

VITON13SCHOOL

АТЛАС ЦИФРОВЫХ ТЕХНОЛОГИЙ

678 направлений, инструментов и учебных маршрутов
by VITON13

Практическое электронное издание

КАК ПОЛЬЗОВАТЬСЯ КНИГОЙ

Эта книга устроена как карта технологий. Не пытайтесь изучить всё подряд. Сначала выберите цель, затем один раздел и доведите обучение до измеримого результата.

ПРОТОКОЛ VITON13

1. **ОРИЕНТИР** — сформулируйте конкретный результат, который хотите получить.
2. **РАЗВЕДКА** — откройте 3–5 ресурсов из выбранного раздела и составьте карту понятий.
3. **ПРАКТИКА** — повторите рабочий пример своими руками, а не только прочитайте материал.
4. **ПРОЕКТ** — соберите самостоятельный мини-продукт, скрипт, исследование или прототип.
5. **ПРОВЕРКА** — протестируйте результат, объясните решения своими словами и исправьте слабые места.
6. **АРТЕФАКТ** — сохраните код, скриншоты, выводы и ссылку на результат в портфолио или базе знаний.

ТРИ УРОВНЯ

START — понять терминологию и запустить первый пример.

BUILD — соединить несколько инструментов в работающий проект.

MASTER — сравнить альтернативы, измерить качество и объяснить архитектурные компромиссы.

Правило VITON13: один законченный проект ценнее двадцати сохранённых ссылок.

СОДЕРЖАНИЕ

1. Платформы / Platforms — 57 ресурсов
2. Языки программирования / Programming Languages — 77 ресурсов
3. Front-End разработка / Front-End Development — 85 ресурсов
4. Back-End разработка / Back-End Development — 32 ресурсов
5. Компьютерные науки / Computer Science — 38 ресурсов
6. Большие данные / Big Data — 9 ресурсов
7. Теория / Theory — 14 ресурсов
8. Книги / Books — 6 ресурсов
9. Редакторы и среды / Editors — 5 ресурсов
10. Разработка игр / Gaming — 28 ресурсов
11. Среда разработки / Development Environment — 21 ресурсов
12. Цифровые развлечения / Entertainment — 5 ресурсов
13. Базы данных / Databases — 17 ресурсов
14. Медиа / Media — 14 ресурсов
15. Обучение / Learn — 15 ресурсов
16. Кибербезопасность / Security — 29 ресурсов
17. Системы управления контентом / Content Management Systems — 11 ресурсов
18. Аппаратные технологии / Hardware — 11 ресурсов
19. Бизнес / Business — 12 ресурсов
20. Работа и карьера / Work — 7 ресурсов
21. Сети / Networking — 5 ресурсов
22. Децентрализованные системы / Decentralized Systems — 17 ресурсов
23. Здоровье и социальные науки / Health and Social Science — 12 ресурсов
24. События / Events — 3 ресурсов
25. Тестирование / Testing — 11 ресурсов
26. Междисциплинарные направления / Miscellaneous — 132 ресурсов
27. Навигация и дополнительные каталоги / Related — 5 ресурсов

1. Платформы / Platforms

Платформа определяет среду, в которой живёт продукт: от мобильных устройств и облаков до IoT и desktop.

ИНСТРУКЦИЯ VITON13

Выберите одну платформу, разверните минимальное рабочее окружение и запустите демонстрационное приложение.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 57 ПОЗИЦИЙ

1. [Node.js](#) — Async non-blocking event-driven JavaScript runtime built on Chrome's V8 JavaScript engine.
2. [Cross-Platform](#) — Writing cross-platform code on Node.js.
3. [Frontend Development](#)
4. [iOS](#) — Mobile operating system for Apple phones and tablets.
5. [Android](#) — Mobile operating system developed by Google.
6. [IoT & Hybrid Apps](#)
7. [Electron](#) — Cross-platform native desktop apps using JavaScript/HTML/CSS.
8. [Cordova](#) — JavaScript API for hybrid apps.
9. [React Native](#) — JavaScript framework for writing natively rendering mobile apps for iOS and Android.
10. Linux
11. [Containers](#)
12. [eBPF](#) — Virtual machine that allows you to write more efficient and powerful tracing and monitoring for Linux systems.
13. [Arch-based Projects](#) — Linux distributions and projects based on Arch Linux.
14. [Applmage](#) — Package apps in a single file that works on various mainstream Linux distributions.
15. [Omarchy](#) — Opinionated Arch Linux and Hyprland desktop environment from the creator of Ruby on Rails.
16. macOS — Operating system for Apple's Mac computers.
17. [Screensavers](#)
18. [Apps](#)

19. [Open Source Apps](#)
20. [watchOS](#) — Operating system for the Apple Watch.
21. [JVM](#)
22. [Salesforce](#)
23. [Amazon Web Services](#)
24. [Windows](#) — Consumer desktop operating system.
25. [PowerToys Run Plugins](#) — Community plugins for the Windows quick launcher.
26. [IPFS](#) — P2P hypermedia protocol.
27. [Fuse](#) — Mobile development tools.
28. [Heroku](#) — Cloud platform as a service.
29. [Raspberry Pi](#) — Credit card-sized computer aimed at teaching kids programming, but capable of a lot more.
30. [WebExtensions](#) — Cross-browser extension system.
31. [Smart TV](#) — Create apps for different TV platforms.
32. [GNOME](#) — Simple and distraction-free desktop environment for Linux.
33. [KDE](#) — A free software community dedicated to creating an open and user-friendly computing experience.
34. [.NET](#)
35. [Core](#)
36. [Roslyn](#) — Open-source compilers and code analysis APIs for C# and VB.NET languages.
37. [DigitalOcean](#) — Cloud computing platform designed for developers.
38. [Flutter](#) — Google's mobile SDK for building native iOS and Android apps from a single codebase written in Dart.
39. [Home Assistant](#) — Open source home automation that puts local control and privacy first.
40. [IBM Cloud](#) — Cloud platform for developers and companies.
41. [Firebase](#) — App development platform built on Google Cloud.
42. [Adafruit IO](#) — Visualize and store data from any device.
43. [Cloudflare](#) — CDN, DNS, DDoS protection, and security for your site.
44. [Actions on Google](#) — Developer platform for Google Assistant.
45. [ESP](#) — Low-cost microcontrollers with WiFi and broad IoT applications.
46. [Deno](#) — A secure runtime for JavaScript and TypeScript that uses V8 and is built in Rust.
47. [DOS](#) — Operating system for x86-based personal computers that was popular during the 1980s and early 1990s.
48. [Nix](#) — Package manager for Linux and other Unix systems that makes package management reliable and reproducible.

49. [Integration](#) — Linking together different IT systems (components) to functionally cooperate as a whole.
50. [Node-RED](#) — A programming tool for wiring together hardware devices, APIs, and online services.
51. [Low Code](#) — Allowing business professionals to address their needs on their own with little to no coding skills.
52. [Capacitor](#) — Cross-platform open source runtime for building Web Native apps.
53. [ArcGIS Developer](#) — Mapping and location analysis platform for developers.
54. [Bluetooth Low Energy](#) — Low-power wireless communication protocol ideal for IoT, wearables, and other battery-powered applications.
55. [Uno Platform](#) — Open-source .NET UI platform for building cross-platform apps.
56. [Firebase Genkit](#) — An open-source framework for building AI-powered apps and features.
57. [Backstage](#) — Open-source platform for building Internal Developer Portals that unify tools and workflows.

2. Языки программирования / Programming Languages

Язык задаёт способ выражения идей, экосистему библиотек и компромиссы между скоростью разработки, безопасностью и производительностью.

ИНСТРУКЦИЯ VITON13

Выберите один язык, реализуйте небольшую CLI-утилиту и сравните экосистему библиотек с уже знакомым вам языком.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 77 ПОЗИЦИЙ

1. [JavaScript](#)
2. [Promises](#)
3. [Standard Style](#) — Style guide and linter.
4. [Must Watch Talks](#)
5. [Tips](#)
6. [Network Layer](#)
7. [Micro npm Packages](#)
8. [Mad Science npm Packages](#) — Impossible sounding projects that exist.
9. [Maintenance Modules](#) — For npm packages.
10. [npm](#) — Package manager.
11. [AVA](#) — Test runner.
12. [ESLint](#) — Linter.
13. [Functional Programming](#)
14. [Observables](#)
15. [npm scripts](#) — Task runner.
16. [30 Seconds of Code](#) — Code snippets you can understand in 30 seconds.
17. [Ponyfills](#) — Like polyfills but without overriding native APIs.
18. [Swift](#) — Apple's compiled programming language that is secure, modern, programmer-friendly, and fast.
19. [Education](#)
20. [Playgrounds](#)
21. [Python](#) — General-purpose programming language designed for readability.

22. [Asyncio](#) — Asynchronous I/O in Python 3.
23. [Scientific Audio](#) — Scientific research in audio/music.
24. [CircuitPython](#) — A version of Python for microcontrollers.
25. [Data Science](#) — Data analysis and machine learning.
26. [Typing](#) — Optional static typing for Python.
27. [MicroPython](#) — A lean and efficient implementation of Python 3 for microcontrollers.
28. [Rust](#)
29. [Pest](#) — Parser generator.
30. [Haskell](#)
31. [PureScript](#)
32. [Go](#)
33. [Scala](#)
34. [Scala Native](#) — Optimizing ahead-of-time compiler for Scala based on LLVM.
35. [Ruby](#)
36. [Clojure](#)
37. [ClojureScript](#)
38. [Elixir](#)
39. [Elm](#)
40. [Erlang](#)
41. [Julia](#) — High-level dynamic programming language designed to address the needs of high-performance numerical analysis and computational science.
42. [Lua](#)
43. [C/C++](#) — General-purpose language with a bias toward system programming and embedded, resource-constrained software.
44. [R](#) — Functional programming language and environment for statistical computing and graphics.
45. [Learning](#)
46. [D](#)
47. [Common Lisp](#) — Powerful dynamic multiparadigm language that facilitates iterative and interactive development.
48. [Learning](#)
49. [Perl](#)
50. [Groovy](#)
51. [Dart](#)
52. [Java](#) — Popular secure object-oriented language designed for flexibility to "write once, run anywhere".
53. [RxJava](#)
54. [J2ME](#) — Java specification designed for old keypad phones and PDAs..

55. [Kotlin](#)
56. [OCaml](#)
57. [ColdFusion](#)
58. [Fortran](#)
59. [PHP](#) — Server-side scripting language.
60. [Composer](#) — Package manager.
61. [Pascal](#)
62. [AutoHotkey](#)
63. [AutoIt](#)
64. [Crystal](#)
65. [CMake](#) — Build, test, and package software.
66. [ActionScript 3](#) — Object-oriented language targeting Adobe AIR.
67. [Idris](#) — General purpose pure functional programming language with dependent types influenced by Haskell and ML.
68. [Ada/SPARK](#) — Modern programming language designed for large, long-lived apps where reliability and efficiency are essential.
69. [Q#](#) — Domain-specific programming language used for expressing quantum algorithms.
70. [Imba](#) — Programming language inspired by Ruby and Python and compiles to performant JavaScript.
71. [Vala](#) — Programming language designed to take full advantage of the GLib and GNOME ecosystems, while preserving the speed of C code.
72. [Coq](#) — Formal language and environment for programming and specification which facilitates interactive development of machine-checked proofs.
73. [V](#) — Simple, fast, safe, compiled language for developing maintainable software.
74. [Move](#) — Domain-specific programming language for writing safe smart contracts.
75. [Esolangs](#) — Programming languages designed for experimentation or as jokes rather than actual use.
76. [VBA](#) — An event-driven version of Visual Basic 6.0 built into most Microsoft Office apps for automation and scripting.
77. [F#](#) — A .NET-based language with focus on functional programming.

3. Front-End разработка / Front-End Development

Front-end соединяет инженерную реализацию, интерфейс, доступность и воспринимаемую скорость продукта.

ИНСТРУКЦИЯ VITON13

Соберите адаптивный интерфейс с доступностью, измерьте производительность и оформите результат как мини-кейс.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 85 ПОЗИЦИЙ

1. [ES6 Tools](#)
2. [Web Performance Optimization](#)
3. [Web Tools](#)
4. [CSS](#) — Style sheet language that specifies how HTML elements are displayed on screen.
5. [Critical-Path Tools](#)
6. [Scalability](#)
7. [Must-Watch Talks](#)
8. [Protips](#)
9. [Frameworks](#)
10. [React](#) — JavaScript library for building user interfaces.
11. [Relay](#) — Framework for building data-driven React apps.
12. [React Hooks](#) — Lets you use state and other React features without writing a class.
13. [Web Components](#)
14. [Polymer](#) — JavaScript library to develop Web Components.
15. [Angular](#) — App framework.
16. [Backbone](#) — App framework.
17. [HTML5](#) — Markup language used for websites & web apps.
18. [SVG](#) — XML-based vector image format.
19. [Canvas](#)
20. [KnockoutJS](#) — JavaScript library.
21. [Dojo Toolkit](#) — JavaScript toolkit.
22. [Inspiration](#)
23. [Ember](#) — App framework.

24. [Android UI](#)
25. [iOS UI](#)
26. [Meteor](#)
27. [BEM](#)
28. [Flexbox](#)
29. [Web Typography](#)
30. [Web Accessibility](#)
31. [Material Design](#)
32. [D3](#) — Library for producing dynamic, interactive data visualizations.
33. [Emails](#)
34. [jQuery](#) — Easy to use JavaScript library for DOM manipulation.
35. [Tips](#)
36. [Web Audio](#)
37. [Offline-First](#)
38. [Static Website Services](#)
39. [Cycle.js](#) — Functional and reactive JavaScript framework.
40. [Text Editing](#)
41. [Motion UI Design](#)
42. [Vue.js](#) — App framework.
43. [Marionette.js](#) — App framework.
44. [Aurelia](#) — App framework.
45. [Charting](#)
46. [Ionic Framework](#)
47. [Chrome DevTools](#)
48. [PostCSS](#) — CSS tool.
49. [Draft.js](#) — Rich text editor framework for React.
50. [Service Workers](#)
51. [Progressive Web Apps](#)
52. [choo](#) — App framework.
53. [Redux](#) — State container for JavaScript apps.
54. [Browserify](#) — Module bundler.
55. [Sass](#) — CSS preprocessor.
56. [Ant Design](#) — Enterprise-class UI design language.
57. [Less](#) — CSS preprocessor.
58. [WebGL](#) — JavaScript API for rendering 3D graphics.
59. [Preact](#) — App framework.
60. [Progressive Enhancement](#)
61. [Next.js](#) — Framework for server-rendered React apps.
62. [lit](#) — Library for building web components with a declarative template system.

63. [JAMstack](#) — Modern web development architecture based on client-side JavaScript, reusable APIs, and prebuilt markup.
64. [WordPress-Gatsby](#) — Web development technology stack with WordPress as a back end and Gatsby as a front end.
65. [Mobile Web Development](#) — Creating a great mobile web experience.
66. [Storybook](#) — Development environment for UI components.
67. [Blazor](#) — .NET web framework using C#/Razor and HTML that runs in the browser with WebAssembly.
68. [PageSpeed Metrics](#) — Metrics to help understand page speed and user experience.
69. [Tailwind CSS](#) — Utility-first CSS framework for rapid UI development.
70. [Seed](#) — Rust framework for creating web apps running in WebAssembly.
71. [Web Performance Budget](#) — Techniques to ensure certain performance metrics for a website.
72. [Web Animation](#) — Animations in the browser with JavaScript, CSS, SVG, etc.
73. [Yew](#) — Rust framework inspired by Elm and React for creating multi-threaded frontend web apps with WebAssembly.
74. [Material-UI](#) — Material Design React components for faster and easier web development.
75. [Building Blocks for Web Apps](#) — Standalone features to be integrated into web apps.
76. [Svelte](#) — App framework.
77. [Design systems](#) — Collection of reusable components, guided by rules that ensure consistency and speed.
78. [Inertia.js](#) — Make single-page apps without building an API.
79. [MDBBootstrap](#) — Templates, layouts, components, and widgets to rapidly build websites.
80. [Master CSS](#) — A virtual CSS language with enhanced syntax.
81. [Hydrogen](#) — Edge-first framework for building Shopify storefronts with React.
82. [Tiny JS](#) — Frontend libraries that fit into 2 kB with dependencies.
83. [Frontend GIS](#) — Geographic Information Systems (GIS) for web browsers.
84. [WebGPU](#) — JavaScript API for rendering and compute on GPUs.
85. [WebAssembly](#) — A portable binary format for running code efficiently across platforms.

4. Back-End разработка / Back-End Development

Back-end отвечает за бизнес-логику, API, данные, интеграции, надёжность и масштабирование.

ИНСТРУКЦИЯ VITON13

Поднимите API с валидацией, логированием, тестами и базой данных; задокументируйте архитектурные решения.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 32 ПОЗИЦИЙ

1. [Flask](#) — Python framework.
2. [Docker](#)
3. [Vagrant](#) — Automation virtual machine environment.
4. [Pyramid](#) — Python framework.
5. [Play1 Framework](#)
6. [CakePHP](#) — PHP framework.
7. [Symfony](#) — PHP framework.
8. [Education](#)
9. [Laravel](#) — PHP framework.
10. [Education](#)
11. [Rails](#) — Web app framework for Ruby.
12. [Gems](#) — Packages.
13. [Phalcon](#) — PHP framework.
14. [Useful `.htaccess` Snippets](#)
15. [nginx](#) — Web server.
16. [Dropwizard](#) — Java framework.
17. [Kubernetes](#) — Open-source platform that automates Linux container operations.
18. [Lumen](#) — PHP micro-framework.
19. [Serverless Framework](#) — Serverless computing and serverless architectures.
20. [Apache Wicket](#) — Java web app framework.
21. [Vert.x](#) — Toolkit for building reactive apps on the JVM.
22. [Terraform](#) — Tool for building, changing, and versioning infrastructure.
23. [Vapor](#) — Server-side development in Swift.

24. [Dash](#) — Python web app framework.
25. [FastAPI](#) — Python web app framework.
26. [CDK](#) — Open-source software development framework for defining cloud infrastructure in code.
27. [IAM](#) — User accounts, authentication and authorization.
28. [Slim](#) — PHP framework.
29. [Fiber](#) — Web framework built on top of Fasthttp, the fastest HTTP engine for Go.
30. [Kustomize](#) — Kubernetes native declarative configuration management tool.
31. [OpenTofu](#) — Open-source infrastructure as code tool.
32. [Reflex](#) — Python web framework for building both your frontend and backend with no JavaScript.

5. Компьютерные науки / Computer Science

Фундаментальные знания позволяют понимать не только как использовать инструмент, но и почему система ведёт себя именно так.

ИНСТРУКЦИЯ VITON13

Возьмите одну фундаментальную тему, реализуйте алгоритм или структуру данных и объясните её сложность своими словами.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 38 ПОЗИЦИЙ

1. [University Courses](#)
2. [Data Science](#)
3. [Tutorials](#)
4. [Machine Learning](#)
5. [Tutorials](#)
6. [ML with Ruby](#) — Learning, implementing, and applying Machine Learning using Ruby.
7. [Core ML Models](#) — Models for Apple's machine learning framework.
8. [H2O](#) — Open source distributed machine learning platform written in Java with APIs in R, Python, and Scala.
9. [Software Engineering for Machine Learning](#) — From experiment to production-level machine learning.
10. [AI in Finance](#) — Solving problems in finance with machine learning.
11. [JAX](#) — Automatic differentiation and XLA compilation brought together for high-performance machine learning research.
12. [XAI](#) — Providing insight, explanations, and interpretability to machine learning methods.
13. [Speech and Natural Language Processing](#)
14. [Spanish](#)
15. [NLP with Ruby](#)
16. [Question Answering](#) — The science of asking and answering in natural language with a machine.
17. [Natural Language Generation](#) — Generation of text used in data-to-text, conversational agents, and narrative generation applications.
18. [Linguistics](#)

19. [Cryptography](#)
20. [Papers](#) — Theory basics for using cryptography by non-cryptographers.
21. [Computer Vision](#)
22. [Deep Learning](#) — Neural networks.
23. [TensorFlow](#) — Library for machine intelligence.
24. [TensorFlow.js](#) — WebGL-accelerated machine learning JavaScript library for training and deploying models.
25. [TensorFlow Lite](#) — Framework that optimizes TensorFlow models for on-device machine learning.
26. [Papers](#) — The most cited deep learning papers.
27. [Education](#)
28. [Deep Vision](#)
29. [Open Source Society University](#)
30. [Functional Programming](#)
31. [Empirical Software Engineering](#) — Evidence-based research on software systems.
32. [Static Analysis & Code Quality](#)
33. [Information Retrieval](#) — Learn to develop your own search engine.
34. [Quantum Computing](#) — Computing that utilizes quantum mechanics and qubits on quantum computers.
35. [Theoretical Computer Science](#) — The interplay of computer science and pure mathematics, distinguished by its emphasis on mathematical rigour and technique.
36. [Conversational AI](#) — Build awesome chatbots and digital assistants.
37. [Generative AI](#) — Automatically generates a wide range of unique content in text, image, and audio format.
38. [Position-Based Quantum Cryptography](#) — Theory on quantum cryptography that utilizes special relativistic constraints to achieve quantum-security under certain conditions.

6. Большие данные / Big Data

Работа с большими данными требует архитектуры, потоковой обработки, качества данных и наблюдаемости.

ИНСТРУКЦИЯ VITON13

Соберите небольшой pipeline: получение данных, очистка, хранение, вычисление метрик и визуализация результата.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 9 ПОЗИЦИЙ

1. [Big Data](#)
2. [Public Datasets](#)
3. [Hadoop](#) — Framework for distributed storage and processing of very large data sets.
4. [Data Engineering](#)
5. [Streaming](#)
6. [Apache Spark](#) — Unified engine for large-scale data processing.
7. [Qlik](#) — Business intelligence platform for data visualization, analytics, and reporting apps.
8. [Splunk](#) — Platform for searching, monitoring, and analyzing structured and unstructured machine-generated big data in real time.
9. [Network Analysis](#)

7. Теория / Theory

Теория даёт модели, с помощью которых легче выбирать правильные решения и оценивать их ограничения.

ИНСТРУКЦИЯ VITON13

Выберите одну теоретическую область и свяжите её с практическим примером в коде или системном дизайне.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 14 ПОЗИЦИЙ

1. [Papers We Love](#)
2. [Talks](#)
3. [Algorithms](#)
4. [Education](#) — Learning and practicing.
5. [Algorithm Visualizations](#)
6. [Artificial Intelligence](#)
7. [Search Engine Optimization](#)
8. [Competitive Programming](#)
9. [Math](#)
10. [Recursion Schemes](#) — Traversing nested data structures.
11. [Audit Algorithms](#) — Algorithmic audits of algorithms.
12. [AGI & CoCoSci](#) — The reciprocation of Artificial General Intelligence (AGI) and Computational Cognitive Sciences (CoCoSci).
13. [Complex Systems](#) — The scientific field studying systems with multiple interacting parts and emergent properties.
14. [VLM Architectures](#) — Vision Language Model architectures.

8. Книги / Books

Книги дают длинную связную траекторию обучения, которой часто не хватает фрагментарным урокам.

ИНСТРУКЦИЯ VITON13

Сформируйте маршрут из трёх книг: базовая, прикладная и углублённая; после каждой создавайте конспект и практический артефакт.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 6 ПОЗИЦИЙ

1. [Free Programming Books](#)
2. [Go Books](#)
3. [R Books](#)
4. [Mind Expanding Books](#)
5. [Book Authoring](#)
6. [Elixir Books](#)

9. Редакторы и среды / Editors

Хорошо настроенная среда разработки сокращает трение и делает повторяющиеся операции почти незаметными.

ИНСТРУКЦИЯ VITON13

Настройте рабочую среду под один реальный проект: навигация, форматирование, линтинг, отладка и быстрые команды.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 5 ПОЗИЦИЙ

1. [Sublime Text](#)
2. [Vim](#)
3. [Neovim](#) — Vim-fork focused on extensibility and usability.
4. [Emacs](#)
5. [Visual Studio Code](#) — Cross-platform open-source text editor.

10. Разработка игр / Gaming

Игры объединяют графику, физику, AI, звук, UI, нарратив и высокие требования к производительности.

ИНСТРУКЦИЯ VITON13

Соберите вертикальный срез игры: управление, одна механика, противник или препятствие, интерфейс и сборка.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 28 ПОЗИЦИЙ

1. [Game Development](#)
2. [Game Talks](#)
3. [Godot](#) — Game engine.
4. [Open Source Games](#)
5. [Chess](#)
6. [LÖVE](#) — Game engine.
7. [PICO-8](#) — Fantasy console.
8. [Game Boy Development](#)
9. [Construct 2](#) — Game engine.
10. [Gideros](#) — Game engine.
11. [Minecraft](#) — Sandbox video game.
12. [ComputerCraft](#) — Minecraft mod that adds programmable computers.
13. [Game Datasets](#) — Materials and datasets for Artificial Intelligence in games.
14. [Haxe Game Development](#) — A high-level strongly typed programming language used to produce cross-platform native code.
15. [libGDX](#) — Java game framework.
16. [PlayCanvas](#) — Game engine.
17. [Game Remakes](#) — Actively maintained open-source game remakes.
18. [Flame](#) — Game engine for Flutter.
19. [Discord Communities](#) — Chat with friends and communities.
20. [CHIP-8](#) — Virtual computer game machine from the 70s.
21. [Games of Coding](#) — Learn a programming language by making games.
22. [Esports](#) — Video games played as a sport.
23. [Learn Gamedev](#) — The craft of video game creation.

- 24. [Game Engine Development](#) — Building software to speed up game creation.
- 25. [GameMaker](#) — Game engine.
- 26. [Game Production](#) — Leading the process of designing, developing, testing and distributing a video game from concept to release.
- 27. [Babylon.js](#) — Game engine for cross-platform web and native game development.
- 28. [Roblox](#) — Immersive platform for hosting millions of diverse experiences.

11. Среда разработки / Development Environment

Воспроизводимая среда уменьшает ошибки «у меня работает» и ускоряет командную разработку.

ИНСТРУКЦИЯ VITON13

Автоматизируйте локальную разработку: установка, переменные окружения, команды запуска, тесты и воспроизводимая сборка.
Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 21 ПОЗИЦИЙ

1. [Quick Look Plugins](#) — For macOS.
2. [Dev Env](#)
3. [Dotfiles](#)
4. [Shell](#)
5. [Fish](#) — User-friendly shell.
6. [Command-Line Apps](#)
7. [ZSH Plugins](#)
8. [GitHub](#) — Hosting service for Git repositories.
9. [Browser Extensions](#)
10. [Cheat Sheet](#)
11. [Pinned Gists](#) — Dynamic pinned gists for your GitHub profile.
12. [Git Cheat Sheet & Git Flow](#)
13. [Git Tips](#)
14. [Git Add-ons](#) — Enhance the `git` CLI.
15. [Git Hooks](#) — Scripts for automating tasks during `git` workflows.
16. [SSH](#)
17. [FOSS for Developers](#)
18. [Hyper](#) — Cross-platform terminal app built on web technologies.
19. [Terminals Are Sexy](#)
20. [GitHub Actions](#) — Create tasks to automate your workflow and share them with others on GitHub.
21. [WezTerm](#) — Powerful cross-platform terminal emulator.

12. Цифровые развлечения / Entertainment

Творческие технологии расширяют понимание интерактивности, генеративных систем и цифрового опыта.

ИНСТРУКЦИЯ VITON13

Выберите одно творческое направление и создайте демонстрационный проект, который можно показать в портфолио.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 5 ПОЗИЦИЙ

1. [Science Fiction](#) — Scifi.
2. [Fantasy](#)
3. [Podcasts](#)
4. [Email Newsletters](#)
5. [IT Quotes](#)

13. Базы данных / Databases

Данные становятся ценными только тогда, когда их можно надёжно хранить, находить, изменять и восстанавливать.

ИНСТРУКЦИЯ VITON13

Спроектируйте модель данных, реализуйте CRUD, индексы и резервное копирование; затем проверьте типовые запросы.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 17 ПОЗИЦИЙ

1. [Database](#)
2. [MySQL](#)
3. [SQLAlchemy](#)
4. [InfluxDB](#)
5. [Neo4j](#)
6. [MongoDB](#) — NoSQL database.
7. [RethinkDB](#)
8. [TinkerPop](#) — Graph computing framework.
9. [PostgreSQL](#) — Object-relational database.
10. [CouchDB](#) — Document-oriented NoSQL database.
11. [HBase](#) — Distributed, scalable, big data store.
12. [NoSQL Guides](#) — Help on using non-relational, distributed, open-source, and horizontally scalable databases.
13. [Database Tools](#) — Everything that makes working with databases easier.
14. [TypeDB](#) — Logical database to organize large and complex networks of data as one body of knowledge.
15. [Supabase](#) — An open-source alternative to Firebase.
16. [PocketBase](#) — An open-source, Go-based backend in one file.
17. [Neon](#) — An open-source alternative to AWS Aurora.

14. Медиа / Media

Медиа-технологии охватывают создание, кодирование, обработку, доставку и анализ визуального и звукового контента.

ИНСТРУКЦИЯ VITON13

Создайте короткий pipeline для изображения, аудио или видео: импорт, обработка, экспорт и контроль качества.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 14 ПОЗИЦИЙ

1. [Fonts](#)
2. [Codeface](#) — Text editor fonts.
3. [Stock Resources](#)
4. [GIF](#) — Image format known for animated images.
5. [Music](#)
6. [Open Source Documents](#)
7. [Audio Visualization](#)
8. [Broadcasting](#)
9. [Pixel Art](#) — Pixel-level digital art.
10. [FFmpeg](#) — Cross-platform solution to record, convert and stream audio and video.
11. [Icons](#) — Downloadable SVG/PNG/font icon projects.
12. [Audiovisual](#) — Lighting, audio and video in professional environments.
13. [VLC](#) — Cross-platform media player software and streaming server.
14. [Audio Over IP](#) — Reliably transmit uncompressed, low-latency audio over an IP (layer 3) network.

15. Обучение / Learn

Сильная система обучения превращает хаотичный поток материалов в последовательный рост навыков.

ИНСТРУКЦИЯ VITON13

Соберите персональную программу на 30 дней: одна тема в день, одно упражнение и еженедельный мини-проект.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 15 ПОЗИЦИЙ

1. [CLI Workshoppers](#) — Interactive tutorials.
2. [Learn to Program](#)
3. [Speaking](#)
4. [Tech Videos](#)
5. [Computer History](#)
6. [Programming for Kids](#)
7. [STEAM](#) — Supplements for specific science, technology, engineering, arts, and math (STEAM) courses across all grade levels.
8. [Educational Games](#) — Learn while playing.
9. [JavaScript Learning](#)
10. [CSS Learning](#) — Mainly about CSS – the language and the modules.
11. [Product Management](#) — Learn how to be a better product manager.
12. [Roadmaps](#) — Gives you a clear route to improve your knowledge and skills.
13. [YouTubers](#) — Watch video tutorials from YouTubers that teach you about technology.
14. [Hackathon](#) — Running fun and productive hackathons.
15. [Certificates](#) — Free computer science certifications to showcase your knowledge.

16. Кибербезопасность / Security

Безопасность — это управление риском: активы, угрозы, уязвимости, контроль доступа, наблюдаемость и восстановление.

ИНСТРУКЦИЯ VITON13

Работайте только с собственными системами или с явным разрешением. Постройте защитный стенд: инвентаризация, проверка конфигурации, логирование, обновления и отчёт.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

Этическое правило: любые проверки безопасности выполняйте только на собственных системах или при явном разрешении владельца.

КАТАЛОГ: 29 ПОЗИЦИЙ

1. [Application Security](#)
2. [Security](#)
3. [CTF](#) — Capture The Flag.
4. [Cyber Security University](#) — Free educational resources that focus on learning by doing.
5. [Malware Analysis](#)
6. [Android Security](#)
7. [Hacking](#)
8. [Hacking Spots](#)
9. [Honeypots](#) — Deception trap, designed to entice an attacker into attempting to compromise the information systems in an organization.
10. [Incident Response](#)
11. [Vehicle Security and Car Hacking](#)
12. [Web Security](#) — Security of web apps & services.
13. [Lockpicking](#) — The art of unlocking a lock by manipulating its components without the key.
14. [Cybersecurity Blue Team](#) — Groups of individuals who identify security flaws in information technology systems.
15. [Fuzzing](#) — Automated software testing technique that involves feeding pseudo-randomly generated input data.
16. [Embedded and IoT Security](#)

17. [GDPR](#) — Regulation on data protection and privacy for all individuals within EU.
18. [DevSecOps](#) — Integration of security practices into [DevOps](https://en.wikipedia.org/wiki/DevOps).
19. [Executable Packing](#) — Packing and unpacking executable formats.
20. [Malware Persistence](#) — Techniques that adversaries use to keep access to systems across restarts.
21. [EVM Security](#) — Understanding the Ethereum Virtual Machine security ecosystem.
22. [Password Cracking](#) — The process of recovering passwords from data that has been stored in or transmitted by a system in scrambled form.
23. [Security Card Games](#) — Train your skills and discuss various security topics.
24. [Suricata](#) — Intrusion detection/prevention system and network security monitoring engine.
25. [Prompt Injection](#) — A type of vulnerability that specifically targets machine learning models.
26. [Detection Engineering](#) — Design, build, and operate detective cybersecurity controls.
27. [Annual Security Reports](#) — Exploring cybersecurity trends, insights, and challenges.
28. [CI/CD Attacks](#) — Offensive research of systems and processes related to developing and deploying code.
29. [OpenID Connect](#) — Identity standard and authentication protocol built on OAuth 2.0 for user identity assertion.

17. Системы управления контентом / Content Management Systems

CMS отделяет контент от разработки и позволяет командам быстрее публиковать и поддерживать цифровые продукты.

ИНСТРУКЦИЯ VITON13

Разверните CMS, создайте структуру контента, роли, шаблон страницы и резервное копирование.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 11 ПОЗИЦИЙ

1. [Umbraco](#)
2. [Refinery CMS](#) — Ruby on Rails CMS.
3. [Wagtail](#) — Django CMS focused on flexibility and user experience.
4. [Textpattern](#) — Lightweight PHP-based CMS.
5. [Drupal](#) — Extensible PHP-based CMS.
6. [Craft CMS](#) — Content-first CMS.
7. [Sitecore](#) — .NET digital marketing platform that combines CMS with tools for managing multiple websites.
8. [Silverstripe CMS](#) — PHP MVC framework that serves as a classic or headless CMS.
9. [Directus](#) — A real-time API and app dashboard for managing SQL database content.
10. [Plone](#) — Open source Python CMS.
11. [Payload](#) — Next.js native and open source headless CMS.

18. Аппаратные технологии / Hardware

Аппаратные проекты соединяют код с физическим миром и учат учитывать питание, интерфейсы, ограничения и надёжность.

ИНСТРУКЦИЯ VITON13

Соберите безопасный прототип устройства или симуляцию: датчик/ввод, логика, вывод и документация схемы.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 11 ПОЗИЦИЙ

1. [Robotics](#)
2. [Internet of Things](#)
3. [Electronics](#) — For electronic engineers and hobbyists.
4. [Bluetooth Beacons](#)
5. [Electric Guitar Specifications](#) — Checklist for building your own electric guitar.
6. [Plotters](#) — Computer-controlled drawing machines and other visual art robots.
7. [Robotic Tooling](#) — Free and open tools for professional robotic development.
8. [LIDAR](#) — Sensor for measuring distances by illuminating the target with laser light.
9. [Open Hardware](#) — Open-source hardware projects.
10. [ADS-B](#) — Technology broadcasting aircraft's identity, position, and data periodically.
11. [Flying FPV](#) — Open hardware and software related to drones / UAVs.

19. Бизнес / Business

Технология имеет коммерческую ценность, когда решает понятную проблему и её эффект можно измерить.

ИНСТРУКЦИЯ VITON13

Выберите одну бизнес-задачу и создайте измеримый эксперимент: гипотеза, канал, метрика, срок и критерий успеха.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 12 ПОЗИЦИЙ

1. [Open Companies](#)
2. [Places to Post Your Startup](#)
3. [OKR Methodology](#) — Goal setting & communication best practices.
4. [Indie](#) — Independent developer businesses.
5. [Tools of the Trade](#) — Tools used by companies on Hacker News.
6. [Clean Tech](#) — Fighting climate change with technology.
7. [Wardley Maps](#) — Provides high situational awareness to help improve strategic planning and decision making.
8. [Social Enterprise](#) — Building an organization primarily focused on social impact that is at least partially self-funded.
9. [Engineering Team Management](#) — How to transition from software development to engineering management.
10. [Developer-First Products](#) — Products that target developers as the user.
11. [Billing](#) — Payments, invoicing, pricing, accounting, marketplace, fraud, and business intelligence.
12. [Engineering Strategy](#) — How to design and execute engineering strategies for tech leadership.

20. Работа и карьера / Work

Навыки нужно уметь превращать в доказательства компетентности и понятный рынок труда.

ИНСТРУКЦИЯ VITON13

Обновите карьерный стек: портфолио, резюме, профиль, процесс откликов и таблицу измерения конверсии.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 7 ПОЗИЦИЙ

1. [Slack](#) — Team collaboration.
2. [Communities](#)
3. [Remote Jobs](#)
4. [Productivity](#)
5. [Niche Job Boards](#)
6. [Code Review](#) — Reviewing code.
7. [Creative Technology](#) — Businesses & groups that specialize in combining computing, design, art, and user experience.

21. Сети / Networking

Сети связывают системы; понимание протоколов и инфраструктуры критично для производительности и устойчивости.

ИНСТРУКЦИЯ VITON13

Соберите тестовую сеть или лабораторию, опишите топологию, адресацию, наблюдаемость и сценарий отказа.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 5 ПОЗИЦИЙ

1. [PCAPTools](#)
2. [Real-Time Communications](#) — Network protocols for near simultaneous exchange of media and data.
3. [SNMP](#) — A protocol for collecting, modifying, and organizing information about managed devices on IP networks.
4. [Scapy](#) — Python-based interactive packet manipulation.
5. [Cilium](#) — Provides networking and security capabilities for containerized apps, microservices, and virtual machines.

22. Децентрализованные системы / Decentralized Systems

Децентрализация меняет модель доверия, владения данными и согласования состояния между участниками.

ИНСТРУКЦИЯ VITON13

Постройте учебный прототип с прозрачной моделью доверия, угроз и ограничений; не используйте реальные деньги без необходимости.
Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 17 ПОЗИЦИЙ

1. [Bitcoin](#) — Services and tools for Bitcoin software developers.
2. [Ripple](#) — Open source distributed settlement network.
3. [Non-Financial Blockchain](#) — Applications of Blockchain beyond finance.
4. [Mastodon](#) — Open source decentralized microblogging network.
5. [Ethereum](#) — Distributed computing platform for smart contract development.
6. [Blockchain AI](#) — AI and machine-learning projects built on Blockchain.
7. [EOSIO](#) — A decentralized operating system supporting industrial-scale apps.
8. [Corda](#) — Open source Blockchain platform designed for business.
9. [Waves](#) — Open source Blockchain platform and development toolset for Web 3.0 apps and decentralized solutions.
10. [Substrate](#) — Framework for writing scalable, upgradeable blockchains in Rust.
11. [Golem](#) — Open source peer-to-peer marketplace for computing resources.
12. [Stacks](#) — A smart contract platform secured by Bitcoin.
13. [Algorand](#) — An open-source, proof of stake Blockchain and smart contract computing platform.
14. [ZeroNet](#) — A decentralized web-like network of peer-to-peer users.
15. [Cosmos SDK](#) — Modular framework for building app-specific blockchains in Go.
16. [Tor](#) — A free overlay network for enabling anonymous communication.
17. [ATProto](#) — Open, decentralized network for building social apps.

23. Здоровье и социальные науки / Health and Social Science

Эти области показывают, как вычислительные методы взаимодействуют с человеком, обществом и научными данными.

ИНСТРУКЦИЯ VITON13

Выберите исследовательскую задачу, сформулируйте вопрос, источник данных, метод анализа и ограничения интерпретации.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 12 ПОЗИЦИЙ

1. [Biomedical Information Extraction](#) — How to extract information from unstructured biomedical data and text.
2. [Computational Neuroscience](#) — A multidisciplinary science which uses computational approaches to study the nervous system.
3. [Diversity](#) — Creating a more inclusive and diverse tech community.
4. [Digital History](#) — Computer-aided scientific investigation of history.
5. [Empathy in Engineering](#) — Building and promoting more compassionate engineering cultures.
6. [Healthcare](#) — Open source healthcare software for facilities, providers, developers, policy experts, and researchers.
7. [Mental Health](#) — Awareness and self-care in the software industry.
8. [Neuroscience](#) — Study of the nervous system and brain.
9. [Digital Humanities](#) — Software for humanities scholars using quantitative or computational methods.
10. [Lucid Dreams](#) — A dream where one becomes aware they are dreaming.
11. [Neuroimaging](#) — Software for analyzing brain data from living subjects.
12. [Transgender](#) — Someone whose gender identity differs from their assigned birth sex.

24. События / Events

События ускоряют обучение через обмен опытом, новые связи и наблюдение за практиками профессионального сообщества.

ИНСТРУКЦИЯ VITON13

Сформируйте календарь профессионального развития и для каждого события заранее задайте цель, вопросы и ожидаемый результат.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 3 ПОЗИЦИЙ

1. [Creative Tech Events](#) — Events around the globe for creative coding, tech, design, music, arts and cool stuff.
2. [Events in Italy](#) — Tech-related events in Italy.
3. [Events in the Netherlands](#) — Tech-related events in the Netherlands.

25. Тестирование / Testing

Тестирование превращает ожидания от системы в проверяемые утверждения и снижает цену изменений.

ИНСТРУКЦИЯ VITON13

Покройте один небольшой проект модульными, интеграционными и end-to-end тестами и измерьте, какие дефекты они реально ловят. Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 11 ПОЗИЦИЙ

1. [Testing](#) — Software testing.
2. [Visual Regression Testing](#) — Ensures changes did not break the functionality or style.
3. [Selenium](#) — Open-source browser automation framework and ecosystem.
4. [Appium](#) — Test automation tool for apps.
5. [TAP](#) — Test Anything Protocol.
6. [JMeter](#) — Load testing and performance measurement tool.
7. [k6](#) — Open-source, developer-centric performance monitoring and load testing solution.
8. [Playwright](#) — Node.js library to automate Chromium, Firefox and WebKit with a single API.
9. [Quality Assurance Roadmap](#) — How to start & build a career in software testing.
10. [Gatling](#) — Open-source load and performance testing framework based on Scala, Akka, and Netty.
11. [CodeRabbit](#) — AI-powered code review platform.

26. Междисциплинарные направления / Miscellaneous

Междисциплинарные темы часто дают самые нестандартные идеи и связывают технологии с реальными задачами.

ИНСТРУКЦИЯ VITON13

Выберите одну необычную область и превратите её в междисциплинарный мини-проект с чётким результатом.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 132 ПОЗИЦИЙ

1. [Scientific Writing](#) — Distraction-free scientific writing with Markdown, reStructuredText and Jupyter notebooks.
2. [JSON](#) — Text based data interchange format.
3. [GeoJSON](#)
4. [CSV](#) — A text file format that stores tabular data and uses a comma to separate values.
5. [Discounts for Student Developers](#)
6. [Radio](#)
7. [Метакаталог](#) — рекурсивная навигация по тематическим подборкам.
8. [Analytics](#)
9. [REST](#)
10. [Continuous Integration and Continuous Delivery](#)
11. [Services Engineering](#)
12. [Free for Developers](#)
13. [Boilerplate Projects](#)
14. [Readme](#)
15. [GitHub Wiki](#) — Comprehensive documentation on GitHub beyond README.
16. [Design and Development Guides](#)
17. [Software Engineering Blogs](#)
18. [Self Hosted](#)
19. [FOSS Production Apps](#)
20. [Gulp](#) — Task runner.
21. [AMA](#) — Ask Me Anything.
22. [Answers](#)

23. [Open Source Photography](#)
24. [OpenGL](#) — Cross-platform API for rendering 2D and 3D graphics.
25. [GraphQL](#)
26. [Urban & Regional Planning](#) — Concerning the built environment and communities.
27. [Transit](#) — Data standards, APIs, apps, tools, datasets, and research around open source technology of public transit.
28. [Research Tools](#)
29. [Data Visualization](#)
30. [Microservices](#)
31. [Unicode](#) — Standards, quirks, packages and resources for Unicode.
32. [Code Points](#)
33. [Beginner-Friendly Projects](#)
34. [Katas](#)
35. [Tools for Activism](#)
36. [Citizen Science](#) — For community-based and non-institutional scientists.
37. [MQTT](#) — "Internet of Things" connectivity protocol.
38. [For Girls](#)
39. [Vorpal](#) — Node.js CLI framework.
40. [Vulkan](#) — Low-overhead, cross-platform 3D graphics and compute API.
41. [LaTeX](#) — Typesetting language.
42. [Economics](#) — An economist's starter kit.
43. [Funny Markov Chains](#)
44. [Bioinformatics](#)
45. [Cheminformatics](#) — Informatics techniques applied to problems in chemistry.
46. [Colorful](#) — Choose your next color scheme.
47. [Steam](#) — Digital distribution platform.
48. [Bots](#) — Building bots.
49. [Site Reliability Engineering](#)
50. [DTrace](#) — Dynamic tracing framework.
51. [Userscripts](#) — Enhance your browsing experience.
52. [Pokémon](#) — Resources for Pokémon and Pokémon GO.
53. [ChatOps](#) — Managing technical and business operations through a chat.
54. [Falsehood](#) — Falsehoods programmers believe in.
55. [Domain-Driven Design](#) — Software development approach for complex needs by connecting the implementation to an evolving model.
56. [Quantified Self](#) — Self-tracking through technology.
57. [Web Design](#) — For digital designers.

58. [Creative Coding](#) — Programming something expressive instead of something functional.
59. [No-Login Web Apps](#) — Web apps that work without login.
60. [Free Software](#) — Free as in freedom.
61. [Framer](#) — Prototyping interactive UI designs.
62. [Markdown](#) — Markup language.
63. [Dev Fun](#) — Funny developer projects.
64. [Magento 2](#) — Open Source eCommerce built with PHP.
65. [TikZ](#) — Graph drawing packages for TeX/LaTeX/ConTeXt.
66. [Ad-Free](#) — Alternatives without ads.
67. [Prometheus](#) — Open-source monitoring system.
68. [Homematic](#) — Smart home devices.
69. [Ledger](#) — Double-entry accounting on the command-line.
70. [Web Monetization](#) — A free open web standard service that allows you to send money directly in your browser.
71. [Uncopyright](#) — Public domain works.
72. [Crypto Currency Tools & Algorithms](#) — Digital currency where encryption is used to regulate the generation of units and verify transfers.
73. [Open Source Supporters](#) — Companies that offer their tools and services for free to open source projects.
74. [Design Principles](#) — Create better and more consistent designs and experiences.
75. [Theravada](#) — Teachings from the Theravada Buddhist tradition.
76. [inspectIT](#) — Open source Java app performance management tool.
77. [Open Source Maintainers](#) — The experience of being an open source maintainer.
78. [Calculators](#) — Tools for every platform.
79. [Captcha](#) — A type of challenge–response test used in computing to determine whether or not the user is human.
80. [Jupyter](#) — Create and share documents that contain code, equations, visualizations and narrative text.
81. [FIRST Robotics Competition](#) — International high school robotics championship.
82. [Speakers](#) — Conference and meetup speakers in the programming and design community.
83. [Board Games](#) — Table-top gaming fun for all.
84. [Software Patreons](#) — Fund individual programmers or the development of open source projects.
85. [Parasite](#) — Parasites and host-pathogen interactions.
86. [Food](#) — Projects on GitHub related to food.

87. [Bitcoin Payment Processors](#) — Start accepting Bitcoin.
88. [Scientific Computing](#) — Solving complex scientific problems using computers.
89. [Amazon Sellers](#)
90. [Agriculture](#) — Open source technology for farming and gardening.
91. [Product Design](#) — Design a product from the initial concept to production.
92. [Prisma](#) — Turn your database into a GraphQL API.
93. [Software Architecture](#) — The discipline of designing and building software.
94. [Connectivity Data and Reports](#) — Better understand who has access to telecommunication and internet infrastructure and on what terms.
95. [Stacks](#) — Tech stacks for building different apps and features.
96. [Cytodata](#) — Image-based profiling of biological phenotypes for computational biologists.
97. [IRC](#) — Open source messaging protocol.
98. [Advertising](#) — Programmatic media and ad tech for websites.
99. [Earth](#) — Find ways to resolve the climate crisis.
100. [Naming](#) — Guides for naming things in computer science.
101. [Web Archiving](#) — An effort to preserve the Web for future generations.
102. [WP-CLI](#) — Command-line interface for WordPress.
103. [Credit Modeling](#) — Methods for classifying credit applicants into risk classes.
104. [Ansible](#) — A Python-based, open-source IT configuration management and automation platform.
105. [Biological Visualizations](#) — Interactive visualization of biological data on the web.
106. [QR Code](#) — A type of matrix barcode that can be used to store and share a small amount of information.
107. [Veganism](#) — Making the plant-based lifestyle easy and accessible.
108. [Translations](#) — The transfer of the meaning of a text from one language to another.
109. [Scriptable](#) — An iOS app for automation in JavaScript.
110. [WebXR](#) — Enables immersive virtual reality and augmented reality content on the web.
111. [Computational Geometry](#) — Computational approaches for problems in geometry.
112. [OpenStreetMap](#) — An open data mapping project utilized by many apps and devices.

113. [Computational Biology](#) — Computational approaches applied to problems in biology.
114. [Read the Docs](#) — Example documentation projects to inspire and help bootstrap new documentation projects.
115. [Quarto](#) — Scientific and technical open-source publishing system built on Pandoc.
116. [Biological Image Analysis](#) — Interpreting biological phenomena using images.
117. [ChatGPT](#) — Artificial intelligence chatbot developed by OpenAI.
118. [Whisper](#) — Open-source AI-powered speech recognition system developed by OpenAI.
119. [Stock Trading](#) — Purchase and sale of equities of publicly traded companies to generate profits.
120. [Steam Deck](#) — A handheld gaming computer developed by Valve.
121. [Astrophotography](#) — Photography of astronomical objects, celestial events, or areas of the night sky.
122. [HPC](#) — High Performance Computing.
123. [Geocaching](#) — Outdoor treasure-hunting activity that uses GPS-enabled devices.
124. [Regex](#) — Specialized language for matching patterns in text.
125. [Event-Driven Architecture](#) — A software architecture approach where services collaborate by publishing and subscribing to events.
126. [Permacomputing](#) — Resilient and regenerative computing practices inspired by permaculture.
127. [Standards](#) — Proposals and standards that define and enhance software, languages, and related technologies.
128. [Claude Code](#) — Terminal-based AI coding assistant by Anthropic.
129. [Gemini CLI](#) — Terminal-based AI coding assistant by Google.
130. [SAP Commerce](#) — An e-commerce platform built with Java, Spring MVC, and Angular.
131. [Tech Ethics](#) — Mitigating and avoiding the potential negative effects of technology on society.
132. [Copilot Agents](#) — AI pair programming assistant by GitHub that provides code suggestions and completions.

27. Навигация и дополнительные каталоги / Related

Дополнительная навигация помогает расширять карту знаний, не превращая обучение в бесконечное накопление ссылок.

ИНСТРУКЦИЯ VITON13

Используйте навигационные инструменты для расширения собственного учебного графа и регулярно очищайте его от устаревших или дублирующихся ресурсов.

Рабочий цикл: цель -> 3 ресурса -> практика -> мини-проект -> проверка -> артефакт.

КАТАЛОГ: 5 ПОЗИЦИЙ

1. [Глобальные тематические каталоги](#) — карта тематических коллекций.
2. [Поиск по тематическим каталогам](#) — быстрый поиск по коллекциям.
3. [Случайный навигатор](#) — случайные материалы из большой базы.
4. [CLI-навигатор](#) — командная навигация по коллекциям.
5. [Трекер обновлений](#) — отслеживание изменений коллекций.

ФИНАЛЬНЫЙ МАРШРУТ VITON13

Не измеряйте прогресс количеством открытых страниц. Измеряйте его количеством вещей, которые вы способны спроектировать, построить, проверить и объяснить.

Рекомендуемый цикл на 90 дней:

Дни 1–30: одна фундаментальная область + ежедневная практика.
Дни 31–60: один прикладной стек + проект средней сложности.
Дни 61–90: интеграция нескольких областей + итоговый продукт + документация.