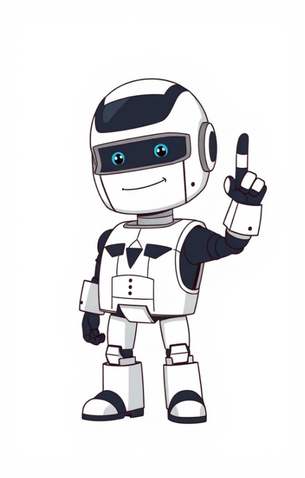
Continue



You cant perform that action at this time. You cant perform that action at this time. A demonstration animation of a code editor using GitHub Copilot Chat, where the user requests GitHub Copilot to refactor duplicated logic and extract it into a reusable function for a given code snippet.Write, test, and fix code quickly with GitHub Copilot, from simple boilerplate to complex features.From your first line of code to final deployment, GitHub provides AI and automation tools to help you build and ship better software faster.A Copilot chat window with the 'Ask' mode enabled. The user switches from 'Ask' mode to 'Agent' mode from a dropdown menu, then sends the prompt 'Update the website to allow searching for running races by name.' Copilot analyzes the codebase, then explains the required edits for three files before generating them. Copilot then confirms completion and summarizes the implemented changes for the new functionality allowing users to search races by name and view paginated, filtered results.70% MTTR reduction with Copilot Autofix8.3M secret leaks stopped in the past 12 months with push protectionFrom planning and discussion to code review, GitHub keeps your teams conversation and context next to your code.It helps us onboard new software engineers and get them productive right away. We have all our source code, issues, and pull requests in one place... GitHub is a complete platform that frees us from menial tasks and enables us to do our best work.Whether youre scaling your development process or just learning how to code, GitHub is where you belong. Join the worlds most widely adopted developer platform to build the technologies that shape whats next. You cant perform that action at this time. You cant perform that action at this time. GitHub is a cloud-based platform where you can store, share, and work together with others to write code.Storing your code in a "repository" on GitHub allows you to:Showcase or share your work.Track and manage changes to your code over time.Let others review your code, and make suggestions to improve it.Collaborate on a shared project, without worrying that your changes will impact the work of your collaborators before you're ready to integrate them.Collaborative working, one of GitHubs fundamental features, is made possible by the open-source software, Git, upon which GitHub is built.About GitGit is a version control system that intelligently tracks changes in files. Git is particularly useful when you and a group of people are all making changes to the same files at the same time.Typically, to do this in a Git-based workflow, you would:Create a branch off from the main copy of files that you (and your collaborators) are working on.Make edits to the files independently and safely on your own personal branch.Let Git intelligently merge your specific changes back into the main copy of files, so that your changes don't impact other people's updates.Let Git keep track of your and other people's changes, so you all stay working on the most up-to-date version of the project.To try using Git yourself, see Getting started with Git.How do Git and GitHub work together?When you upload files to GitHub, you'll store them in a "Git repository." This means that when you make changes (or "commits") to your files in GitHub, Git will automatically start to track and manage your changes.There are plenty of Git-related actions that you can complete on GitHub directly in your browser, such as creating a Git repository, creating branches, and uploading and editing files.However, most people work on their files locally (on their own computer), then continually sync these local changesand all the related Git datawith the central "remote" repository on GitHub. There are plenty of tools that you can use to do this, such as GitHub Desktop.Once you start to collaborate with others and all need to work on the same repository at the same time, youll continually:Pull all the latest changes made by your collaborators from the remote repository on GitHub.Push back your own changes to the same remote repository on GitHub.Git figures out how to intelligently merge this flow of changes, and GitHub helps you manage the flow through features such as "pull requests."Where do I start?If you're new to GitHub, and unfamiliar with Git, we recommend working through the articles in the Start your journey category. The articles focus on tasks you can complete directly in your browser on GitHub and will help you to:Create an account on GitHub.Learn the "GitHub Flow", and the key principles of collaborative working (branches, commits, pull requests, merges).Personalise your profile to share your interests and skills.Explore GitHub to find inspiration for your own projects and connect with others.Learn how to download interesting code for your own use.Learn how to upload something you're working on to a GitHub repository.Next stepsCreating an account on GitHubFurther reading GitHub DesktopWhether you're new to Git or a seasoned user, GitHub Desktop simplifies your development workflow. You cant perform that action at this time. 2025 GitHub, Inc.TermsPrivacyStatusPricingExpert servicesBlog The complete developer platform to build, scale, and deliver secure software.Read up on product innovations and updates, company announcements, community spotlights, and more.Want to use Mona the octocat? Looking for the right way to display the GitHub logo for your latest project? Download the assets and see how and where to use them.Developers are building the future on GitHub every day, explore their stories, celebrate their accomplishments, and find inspiration for your own work.See how some of the most influential businesses around the world use GitHub to provide the best services, products, and experiences for their customers.Help us build the home for all developers. Were a passionate group of people dedicated to software development and collaboration. Come join us!We are dedicated to building a community and team that reflects the world we live in and pushes the boundaries of software innovation.We are always monitoring the status of github.com and all its related services. Updates and status interruptions are posted in real-time here.Meet the leadership team guiding us as we continue on this journey building the worlds largest and most advanced software development platform in the world.Dive into the details with our annual State of the Octoverse report looking at the trends and patterns in the code and communities that build on GitHub.Were focused on fighting for developer rights by shaping the policies that promote their interests and the future of software.Explore the latest press stories on our company, products, and global community.Learn about how GitHubs people, products, and platform are creating positive and lasting change around the world. You cant perform that action at this time. Innovate faster with seamless collaboration.Spin up fully configured dev environments in the cloud with the full power of your favorite editor.Get suggestions for whole lines of code or entire functions right inside your editor.Receive notifications of contributor changes to a repository, with specified access limits, and seamlessly merge accepted updates.Dedicated space for your community to come together, ask and answer questions, and have open-ended conversations.Rapidly search, navigate, and understand code right from GitHub.com with our powerful new tools.Review new code, visualize changes, and merge confidently with automated status checks.Collaborate and discuss changes without a formal review or the risk of unwanted merges.Enforce branch merge restrictions by requiring reviews or limiting access to specific contributors.Automate everything: CI/CD, testing, planning, project management, issue labeling, approvals, onboarding, and moreAutomate your software workflows by writing tasks and combining them to build, test, and deploy faster from GitHub.Host your own software packages or use them as dependencies in other projects, with both private and public hosting available.Create calls to get all the data and events you need within GitHub, and automatically kick off and advance your software workflows.Leverage thousands of actions and applications from our community to help build, improve, and accelerate your workflows.Dozens of events and a webhooks API help you integrate with and automate work for your repository, organization, or application.Move automation to the cloud with on-demand Linux, macOS, Windows, ARM, and GPU environments for your workflow runs, all hosted by GitHub.Gain more environments and fuller control with labels, groups, and policies to manage runs on your own machines, plus an open source runner application.Map workflows, track their progression in real time, understand complex workflows, and communicate status with the rest of the team.Standardize and scale best practices and processes with preconfigured workflow templates shared across your organization.Application security where found means fixed. Powered by GitHub Copilot Autofix.Find vulnerabilities in your code with CodeQL, GitHubs industry-leading semantic code analysis. Prevent new vulnerabilities from being introduced by scanning every pull request.Powered by GitHub Copilot, generate automatic fixes for 90% of alert types in JavaScript, TypeScript, Java, and Python. Quickly remediate with contextual vulnerability intelligence and advice.Solve your backlog of application security debt with security campaigns that target and generate autofixes for up to 1,000 alerts at a time, rapidly reducing the risk of vulnerabilities and zero-day attacks.Detect exposed secrets in your public and private repositories, and revoke them to secure access to your services.Additional AI capabilities to detect elusive secrets like passwords.View the packages your project relies on, the repositories that depend on them, and any vulnerabilities detected in their dependencies.Receive alerts when new vulnerabilities affect your repositories, with GitHub detecting and notifying you of vulnerable dependencies in both public and private repositories.Keep your code secure by automatically opening pull requests that update vulnerable or out-of-date dependencies.Assess the security impact of new dependencies in pull requests before merging.Privately report, discuss, fix, and publish information about security vulnerabilities found in open source repositories.Enable your public repository to privately receive vulnerability reports from the community and collaborate on solutions.Browse or search GitHub's database of known vulnerabilities, featuring curated CVEs and security advisories linked to the GitHub dependency graph.Access GitHub anywhere: On Desktop, Mobile, and Command Line.Take your projects, ideas, and code to go with fully native mobile and tablet experiences.Manage issues and pull requests from the terminal, where you're already working with Git and your code.Simplify your development workflow with a GUI to visualize, commit, and push changesno command line needed.Keep feature requests, bugs, and more organized.Create a customized view of your issues and pull requests to plan and track your work.Track bugs, enhancements, and other requests, prioritize work, and communicate with stakeholders as changes are proposed and merged.Track progress on groups of issues or pull requests in a repository, and map groups to overall project goals.Leverage insights to visualize your projects by creating and sharing charts built from your project's data.View vulnerabilities, licenses, and other important information for the open source projects your organization depends on.Use data about activity, trends, and contributions within your repositories, to make data-driven improvements to your development cycle.Host project documentation in a wiki within your repository, allowing contributors to easily edit it on the web or locally.Simplify access and permissions management across your projects and teams.Create groups of user accounts that own repositories and manage access on a team-by-team or individual user basis.Organize your members to mirror your company's structure, with cascading access to permissions and mentions.Enable team synchronization between your identity provider and your organization on GitHub, including Entra ID and Okta.Define users' access level to your code, data, and settings based on their role in your organization.Ensure members have only the permissions they need by creating custom roles with fine-grained permission settings.Verify your organization's identity on GitHub and display that verification through a profile badge.Take care of your security assessment and certification needs by accessing GitHubs cloud compliance reports, such as our SOC reports and Cloud Security Alliance CAIQ self-assessments (CSA CAIQ).Quickly review the actions performed by members of your organization. Monitor access, permission changes, user changes, and other events.Enhance your organization's security with scalable source code protections, and use rule insights to easily review how and why code changes occurred in your repositories.Requires GitHub EnterpriseEnable collaboration between your organization and GitHub environments with a single point of visibility and management via an enterprise account.Requires GitHub EnterpriseShare features and workflows between your GitHub Enterprise Server instance and GitHub Enterprise Cloud.Requires GitHub EnterpriseSecurely control access to organization resources like repositories, issues, and pull requests with SAML, while allowing users to authenticate with their GitHub usernames.Requires GitHub EnterpriseCentralize repository management. LDAP is one of the most common protocols used to integrate third-party software with large company user directories.Requires GitHub EnterpriseManage the lifecycle and authentication of users on GitHub Enterprise Cloud from your identity provider (IdP).Requires GitHub EnterpriseUse the SSO and SCIM providers of your choice for Enterprise Managed Users, separate from one another, for a more flexible approach to user lifecycle management.Financially support the open source projects your code depends on. Sponsor a contributor, maintainer, or project with one time or recurring contributions.Learn new skills by completing tasks and projects directly within GitHub, guided by our friendly bot.Write cross-platform desktop applications using JavaScript, HTML, and CSS with the Electron framework, based on Node.js and Chromium.GitHub Education is a commitment to bringing tech and open source collaboration to students and educators across the globe. You cant perform that action at this time. Everything you need to know about Git, from getting started to advanced commands and workflows.Quick links: Git is a distributed version control software. Version control is a way to save changes over time without overwriting previous versions. Being distributed means that every developer working with a Git repository has a copy of that entire repository every commit, every branch, every file. If you're used to working with centralized version control systems, this is a big difference!Whether or not you've worked with version control before, there are a few things you should know before getting started with Git:Branches are lightweight and cheap, so it's OK to have many of themGit stores changes in SHA hashes, which work by compressing text files. That makes Git a very good version control system (VCS) for software programming, but not so good for binary files like images or videos.Git repositories can be connected, so you can work on one locally on your own machine, and connect it to a shared repository. This way, you can push and pull changes to a repository and easily collaborate with others. The tools that make up the core Git distribution are written in C, Shell, Perl, and Tcl. You can find Git's source code on GitHub under git/git. Version control is very important without it, you risk losing your work. With Git, you can make a "commit", or a save point, as often as you'd like. You can also go back to previous commits. This takes the pressure off of you while you're working. Commit often and commit early, and you'll never have that gut-sinking feeling of overwriting or losing changes.There are many version control systems out there but Git has some major advantages. Like we mentioned above, Git uses SHA compression, which makes it very fast. Git can handle merge conflicts, which means that it's OK for multiple people to work on the same file at the same time. This opens up the world of development in a way that isn't possible with centralized version control. You have access to the entire project, and if you're working on a branch, you can do whatever you need to and know that your changes are safe. Speaking of branches, Git offers a lot of flexibility and opportunity for collaboration with branches. By using branches, developers can make changes in a safe sandbox.Instead of only committing code that is 100% sure to succeed, developers can commit code that might still need help. Then, they can push that code to the remote and get fast feedback from integrated tests or peer review.Without sharing the code through branches, this would never be possible. If you make a mistake, it's OK! Commits are immutable, meaning they can't be changed. (Note: You can change history, but it will create new replacement commits instead of editing the existing commits. More on that later!) This means that if you do make a mistake, even on an important branch, like main, it's OK. You can easily revert that change, or roll back the branch pointer to the commit where everything was fine.The benefits of this can't be overstated. Not only does it create a safer environment for the project and code, but it fosters a development environment where developers can be braver, trusting that Git has their back. If you're getting started with Git, a great place to learn the basic commands is the Git Cheat sheet. It's translated into many languages, open source as a part of the github/training-kit repository, and a great starting place for the fundamentals on the command line.Some of the most important and most used commands that you'll find there are:git clone [url]: Clone (download) a repository that already exists on GitHub, including all of the files, branches, and commits.git status: Always a good idea, this command shows you what branch you're on, what files are in the working or staging directory, and any other important information.git branch: This shows the existing branches in your local repository. You can also use git branch [branch-name] to create a branch from your current location, or git branch --all to see all branches, both the local ones on your machine, and the remote tracking branches stored from the last git pull or git fetch from the remote.git checkout [branch-name]: Switches to the specified branch and updates the working directory.git add [file]: Snapshots the file in preparation for versioning, adding it to the staging area.git commit -m "descriptive message": Records file snapshots permanently in the version history.git pull: Updates your current local working branch with all new commits from the corresponding remote branch on GitHub. git pull is a combination of git fetch and git merge.git push: Uploads all local branch commits to the remote.git log: Browse and inspect the evolution of project files.git remote -v: Show the associated remote repositories and their stored name, like origin.If you're looking for more GitHub-specific technical guidance, check out GitHub's help documentation or our GitHub for Developers series on YouTube. Depending on your operating system, you may already have Git installed. But, getting started means more than having the software! To get started, it's important to know the basics of how Git works. You may choose to do the actual work within a terminal, an app like GitHub Desktop, or through GitHub.com. (Note: while you can interact with Git through GitHub.com, your experience may be limited. Many local tools can give you access to the most widely used Git functionalities, though only the terminal will give you access to them all.)There are many ways to use Git, which doesn't necessarily make it easier! But, the fundamental Git workflow has a few main steps. You can practice all of these in the Introduction to GitHub Learning Lab course. The main branch is usually called main. We want to work on another branch, so we can make a pull request and make changes safely. To get started, create a branch off of main. Name it however you'd like but we recommend naming branches based on the function or feature that will be the focus of this branch. One person may have several branches, and one branch may have several people collaborate on it branches are for a purpose, not a person. Wherever you currently "are" (wherever HEAD is pointing, or whatever branch you're currently "checked out" to) will be the parent of the branch you create. That means you can create branches from other branches, tags, or any commit! But, the most typical workflow is to create a branch from main which represents the most current production code. Once you've created a branch, and moved the HEAD pointer to it by "checking out" to that branch, you're ready to get to work. Make the changes in your repository using your favorite text editor or IDE.Next, save your changes. You're ready to start the commit!To start your commit, you need to let Git know what changes you'd like to include with git add [file].Once you've saved and staged the changes, you're ready to make the commit with git commit -m "descriptive commit message". So far, if you've made a commit locally, you're the only one that can see it. To let others see your work and begin collaboration, you should "push" your changes using git push. If you're pushing from a branch for the first time that you've created locally, you may need to give Git some more information. git push -u origin [branch-name] tells Git to push the current branch, and create a branch on the remote that matches it with the same name and also, create a relationship with that branch so that git push will be enough information in the future.By default, git push only pushes the branch that you've currently checked out to.Sometimes, if there has been a new commit on the branch on the remote, you may be blocked from pushing. Don't worry! Start with a simple git pull to incorporate the changes on the remote into your own local branch, resolve any conflicts or finish the merge from the remote into the local branch, and then try the push again. Pushing a branch, or new commits, to a remote repository is enough if a pull request already exists, but if it's the first time you're pushing that branch, you should open a new pull request. A pull request is a comparison of two branches typically main, or the branch that the feature branch was created from, and the feature branch. This way, like branches, pull requests are scoped around a specific function or addition of work, rather than the person making the changes or the amount of time the changes will take.Pull requests are the powerhouse of GitHub. Integrated tests can automatically run on pull requests, giving you immediate feedback on your code. Peers can give detailed code reviews, letting you know if there are changes to make, or if it's ready to go.Make sure you start your pull requests off with the right information. Put yourself in the shoes of your teammates, or even of your future self. Include information about what this change relates to, what prompted it, what is already done, what is left to do, and any specific asks for help or reviews. Include links to relevant work or conversations. Pull request templates can help make this process easy by automating the starting content of the body of pull requests. Once the pull request is open, then the real fun starts. It's important to recognize that pull requests aren't meant to be open when work is finished. Pull requests should be open when work is beginning! The earlier you open a pull request, the more visibility the entire team has to the work that you're doing. When you're ready for feedback, you can get it by integrating tests or requesting reviews from teammates.It's very likely that you will want to make more changes to your work. That's great! To do that, make more commits on the same branch. Once the new commits are present on the remote, the pull request will update and show the most recent version of your work. Once you and your team decide that the pull request looks good, you can merge it. By merging, you integrate the feature branch into the other branch (most typically the main branch). Then, main will be updated with your changes, and your pull request will be closed. Don't forget to delete your branch! You won't need it anymore. Remember, branches are lightweight and cheap, and you should create a new one when you need it based on the most recent commit on the main branch.If you choose not to merge the pull request, you can also close pull requests with unmerged changes. If you're wondering where Git ends and GitHub begins, you're not alone. They are tied closely together to make working with them both a seamless experience. While Git takes care of the underlying version control, GitHub is the collaboration platform built on top of it. GitHub is the place for pull requests, comments, reviews, integrated tests, and so much more. Most developers work locally to develop and use GitHub for collaboration. That ranges from using GitHub to host the shared remote repository to working with colleagues and capitalizing on features like protected branches, code review, GitHub Actions, and more.The best place to practice using Git and GitHub is the Introduction to GitHub Learning Lab course.If you already know Git and need to sign up for a GitHub account, head over to github.com. Contribute to this article on GitHub.

Math chart for 9th class. Formula chart 9th grade. 9th class formulas maths. Class 9th maths all formulas chart. 9th maths formula. Formula chart 9th class. Formula chart of maths class 9. 9th std maths formulas.