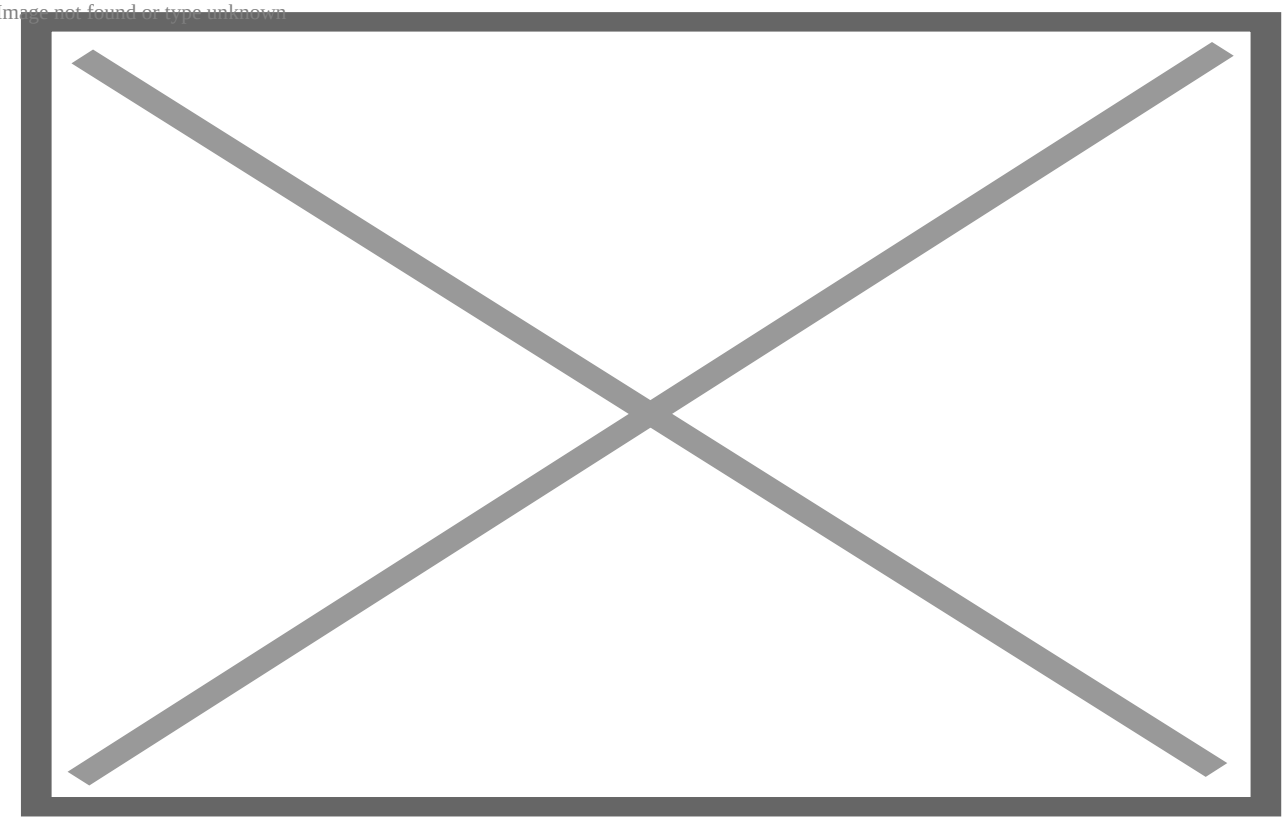
- “Основы” Тома Скотта.

Оба они охватывают более широкие темы в области вычислительной техники.

5. Читайте книги и электронные книги.

Предпочитаете старый добрый подход? Возьмите книгу по выбранному вами языку для начинающих. Книги познакомят вас с фундаментальными концепциями и помогут вам в написании кода.

Image not found or type unknown



Ниже приведены некоторые авторитетные тексты по каждому рекомендуемому языку для начинающих:

- HTML/CSS: Руководство для начинающих по HTML и CSS для маркетологов
- JavaScript: Eloquent JavaScript: Современное введение в программирование
- Python: Краткий курс Python
- C: Абсолютное руководство для начинающих программистов на языке C
- C++: C++ Primer
- C#: C# 8.0 и .NET Core 3.0 – современная кроссплатформенная разработка: Создавайте приложения с помощью C#
- Java: Эффективная Java

- PHP: Первый курс PHP и MySQL
- Ruby: Хорошо подготовленный рубист
- SQL: SQL за 10 минут, Sams Teach Yourself
- Swift: Программирование на Swift: Руководство по большому ботаническому ранчо

6. Используйте инструменты, облегчающие изучение кода.

Хотя приятно осознавать, что для написания кода не нужны никакие специальные инструменты, есть инструменты, которые могут помочь.

Редактор кода

Текстовые редакторы включают в себя функции, облегчающие написание кода, такие как цветовое кодирование, автозаполнение, поиск и замена и темный режим.

Большинство профессионалов используют редактор кода. Этот инструмент поможет вам быстрее написать чистый код. Редакторы кода похожи на текстовые редакторы, но у них есть дополнительные функции, которые помогают вам управлять и редактировать код, например:

- Подсветка синтаксиса
- Поиск кода
- Встроенная среда терминала

Notepad++, Sublime Text, VScode и Emacs – популярные редакторы для начинающих.

Консоль

Вы также захотите узнать свой компьютер получше. Когда вы начнете кодировать, вы должны научиться работать с консолью. Это текстовый интерфейс вашей операционной системы. Консоль позволяет находить файлы и выполнять команды на них быстрее, чем стандартный графический интерфейс пользователя (GUI). Ознакомьтесь с тем, как ее использовать, включая основные команды UNIX, поскольку вы будете делать то, что невозможно в графическом интерфейсе.

Управление проектами

Многие проекты начинаются просто, но по мере реализации становятся все

сложнее. Используя инструмент управления проектами, вы можете обмениваться файлами, обновлять графики и выделять отдельные части проектов кодирования. Это облегчает отслеживание побочных проектов и дополнительных файлов при соблюдении графика.

Устранение неполадок

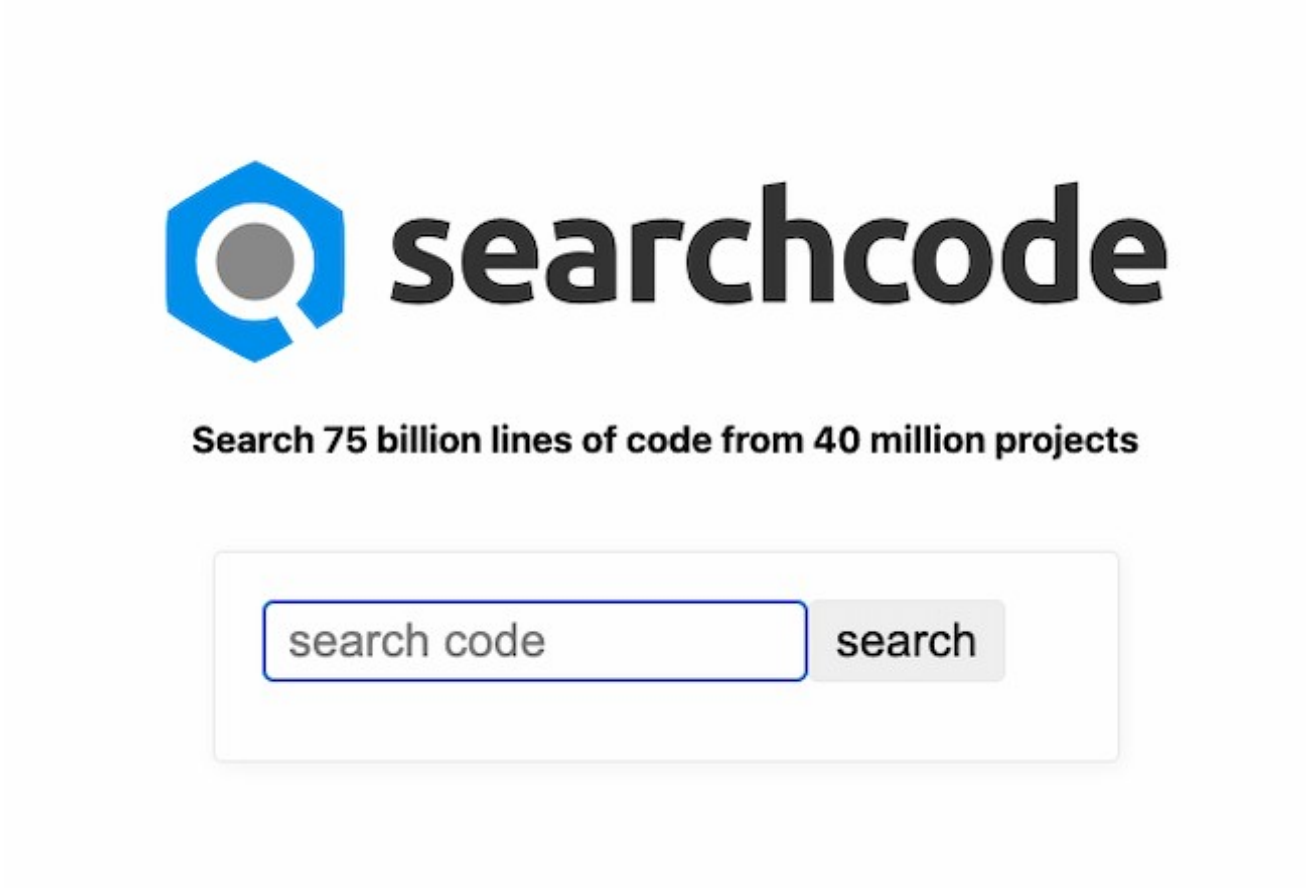
Работая над своими проектами по кодированию, вы можете столкнуться с ошибками, препятствиями и другими проблемами. Это может быть очень неприятно. Отладка резиновой утки может показаться глупой, но она может помочь вам упростить проблему и найти полезные решения.

7. Посмотрите, как другие люди пишут код.

Некоторым людям проще редактировать чужую работу, чем создавать с чистого листа. Если вы относитесь к таким людям, то отличным способом научиться кодировать будет изучение чужого кода.

Изучение кода других программистов также может вдохновить вас и продвинуть ваши навыки кодирования еще дальше. Такой подход поможет вам:

- Лучше понять свой собственный код
- Быстрее понять новый код
- Получить наглядные примеры качественного кодирования



Начните с программы или файла кода, который вам нравится. Если вы не уверены, с какого кода начать, GitHub и searchcode – отличные отправные точки. Старайтесь начинать с рецензируемого кода или проектов с открытым исходным кодом, если это возможно. Если вы знаете, что делает код, выберите один небольшой раздел, а затем работайте в обратном направлении. Это поможет вам понять функцию каждой строки кода. Также полезно прочитать документацию. Это покажет вам идеи, лежащие в основе кода, который вы просматриваете.

8. Выполняйте проекты по кодированию.

Вы учитесь программированию на практике – без этого никак не обойтись. Вы можете прочитать все концепции и синтаксис, необходимые для написания функционального кода. Но пока вы не примените полученные знания на практике, идеи не смогут полностью реализоваться в вашем сознании. Вот тут-то и приходят на помощь проекты. Проект – это любая программа (или веб-сайт), созданная на выбранном вами языке. В начале работы проекты должны быть краткосрочными.

Если вы изучаете курс, вы можете получить проекты, предназначенные для

закрепления концепции. Существует также множество проектов для начинающих программистов, которые вы можете попробовать выполнить самостоятельно. Некоторые классические проекты включают:

- **Конвертер времени,** в котором пользователь задает количество секунд, а ваша программа выдает эквивалент в часах, минутах, днях и т.д.
- **Генератор случайных чисел,** который производит случайное число между двумя значениями, указанными пользователем.
- **Калькулятор,** в котором пользователь указывает свои входные данные и математическую операцию, а ваша программа выдает результат.
- **Адресная книга,** в которой пользователи могут вводить имена контактов, а затем искать их в вашей программе.
- **Алфавитный указатель,** в котором пользователь предлагает список слов, а ваша программа сортирует их в алфавитном порядке.
- **Игра “Палач”,** в которой пользователь пытается угадать скрытое слово, вводя буквы, а ваша программа дает обратную связь для правильных или неправильных догадок. Если пользователь угадывает все буквы вашего слова, он выигрывает.

Быстрый поиск в Google позволит найти еще больше мини-задач, которые потребуют от вас применения ваших навыков для решения реальных проблем.

Преимущества проектов по кодированию

Помимо отработки концепций, проекты предлагают еще два преимущества для вашего обучения. Во-первых, они будут поддерживать вашу мотивацию. Проекты помогают укрепить “почему”, стоящее за кодированием, и устанавливают четкие, осязаемые ориентиры для вашего прогресса. Каждый завершенный проект означает еще один навык за плечами. Когда я учился, это очень воодушевляло. Во-вторых, проекты по кодингу, особенно долгосрочные, дают вам повод показать свою работу. Одно дело – указать в резюме “Python”, другое – показать, что вы создали целый сайт или приложение с нуля. Проекты – обязательное условие для программистов начального уровня, так как они доказывают компетентность в данном языке.

Как оставаться мотивированным во время работы над кодом

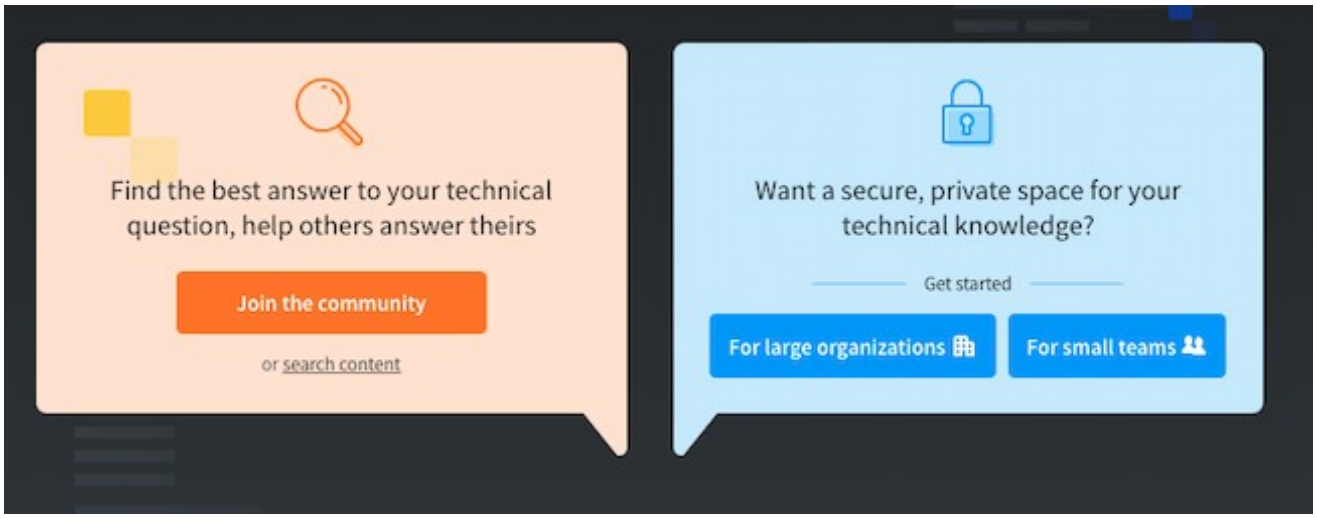
Приступая к долгосрочным проектам, подумайте о том, на что вы готовы потратить

время. Будь то личный веб-сайт, мобильное приложение или настольный инструмент, вы будете преодолевать препятствия на своем пути. Выбрав проект, который вам действительно нравится и о котором вы заботитесь, вы сможете довести его до конца. Еще один отличный способ продолжать заниматься проектами по кодированию – это фриланс. Вам не нужно быть опытным программистом, чтобы создать полезный инструмент для кого-то. Обратитесь к другу, члену семьи или местному бизнесу, нуждающемуся в инструменте или веб-сайте, – это беспроигрышный вариант.

9. Найдите наставника и сообщество.

Все ресурсы, которые я перечислил до сих пор, являются ценными, но в основном они работают в одиночку. Наличие друга или онлайн-сообщества, которые могут дать дальнейшие рекомендации, может оказать неоценимую помощь в обучении. Во-первых, я рекомендую найти наставника. По мере продвижения вы, вероятно, столкнетесь с проблемами, которые, как бы вы ни старались, вы просто не сможете решить. Вот здесь-то и могут помочь наставники.

Наставник не обязательно должен быть настоящим учителем – им может быть любой человек, знающий ваш язык. Они должны уметь объяснять сложные понятия и указывать вам на решения. Опытный наставник может помочь вам следовать практике кодирования, не описанной в учебниках, и дать советы по выбору карьеры в сфере технологий. Возможно, вы также захотите присоединиться к сообществу. Ищите местные группы, сетевые мероприятия и встречи в вашем регионе, а также хакатоны, где вы сможете лично пообщаться с другими программистами. Онлайн-сообщества разработчиков также являются богатым ресурсом для начинающих.



Обязательно загляните:

- Stack Overflow, форумный сайт для вопросов и обсуждения программирования.
- GitHub, хранилище кода для проектов с открытым исходным кодом с активным сообществом разработчиков.
- Women Who Code – некоммерческая организация, которая организует мероприятия, сообщества и размещает вакансии для женщин, делающих карьеру в сфере технологий.
- r/learnprogramming, сабреддит (микросайт на Reddit.com) для начинающих кодеров.

10. Подумайте о том, чтобы записаться в учебный лагерь по кодированию.

Буткемп по кодированию – это краткосрочная программа обучения, которая включает в себя полный учебный план по кодированию в течение нескольких месяцев. Эти программы проходят в быстром темпе, с погружением и являются стартовой площадкой для карьеры разработчика. Буткемпы по кодингу интенсивны и дороги – это не то, во что стоит погружаться без опыта кодинга.

Эти программы в основном предназначены для новичков, которые намерены строить карьеру в области разработки и готовы потратить время, энергию и деньги, чтобы быстро получить необходимые навыки. Хотя выпускники, как правило, находят работу в технологической отрасли, поймите, что это не гарантированный результат. Вам придется отложить приличный кусок своего года и сбережений на такое стремление. Тем не менее, трудно превзойти условия очного обучения в окружении таких же целеустремленных сверстников и преподавателей, как и вы.

Кодирование против программирования

Термины “кодирование” и “программирование” часто используются как взаимозаменяемые, но они не всегда означают одно и то же. И кодирование, и программирование означают написание инструкций для компьютера. Но программирование может также включать алгоритмы и структуры данных. Как правило, термин “программирование” описывает более сложные проекты. Если для написания кода достаточно компьютера и некоторого времени, то для программирования могут потребоваться специализированные программные инструменты. Проекты по программированию обычно крупнее и сложнее. Они могут потребовать управления проектами и более солидной базы знаний.

Языки программирования

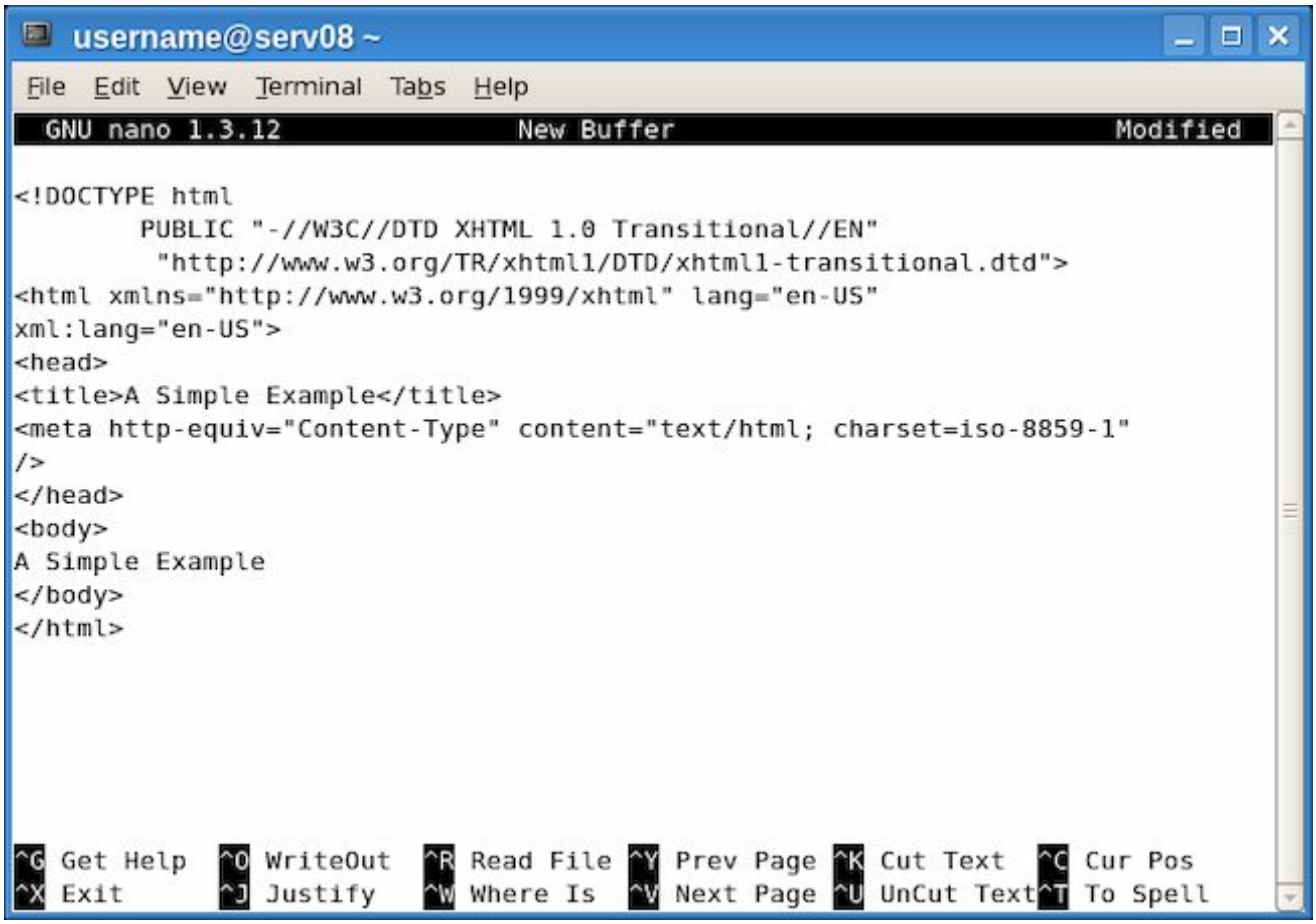
Кодирование требует знания хотя бы одного языка кодирования – набора синтаксиса и правил, понятных компьютерам. Существуют сотни языков кодирования, каждый из которых уникален по своему назначению и возможностям. Но некоторые языки легче выучить, чем другие – с них лучше всего начинать, так как это самый быстрый способ изучить основы программирования. Языки программирования дают вам структуру для инструкций, которые вы пишете. Этот язык похож на английский, но не совсем. Программисты называют термины и грамматику языка программирования синтаксисом. Языки низкого уровня, такие как ассемблер или машинный, легче читаются машинами, чем людьми, поэтому их может быть трудно выучить. Тем не менее, языки среднего уровня, такие как C++, полезно знать, если вы хотите писать:

- Операционные системы.
- Системы баз данных.
- Программное обеспечение для обработки изображений или видео.

Языки высокого уровня обычно легче для начинающих. Некоторые из них больше внимания уделяют структуре, в то время как другие более интерактивны и могут выполнять более сложные функции. Ниже приведены некоторые из лучших языков для начинающих программистов. Постарайтесь освоить только один язык, который соответствует вашим целям, а затем изучите другие, если захотите. Не бойтесь выбрать неправильный язык. Эти языки имеют общие концепции, поэтому вы

можете начать с одного и перейти на другой, если это необходимо.

HTML



Язык гипертекстовой разметки, или HTML, является основой Интернета – он используется для задания содержания веб-страниц. Когда вы загружаете веб-страницу, обычно вы видите HTML-документ, отображаемый вашим браузером. Если вы не уверены, что кодирование – это ваш конек, HTML – самый простой язык для изучения. Это связано с тем, что HTML технически не является языком программирования – он не выполняет скрипты, и на нем нельзя создавать функциональные программы. Тем не менее, HTML присутствует в Интернете повсюду, поэтому если вы хотите понять, что такое Интернет, вам сначала нужно понять HTML.

CSS

```sass
/** SASS code */
.post {
    margin: 2em;

    .title {
        font-size: 2em;
        font-weight: normal;
    }
}

/** Processed result */
.post {
    margin: 2em
}
.post .title {
    font-size: 2em;
    font-weight: normal;
}
```

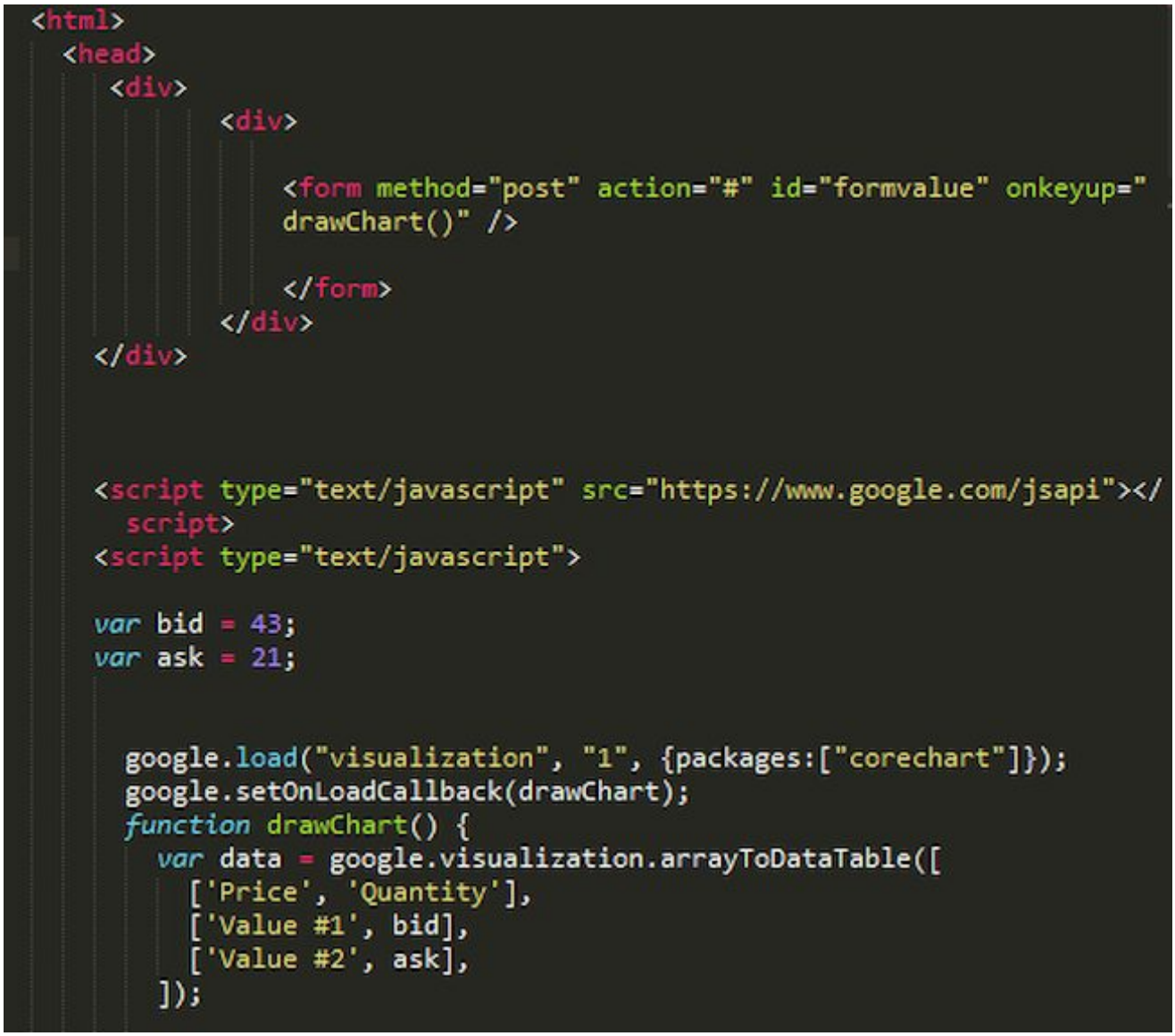
Возможно, вы меньше знакомы с родным языком HTML – CSS. HTML определяет, какое содержимое появляется на веб-странице, но не влияет на то, как это содержимое выглядит. Именно здесь на помощь приходят каскадные таблицы стилей (Cascading Style Sheets, или CSS). Язык CSS обрабатывает стиль HTML – он задает такие характеристики, как цвета, размер, шрифты и даже макеты всей страницы. CSS также не является языком программирования. Это набор правил, применяемых к HTML. HTML и CSS почти всегда используются вместе, поэтому я рекомендую изучить оба. В противном случае ваши веб-страницы будут выглядеть довольно просто.

Если вы не знаете, как научиться кодированию, начните с HTML и CSS.

HTML и CSS легко изучать в основном потому, что они не требуют от вас вычислительной логики языков программирования. Изучение HTML и CSS также может показаться менее абстрактным, чем других языков, поскольку вы быстро видите результаты своего кода – просто создайте файл .html и откройте его в браузере. Или откройте существующий веб-сайт и используйте инструмент проверки, чтобы заглянуть в основной код. Это позволяет изучить два языка, которыми вы пользуетесь каждый день, за один день. Но если вы хотите, чтобы

ваши веб-страницы могли что-то делать, вам понадобится...

JavaScript



JavaScript – это язык программирования, который превращает статические веб-страницы в динамические. Он позволяет элементам страницы двигаться, реагировать на действия пользователя, такие как щелчки, и выполнять любые операции, выходящие за рамки простого существования на странице. Если вы интересуетесь веб-разработкой и уже знакомы с HTML и CSS, JavaScript – это следующий шаг. Вместе эти три языка составляют большую часть веб-контента, который вы видите. Кроме того, код на JavaScript можно легко протестировать в браузере.

Python

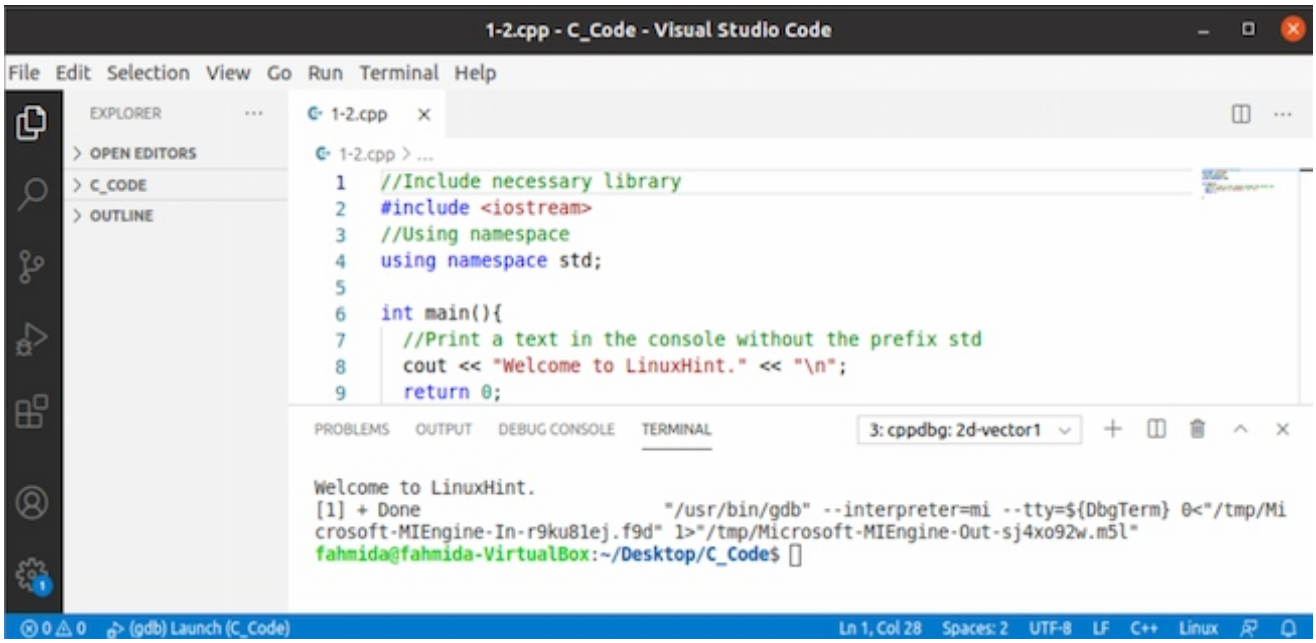
```
In [7]: #iterate through a list
        colors = ["red","green","yellow","black"]
        for x in colors:
            print(x)

        #iterate through a string
        s = "red"
        for x in s:
            print(x)

        red
        green
        yellow
        black
        r
        e
        d
```

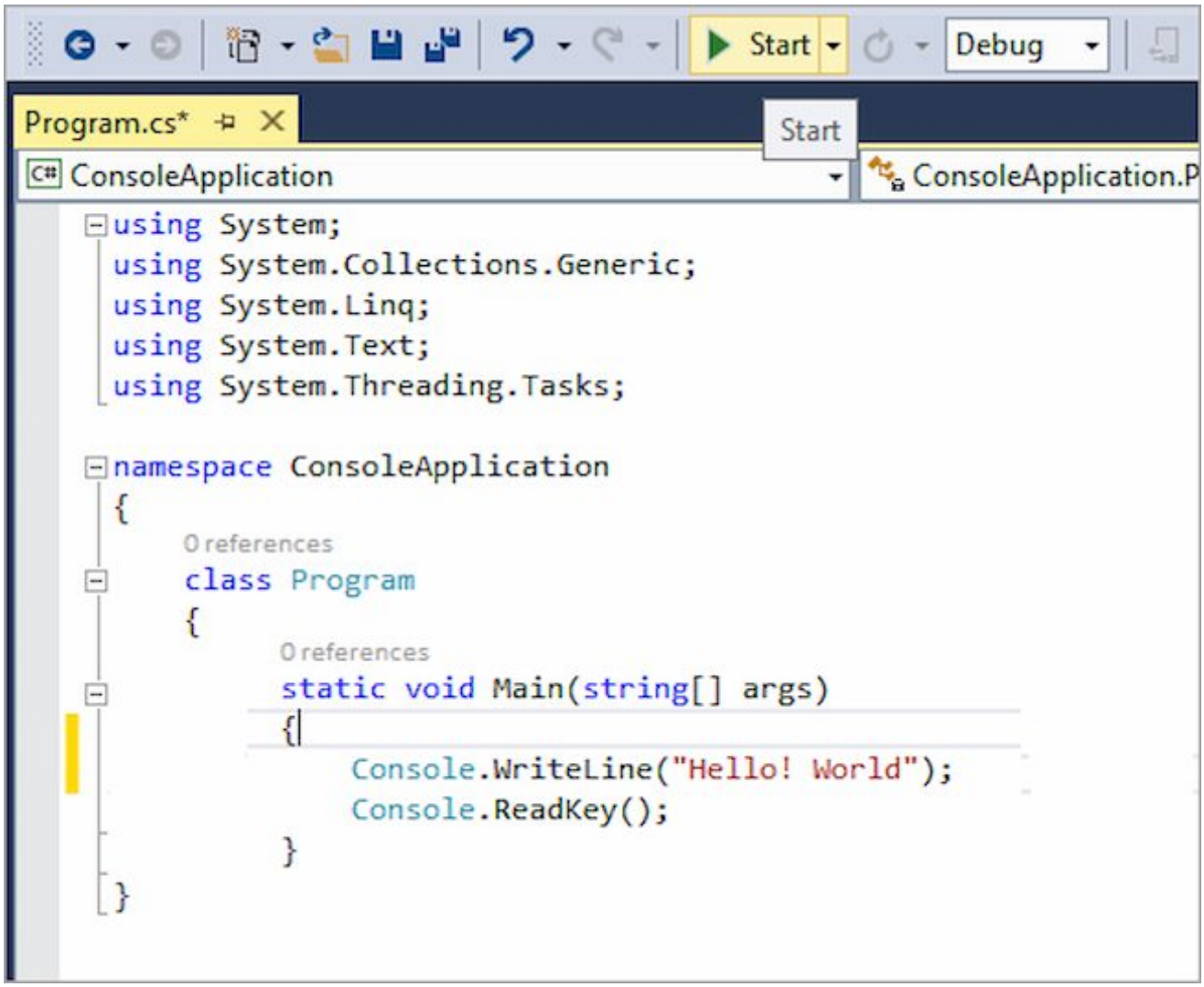
Python – популярный язык программирования для начинающих благодаря удобному синтаксису и универсальности. Большая часть кода Python читается как английский, что помогает начинающим изучать основные понятия, такие как функции. В Python также имеется множество библиотек кода. Это группы предварительно созданных функций, которые вы можете подключить к своему коду вместо того, чтобы писать функции самостоятельно. С помощью Python можно создавать множество различных типов программ. Многие вводные курсы также основывают свои проекты на этом языке.

C/C++



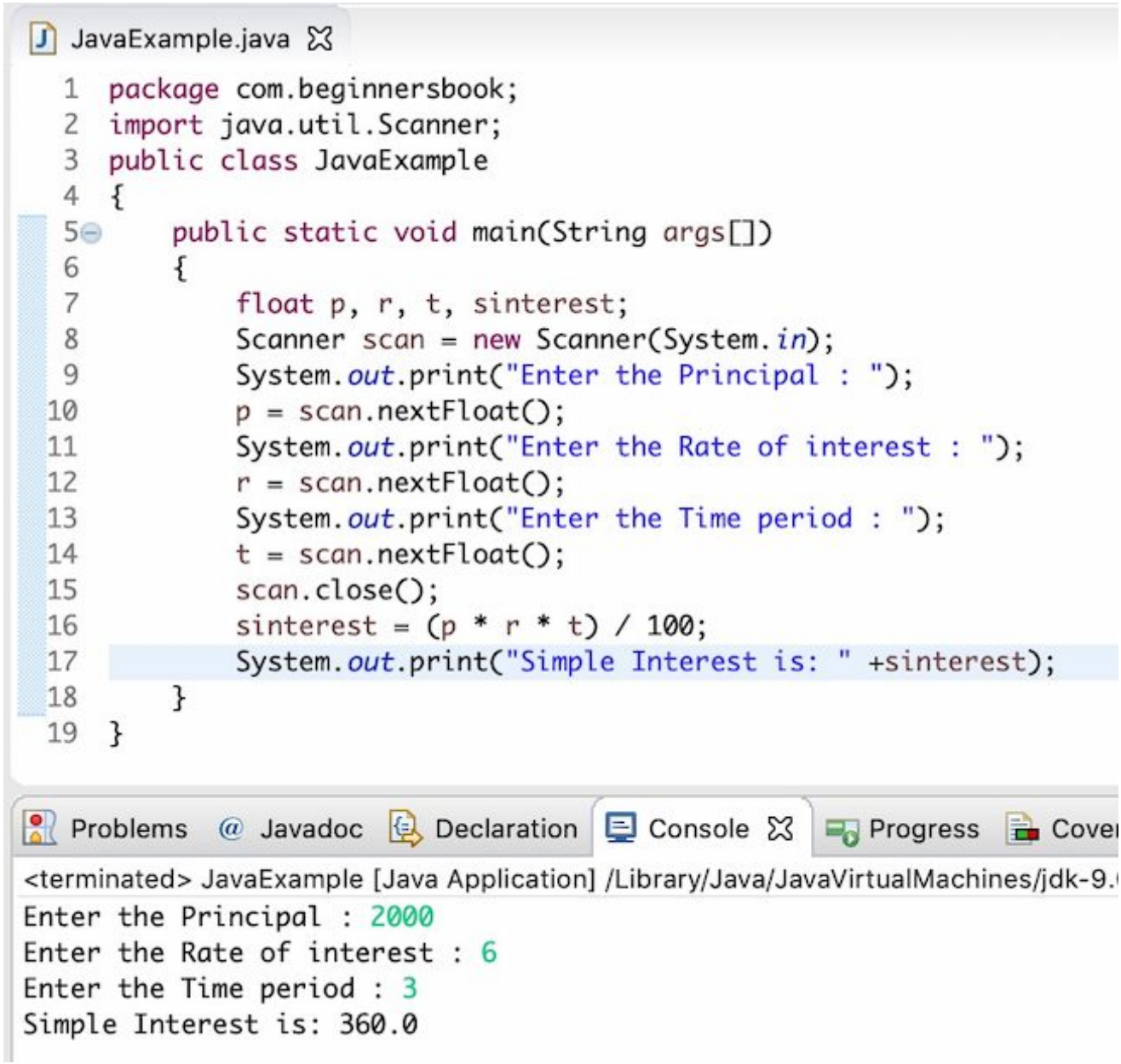
Си – еще один популярный выбор для вводных курсов. Он более сложен, чем Python, и часто требует от новичков писать больше кода для достижения тех же целей. Это больше работы, но полезно для понимания абстрактных концепций. Изучая C, вы приобретете навыки, которые можно будет легко применить к другим, более лаконичным языкам. C++ является преемником C. Синтаксис C++ похож на C с добавлением объектов – мощного типа переменных, который облегчает программирование сложных приложений. Я рекомендую начинать с языка C, так как в нем меньше концепций, которые можно перенести на C++.

C#



Язык C#, также называемый C-sharp, популярен для разработки видеоигр, мобильных и настольных приложений, а также корпоративного программного обеспечения. C# имеет общий источник с C++, но если C++ и C имеют много общего, то C# больше похож на Java. C# легко изучить, и существует множество способов его использования. Если вы планируете использовать и C++, и C#, возможно, вам стоит сначала изучить C++. Это облегчит понимание C# и поможет ускорить процесс обучения.

Java

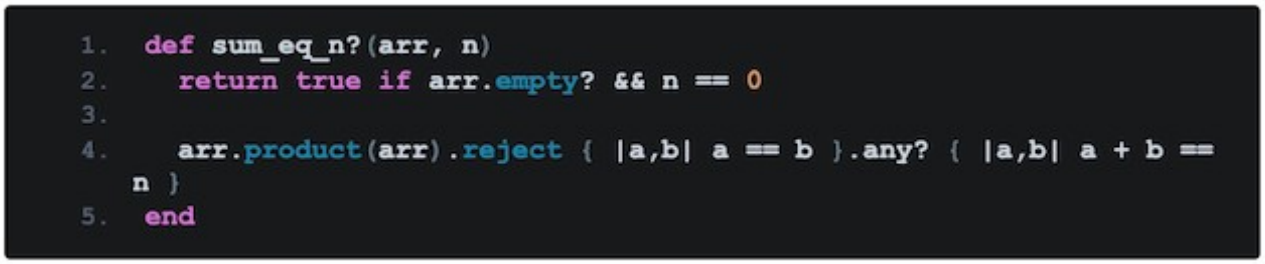


Java (не путать с JavaScript) – это объектно-ориентированный язык программирования общего назначения. Как и Python, синтаксис Java легко читается и понимается программистами-людьми – часто сложные задачи могут быть решены одной командой. Java популярно применяется в мобильных приложениях для android. Это еще один отличный базовый язык с принципами, которые можно интуитивно применять при изучении других языков.

HyperText Preprocessor (PHP)

PHP – это серверный язык программирования с открытым исходным кодом для разработки веб-приложений. Он позволяет легко добавлять на веб-сайты динамическую информацию, например, обновляемые новости. Вы также можете встроить этот язык в HTML, что позволяет легко добавлять функции на ваш сайт без использования внешних файлов. PHP также отлично подходит для работы с базами данных, упрощая доступ и хранение данных.

Ruby



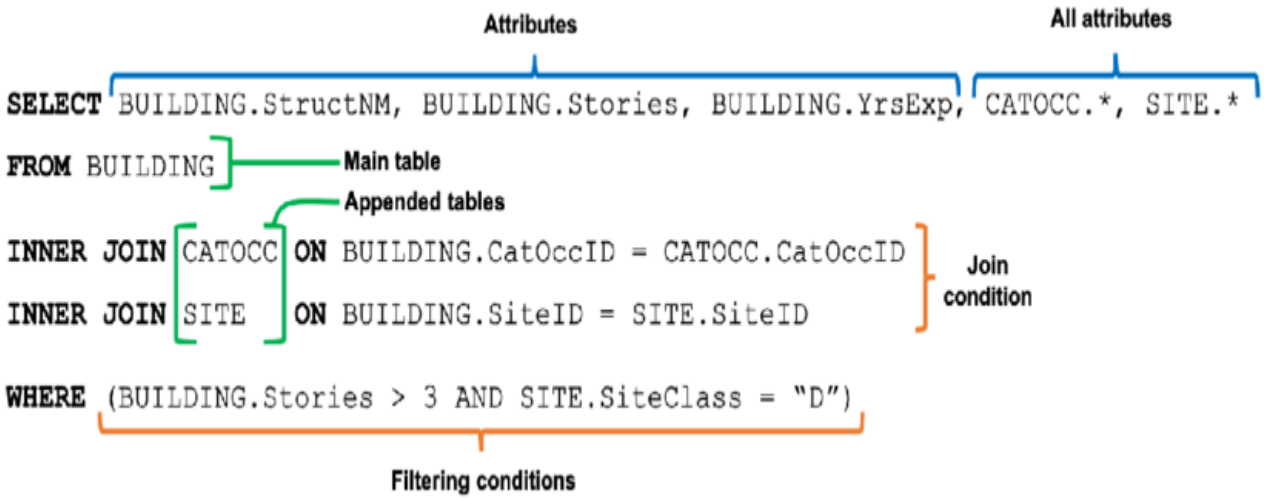
```ruby
def sum_eq_n?(arr, n)
  return true if arr.empty? && n == 0

  arr.product(arr).reject { |a,b| a == b }.any? { |a,b| a + b == n }
end
```

Ruby, также называемый Ruby on Rails, – это простой язык, который полезен для создания средств автоматизации, настольных приложений и быстрых прототипов. Ruby – это еще один язык программирования с открытым исходным кодом для серверной части. Его фреймворк также предлагает стандартные структуры для веб-страниц, веб-сервисов и баз данных. Это делает его полезным инструментом для

разработки веб-приложений. Хотя этот язык легко читать и писать, он может быть не лучшим выбором для вашего первого языка программирования. Это связано с тем, что он также довольно гибкий, что означает, что он легко принимает изменения. Гибкость – это здорово, когда вы только начинаете изучать, как сделать проект кодирования. Но она может повлиять на изменения в вашем коде, когда вы будете поддерживать проект с течением времени.

SQL



SQL – еще один полезный язык для начинающих. С помощью этого языка можно обновлять, хранить и извлекать данные из базы данных. Это также стандартный язык для систем управления базами данных, согласно Американскому национальному институту стандартов. Изучение SQL может занять некоторое время, если у вас еще нет определенных знаний в области программирования. Тем не менее, этот язык популярен в сфере технологий и среди специалистов по работе с данными, поэтому он имеет большое значение на рабочем месте.

Swift

```
struct Player {
    var name: String
    var highScore: Int = 0
    var history: [Int] = []

    init(_ name: String) {
        self.name = name
    }
}

var player = Player("Tomas")
```

Если вы учитесь кодировать, чтобы создавать проекты для устройств Apple, Swift – хороший язык для начала. Разработчики Apple создали этот язык с учетом пожеланий начинающих. И если ваша основная цель – разработка мобильных приложений для устройств iOS, этот язык должен стать вашим выбором. Хотя с 2014 года программисты создали большинство продуктов на Swift, вы также можете захотеть изучить Objective-C. Разработчики создали этот язык в 1980-х годах. Именно на нем их команда создала большинство инструментов для iOS. В нем используется некоторый синтаксис языка C, поэтому, если вы уже изучали C или C++, возможно, вам стоит начать с Objective-C.

Кодирование для начинающих

Существует множество способов подойти к практике кодирования, и некоторые методы и языки подойдут вам лучше, чем другие. Однако независимо от того, что и как вы изучаете, помните эти советы для начинающих:

Будьте терпеливы.

В начале этой статьи я сказал, что научиться кодировать может каждый. Но это не значит, что это легко. Кодинг требует, чтобы вы подходили к проблемам так, как раньше не подходили. Определенные темы могут показаться вам совершенно

бессмысленными, но при этом они являются ключевыми для изучаемого языка. Проблемы, которые поначалу могут показаться простыми, в итоге окажутся гораздо сложнее, чем вы ожидали. Вы можете часами корпеть над заданием, не видя результатов. Нельзя забывать и об отладке. Вы быстро узнаете, что компьютеры придирчивы и будут делать только то, что вы им скажете. Крошечные опечатки, такие как пропущенная точка с запятой или неправильный оператор, сломают всю вашу программу, и именно вам придется искать виновника.

Даже кодирование для начинающих – дело непростое.

Все это может быть неудобно и обескураживающе, и это нормально. Сделайте себе одолжение и продвигайтесь вперед медленно и неуклонно, давая себе время все впитать. Лучшие программисты когда-то были там же, где и вы, и всем приходится изучать одни и те же вещи, чтобы начать. Будьте проще к себе, придерживайтесь поставленных целей, делайте перерывы, и все будет хорошо.

Овладейте основами.

На начальном этапе изучения языка программирования главное – это основы. Вы можете начать с изучения двоичного кода, типов данных и печати на консоли. Затем вы изучите такие темы, как:

- Переменные
- Функции
- Условная логика
- Массивы
- Объекты

Очень важно, чтобы вы полностью усвоили каждое из этих понятий, прежде чем двигаться дальше. Это связано с тем, что все, что вы изучаете в информатике, основывается на предыдущих темах. Если что-то не понятно, продолжайте изучать это, пока не станет понятно. И не думайте, что это будет понятно позже, в контексте будущих уроков. Если вы учитесь по учебнику, убедитесь, что вы понимаете, чему должно научить каждое упражнение. Выполните все упражнения, чтобы на собственном опыте понять, как каждая тема применяется к кодированию. И не теряйте терпения – вы не сможете приступить к амбициозному проекту, пока не освоите основы.

Пишите чистый код с самого начала.

Вот чему онлайн-курс может вас не научить: Помимо изучения того, как писать код, вы также должны практиковаться в его написании. Что это значит? Для любого конкретного вычисления существует более одного способа его программирования. Вы всегда должны стремиться написать его наиболее кратким и читабельным способом. Разработчики обычно работают в командах, поэтому другие будут часто читать ваш код. Если его трудно расшифровать, коллеги-разработчики не захотят с вами работать. Даже если вы решите работать фрилансером, написание чистого кода гарантирует, что вы будете понимать свой собственный код. Лучше выработать привычку к чистому коду сейчас, так как это сэкономит вам часы на попытках расшифровать свою работу после того, как вы не заглядывали в нее несколько месяцев.

Почему чистый код важен?

Возможно, вы задаетесь вопросом, зачем вам изучать чистый код на данном этапе. Вы начинающий, так разве написание функциональных программ не должно быть главной целью? Ну, да. Речь идет о раннем формировании хороших привычек. Если вы приложите дополнительные усилия сейчас, вы сэкономите себе (и другим) немного здравомыслия в будущем. Хорошим способом достижения этой цели является сокращение строк и функций. Я рекомендую ограничивать каждую строку кода максимум 80 символами, а каждую функцию – не более чем 15 строками.

Хотя поначалу эти правила будут ограничены, они приучат вас отдавать предпочтение эффективному коду, а не первой пришедшей в голову идее. Кроме того, возьмите за привычку комментировать. Комментарии – это сегменты кода, которые не обрабатываются компьютером, поэтому вы можете писать в них все, что хотите. Программисты используют комментарии, чтобы прояснить цель своего кода. Изучите, как работают комментарии в вашем языке, и, по крайней мере, оставляйте комментарии в верхней части ваших функций, объясняя назначение каждой из них.

Поиск – ваш друг.

Нет ничего постыдного в том, чтобы использовать Google для поиска решений проблем кодирования. На самом деле, профессиональные разработчики делают это

постоянно. Если вы испытываете трудности, кто-то наверняка был в такой же ситуации и задал вопрос на форуме. Вы будете удивлены тем, сколько решений вы найдете, задавая сверхспецифические вопросы. Кроме того, очень приятно закрыть 20 вкладок Stack Exchange после того, как наконец-то исправили упрямую ошибку.

Кодинг – это больше, чем просто код

В заключение я хочу поделиться еще одним ценным советом из первого курса информатики. С точки зрения новичка может показаться, что научиться кодить – значит научиться писать код. В этом есть смысл: Когда мы представляем себе программиста, мы видим человека, пишущего код на компьютере – в конце концов, это называется “кодирование”. Но как только вы начнете, вы поймете, что это не вся история. Вы будете тратить гораздо больше времени на обдумывание того, что написать, чем на само написание.

Кодирование – это решение проблем.

Это потому, что кодирование – это больше решение проблем, чем знание синтаксиса. Изучение кода – это обучение мыслить, как это делают компьютеры, разлагать проблемы на составляющие и решать их с помощью предоставленных вам инструментов. Поэтому, да, вы научитесь писать впечатляющий код и в конечном итоге создавать удивительные вещи. Но сначала вы разовьете навыки мышления, которые помогут вам достичь этого. За годы работы над кодом такой подход к решению проблем изменил не только то, как я решаю технические задачи, но и то, как я подхожу к проблемам в целом. Я надеюсь, что вы испытаете то же самое. Желаю вам удачи на вашем пути. Не останавливайтесь на достигнутом. У вас все получится.

Дата Создания
14.04.2023