

# Git

Efe ÇİFTÇİ, November 2022  
Çankaya University, Department of  
Computer Engineering



# What is Version Control System?

- Version control systems are responsible for managing and keeping track of changes on program source codes, their documentations, etc.
- Users can store changed versions of files belonging to a project.
- Each change in the system is identified as a “revision”.
- At any time, these revisions can be revisited, compared, and restored.
- Multiple people can access the tracked projects as team members.

# What is Version Control System?

```
1 #include <stdio.h>
2
3 int main() {
4     return 0;
5 }
6 |
```

# What is Version Control System?

```
1 #include <stdio.h>
2
3 int main() {
4     int a, b, result;
5     printf("Please enter a and b: ");
6     scanf("%d %d", &a, &b);
7     result = a + b;
8     printf("Sum of a and b is %d\n", result);
9     return 0;
10 }
11
```

# What is Version Control System?

```
1 #include <stdio.h>
2
3 int main() {
4     int a, b;
5     float result;
6     printf("Please enter a and b: ");
7     scanf("%d %d", &a, &b);
8     result = a + b;
9     printf("Sum of a and b is %d\n", (int) result);
10    result = a / (float) b;
11    printf("a divided by b is %f\n", result);
12    return 0;
13 }
```

# What is Version Control System?

File: main.c [sampleproject]

Id	Message	Author	Authored Date
d8b7e70	<b>master</b> HEAD implemented division operation	Efe Çiftci	1 seconds ago
45f977b	implemented addition operation	Efe Çiftci	2 minutes ago
219bfac	initial version	Efe Çiftci	5 minutes ago

# What is Version Control System?

```
diff --git a/main.c b/main.c
index dc79d60..0a744d2 100644
--- a/main.c
+++ b/main.c
@@ -1,5 +1,10 @@
#include <stdio.h>

int main() {
+ int a, b, result;
+ printf("Please enter a and b: ");
+ scanf("%d %d", &a, &b);
+ result = a + b;
+ printf("Sum of a and b is %d\n", result);
return 0;
}
```

# What is Version Control System?

```
diff --git a/main.c b/main.c
index 0a744d2..e2a9b1f 100644
--- a/main.c
+++ b/main.c
@@ -1,10 +1,13 @@
#include <stdio.h>

int main() {
- int a, b, result;
+ int a, b;
+ float result;
printf("Please enter a and b: ");
scanf("%d %d", &a, &b);
result = a + b;
- printf("Sum of a and b is %d\n", result);
+ printf("Sum of a and b is %d\n", (int) result);
+ result = a / (float) b;
+ printf("a divided by b is %f\n", result);
return 0;
}
```



# History of Git



- Prior to 2005, development and maintenance of Linux kernel was handled with BitKeeper, a proprietary source control system.
- When the owning company of BitKeeper revoked free-of-charge license from Linux developers, Linus Torvalds and other kernel developers started developing their own source control system.
- Thus Git was born.

# Using Git

- Developers can start managing their projects using Git via multiple ways:
  - By creating a local Git repository (i.e., on local computer),
  - By building a local Git server (i.e., on a machine in the company),
  - By using a public Git hosting service (e.g., GitHub).
- Git can be used both from command line interface and GUI applications.

# Using GitHub

- Create a new github.com account,
- Install Git on your computer:
  - <https://git-scm.com/downloads>
- Create a repository at GitHub,
- Clone the repository into a directory on your computer,
- Develop your project,
- Push local changes to your online GitHub repository.

# Using GitHub

The screenshot shows the GitHub user dashboard for the user 'efeciftci'. The top navigation bar includes a search bar, 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. The left sidebar shows the user's profile, 'Recent Repositories' (with a 'New' button and a search bar), a list of repositories, and 'Recent activity'. The main content area is titled 'The home for all developers — including you.' and contains a welcome message, a 'Start writing code' button, and two primary cards: 'Start a new repository' (highlighted with a red box) and 'Introduce yourself with a profile README'. The 'Start a new repository' card includes a text input for the repository name ('ceng105demo'), radio buttons for 'Public' and 'Private' visibility, and a 'Create a new repository' button. The 'Introduce yourself with a profile README' card includes a 'Create' button and a preview of a README file with five lines of text. A 'Latest changes' section on the right lists recent updates. At the bottom, there are partial views of 'Simplify your development workflow' and 'Install a powerful code editor' cards.

efeciftci

Recent Repositories New

Find a repository...

CankayaUniversity/ceng-407-408-2021-2022-Ambilight-Media-Player

efeciftci/mosquito-simulator

efeciftci/ip-subnet-exercise

efeciftci/lkd-uye-plasmoid

efeciftci/bil611bfs

Recent activity

When you take actions across GitHub, we'll provide links to that activity here.

## The home for all developers — including you.

Welcome to your personal dashboard, where you can find an introduction to how GitHub works, tools to help you build software, and help merging your first lines of code.

[Start writing code](#)

### Start a new repository

A repository contains all of your project's files, revision history, and collaborator discussion.

efeciftci /

**Public**  
Anyone on the internet can see this repository

**Private**  
You choose who can see and commit to this repository

Create a new repository

### Introduce yourself with a profile README

Share information about yourself by creating a profile README, which appears at the top of your profile page.

efeciftci / README.md Create

```
1 - 👋 Hi, I'm @efeciftci
2 - ** I'm interested in ...
3 - 🌱 I'm currently learning ...
4 - 💖 I'm looking to collaborate
5 - 📫 How to reach me ...
6
```

### Latest changes

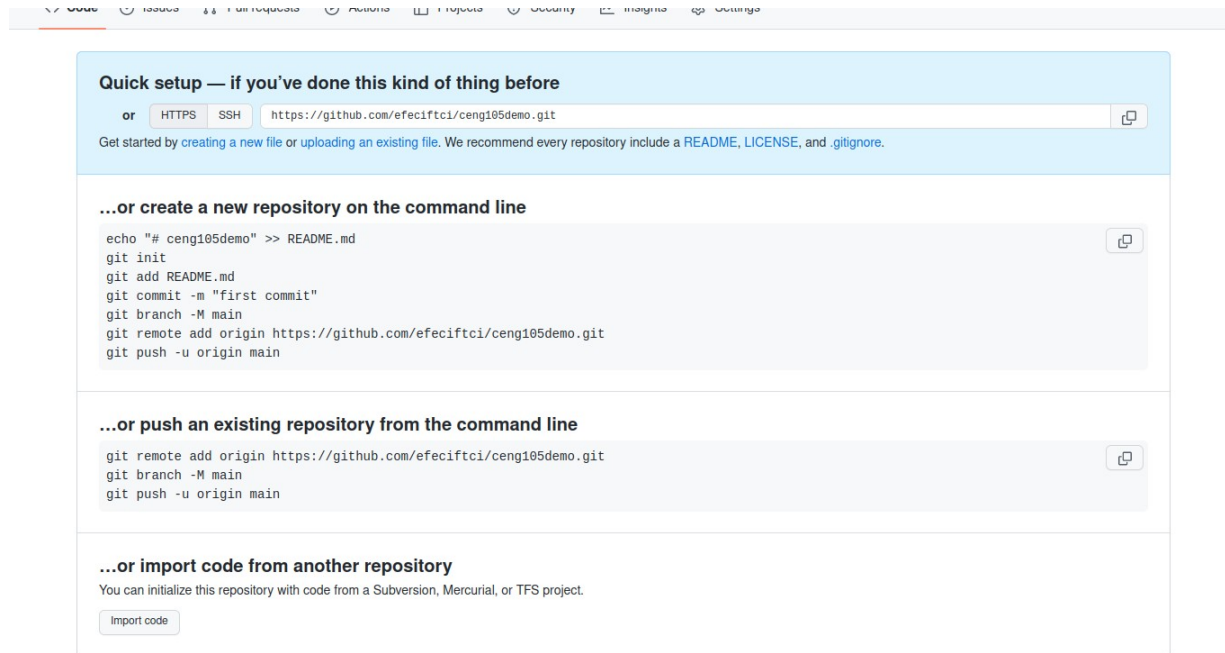
- 13 hours ago  
Secret scanning: Organization admins can now set a link via the API to help...
- 2 days ago  
Launch assignment in GitHub Codespaces
- 3 days ago  
Permissions filtering for organization fine-grained PATs
- 3 days ago  
Fixed bug that allowed OAuth tokens improper access to SAML SSO protect...  
[View changelog](#) →

[Use tools of the trade](#)

**Simplify your development workflow**

**Install a powerful code editor**

# Using GitHub



Quick setup — if you've done this kind of thing before

or  HTTPS  SSH

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# ceng105demo" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/efeciftci/ceng105demo.git
git push -u origin main
```

...or push an existing repository from the command line

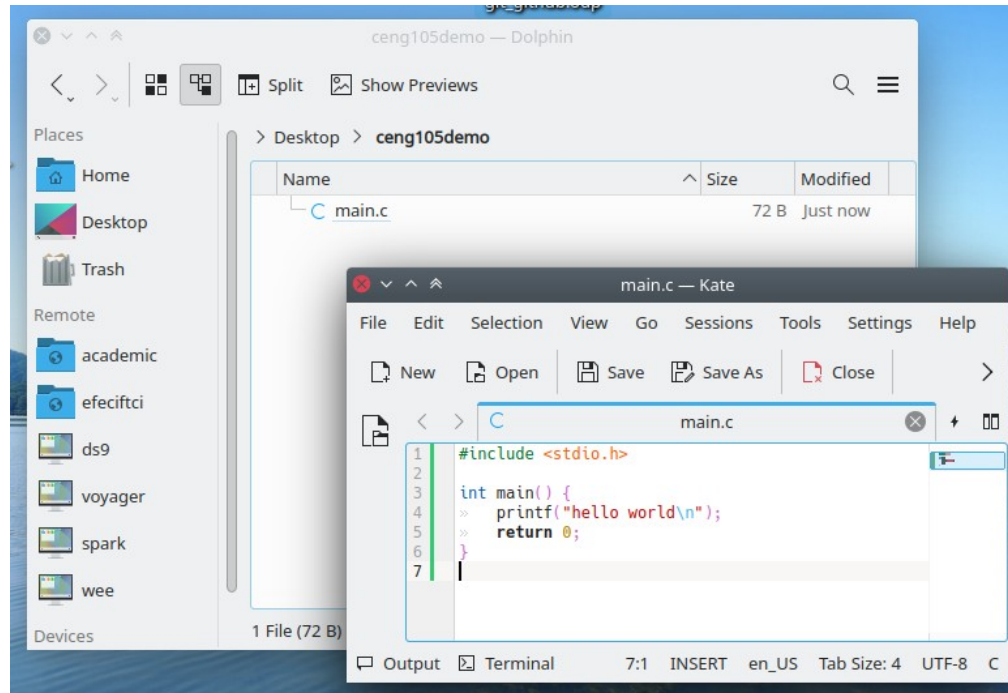
```
git remote add origin https://github.com/efeciftci/ceng105demo.git
git branch -M main
git push -u origin main
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

 **ProTip!** Use the URL for this page when adding GitHub as a remote.

# Using GitHub



# Using GitHub

- If you have created a new file under your project directory or modified existing files:

```
git add
```

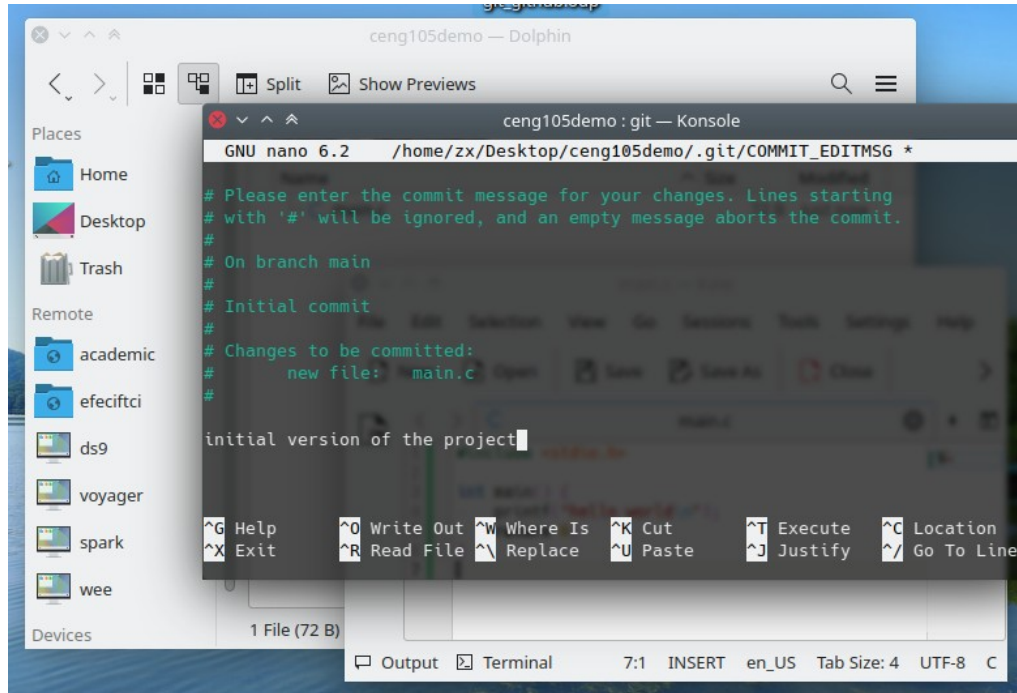
- If you have made changes to your existing project files:

```
git commit
```

- Each commit operation requires you to briefly describe the changes.
- When you are ready to update your remote repository with all locally committed changes:

```
git push
```

# Using GitHub



The image shows a terminal window titled "ceng105demo : git — Konsole" overlaid on a file manager window titled "ceng105demo — Dolphin". The terminal window is running the GNU nano 6.2 editor, editing the file "/home/zx/Desktop/ceng105demo/.git/COMMIT\_EDITMSG". The content of the file is a commit message template. The text in the terminal is as follows:

```
GNU nano 6.2 /home/zx/Desktop/ceng105demo/.git/COMMIT_EDITMSG *
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch main
#
# Initial commit
#
# Changes to be committed:
#   new file:   main.c
#
initial version of the project
```

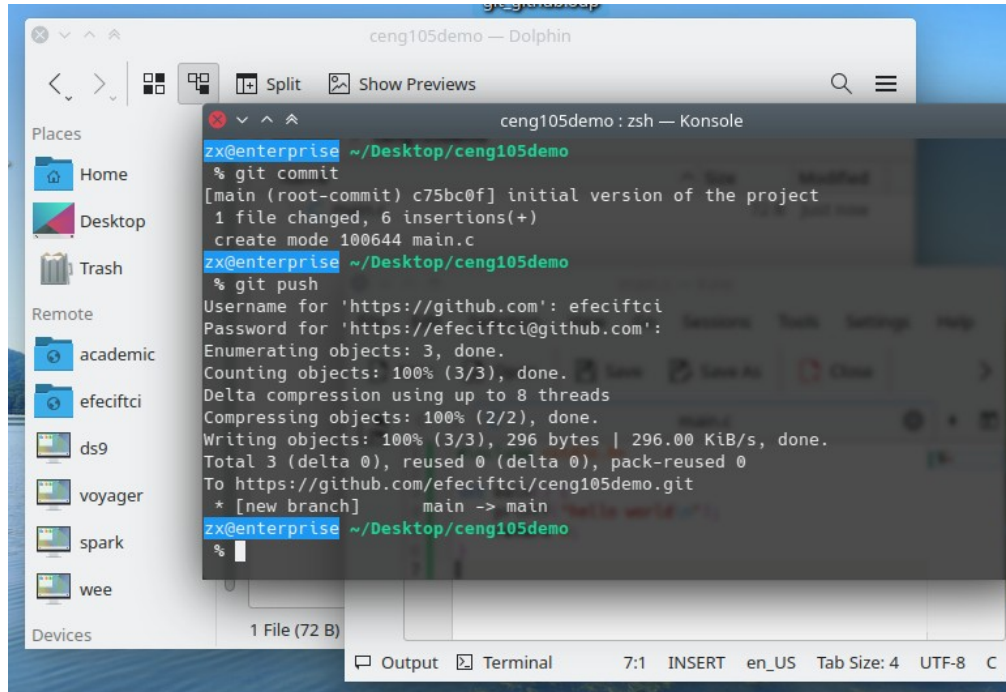
At the bottom of the terminal window, there is a help menu with the following options:

<b>^G</b> Help	<b>^O</b> Write Out	<b>^W</b> Where Is	<b>^K</b> Cut	<b>^T</b> Execute	<b>^C</b> Location
<b>^X</b> Exit	<b>^R</b> Read File	<b>^\<b> Replace</b></b>	<b>^U</b> Paste	<b>^J</b> Justify	<b>^/_</b> Go To Line

The file manager window shows a sidebar with "Places" (Home, Desktop, Trash) and "Remote" (academic, efeciftci, ds9, voyager, spark, wee) sections. The main pane shows "1 File (72 B)". The bottom status bar of the terminal window displays "Output", "Terminal", "7:1", "INSERT", "en\_US", "Tab Size: 4", "UTF-8", and "C".



# Using GitHub



```
zx@enterprise ~/Desktop/ceng105demo
% git commit
[main (root-commit) c75bc0f] initial version of the project
1 file changed, 6 insertions(+)
create mode 100644 main.c
zx@enterprise ~/Desktop/ceng105demo
% git push
Username for 'https://github.com': efeciftci
Password for 'https://efeciftci@github.com':
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 296 bytes | 296.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/efeciftci/ceng105demo.git
 * [new branch]      main -> main
zx@enterprise ~/Desktop/ceng105demo
%
```

# Common Git Commands

start a working area (see also: git help tutorial)

clone	Clone a repository into a new directory
init	Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)

add	Add file contents to the index
mv	Move or rename a file, a directory, or a symlink
restore	Restore working tree files
rm	Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)

bisect	Use binary search to find the commit that introduced a bug
diff	Show changes between commits, commit and working tree, etc
grep	Print lines matching a pattern
log	Show commit logs
show	Show various types of objects
status	Show the working tree status

# Common Git Commands

grow, mark and tweak your common history

branch	List, create, or delete branches
commit	Record changes to the repository
merge	Join two or more development histories together
rebase	Reapply commits on top of another base tip
reset	Reset current HEAD to the specified state
switch	Switch branches
tag	Create, list, delete or verify a tag object signed with GPG

collaborate (see also: `git help workflows`)

fetch	Download objects and refs from another repository
pull	Fetch from and integrate with another repository or a local branch
push	Update remote refs along with associated objects

# Recommended Links

- Git  
<https://git-scm.com/>
- GitHub  
<https://github.com/>
- Tutorials  
<https://githubtraining.github.io/training-manual/>  
  
<https://product.hubspot.com/blog/git-and-github-tutorial-for-beginners>  
  
<https://www.freecodecamp.org/news/learn-the-basics-of-git-in-under-10-minutes-da548267cc91/>

*And many more...*

# Thanks for Listening!

*This presentation has been created on  
Free Operating System [KDE neon](#)  
With  
Free Office Suite [LibreOffice](#).  
"Free as in free speech, not free beer"*

