

Advanced Python Programming Course

Lecture 2.

Git and GitHub

Assoc. Prof. Kovalenko S.M.
Department of software engineering and
intelligent management technologies,
NTU “KhPI”

Google Colaboratory

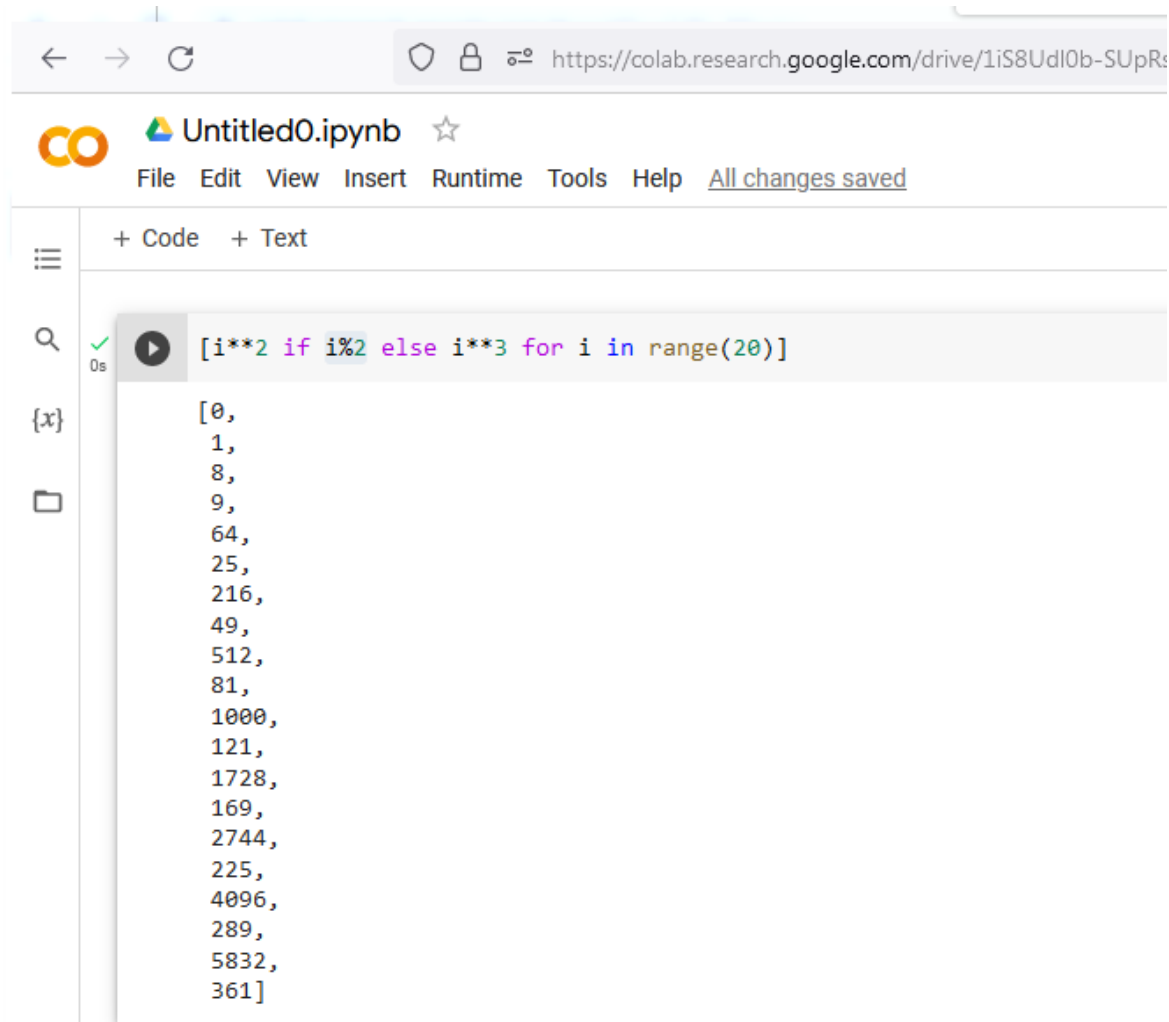
<https://colab.research.google.com/?hl=en>

Colab, or "Colaboratory", allows you to write and execute Python in your browser, with

- Zero configuration required
- Access to GPUs free of charge
- Easy sharing

But you need a Google account

Google Colaboratory



The screenshot displays the Google Colaboratory web interface. The browser address bar shows the URL <https://colab.research.google.com/drive/1iS8Udl0b-SUpRs>. The notebook is titled "Untitled0.ipynb" and has a star icon. The menu bar includes "File", "Edit", "View", "Insert", "Runtime", "Tools", "Help", and a link "All changes saved". The left sidebar contains icons for a menu, search, variables, and files. The main area shows a code cell with a play button icon and a green checkmark, indicating successful execution. The code is a list comprehension: `[i**2 if i%2 else i**3 for i in range(20)]`. The output is a list of 20 numbers: `[0, 1, 8, 9, 64, 25, 216, 49, 512, 81, 1000, 121, 1728, 169, 2744, 225, 4096, 289, 5832, 361]`.

```
[i**2 if i%2 else i**3 for i in range(20)]
```

```
[0,  
 1,  
 8,  
 9,  
 64,  
 25,  
 216,  
 49,  
 512,  
 81,  
 1000,  
 121,  
 1728,  
 169,  
 2744,  
 225,  
 4096,  
 289,  
 5832,  
 361]
```

How to share notebooks

- 1. Push the notebook on GitHub
- 2a. Enter to browser address bar the path Ввести в адресный рядок
https://colab.research.google.com/path_to_your_file_on_GitHub
- 2b. Go to **nbviewer** (<https://nbviewer.jupyter.org/>) and enter GitHub repository in the appropriate field. In the open repository window, select the desired file.

https://github.com/svniko/Advanced_Python/blob/main/Lecture_1.ipynb

What is Git?

- Git is a popular version control system. It was created by Linus Torvalds in 2005
- It is used for:
 - Tracking code changes
 - Tracking who made changes
 - Coding collaboration

What does Git do?

- Manage projects with **Repositories**
- **Clone** a project to work on a local copy
- Control and track changes with **Staging** and **Committing**
- **Branch** and **Merge** to allow for work on different parts and versions of a project
- **Pull** the latest version of the project to a local copy
- **Push** local updates to the main project

What is GitHub?

- git is not the same as GitHub.
- GitHub makes tools that use git.
- GitHub is the largest host of source code in the world, and has been owned by Microsoft since 2018.

Also you can try:

- GitLab
- Bitbucket

Git. Client

- <http://git-scm.com/downloads>



The screenshot shows the Git website's Downloads page. The header features the Git logo and the tagline "--distributed-even-if-your-workflow-isnt". A search bar is located in the top right. The left sidebar contains links for "About", "Documentation", "Downloads" (highlighted in red), "GUI Clients", "Logos", and "Community". The main content area is titled "Downloads" and lists three operating systems: macOS, Windows (circled in red), and Linux/Unix. To the right, a monitor graphic displays the "Latest source Release 2.34.1" with a "Download for Windows" button. A note at the bottom mentions that older releases are available and the source repository is on GitHub.

git --distributed-even-if-your-workflow-isnt

Search entire site...

About
Documentation
Downloads
GUI Clients
Logos
Community

Downloads

macOS **Windows** Linux/Unix

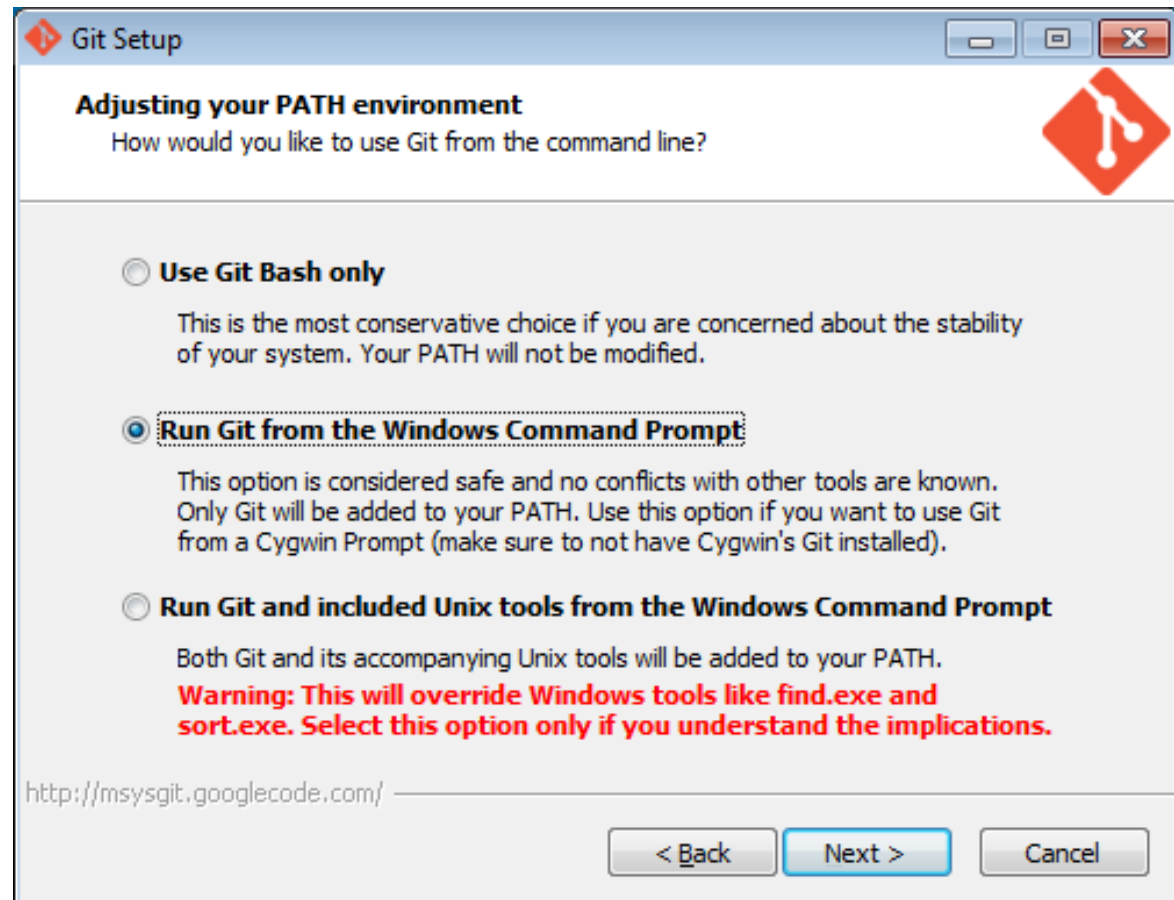
Older releases are available and the Git source repository is on GitHub.

Latest source Release
2.34.1
Release Notes (2021-11-24)
Download for Windows

The entire **Pro Git book** written by Scott Chacon and

Git. Client

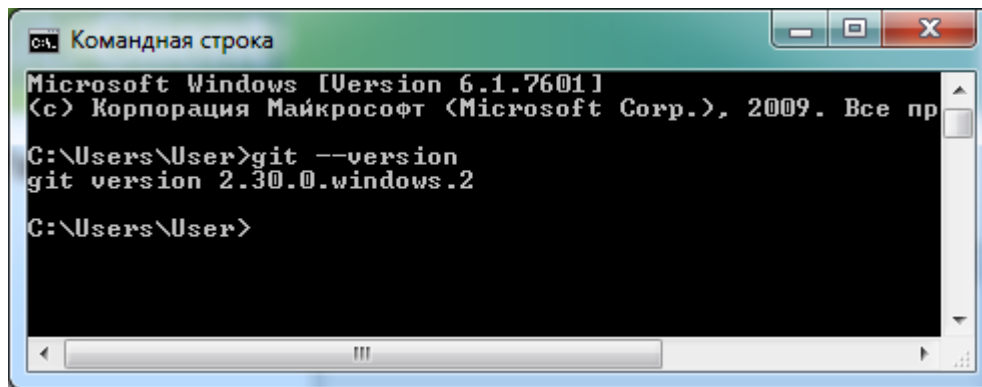
- Path Dialog



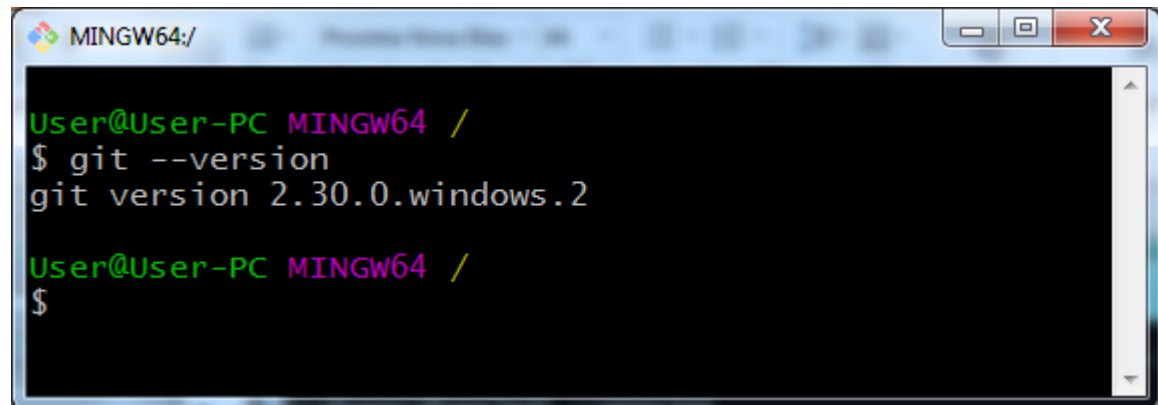
Git. Client

- Check if git is installed

```
git --version
```



```
Командная строка
Microsoft Windows [Version 6.1.7601]
(c) Корпорация Майкрософт (Microsoft Corp.), 2009. Все пр
C:\Users\User>git --version
git version 2.30.0.windows.2
C:\Users\User>
```



```
MINGW64:/
User@User-PC MINGW64 /
$ git --version
git version 2.30.0.windows.2
User@User-PC MINGW64 /
$
```

Git. Create a local repo

- Go to the directory of your project and initialize a local Git repository by

```
git init
```

- Register your name and e-mail in git

```
git config --global user.name <desired name  
to be seen as author of code>
```

```
git config --global user.name spiderman
```

```
git config --global user.email <desired e-mail  
to be seen as authors of code>
```

```
git config --global user.email hohoho@gmail.com
```

```
git config --list
```

.gitignore

In the directory of your project create a file `.gitignore`
(the dot before the name is compulsory)

A `.gitignore` file specifies intentionally untracked files that Git should ignore

`.gitignore` generator

<https://www.toptal.com/developers/gitignore>

Some Linux command

- `cd` - change directory
- `ls` - display a list of files and sub-directories
- `ls -a` - list all files including hidden file starting with '.'
- `touch <file name>` - create a file without any content
- `cat <fileName>` - display the content of a file
- `vim <file name>` - open file for editing in Vim. If file does not exist, it will be created.

- **Some Vim commands:**

- ❖ `<I>` (I key) - insert mode

- ❖ `<Esc>` - command mode

- ❖ `:wq<Enter>` - save file and quit

Git. Status of Files

- See the state of our repo

```
git status
```

```
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .gitignore
        Lecture_1.ipynb
        Out.txt

nothing added to commit but untracked files present (use "git add" to track)
```

Add files to the git staging area

- Add a file

```
git add <file_name>
```

- Add all

```
git add .
```

- Check status

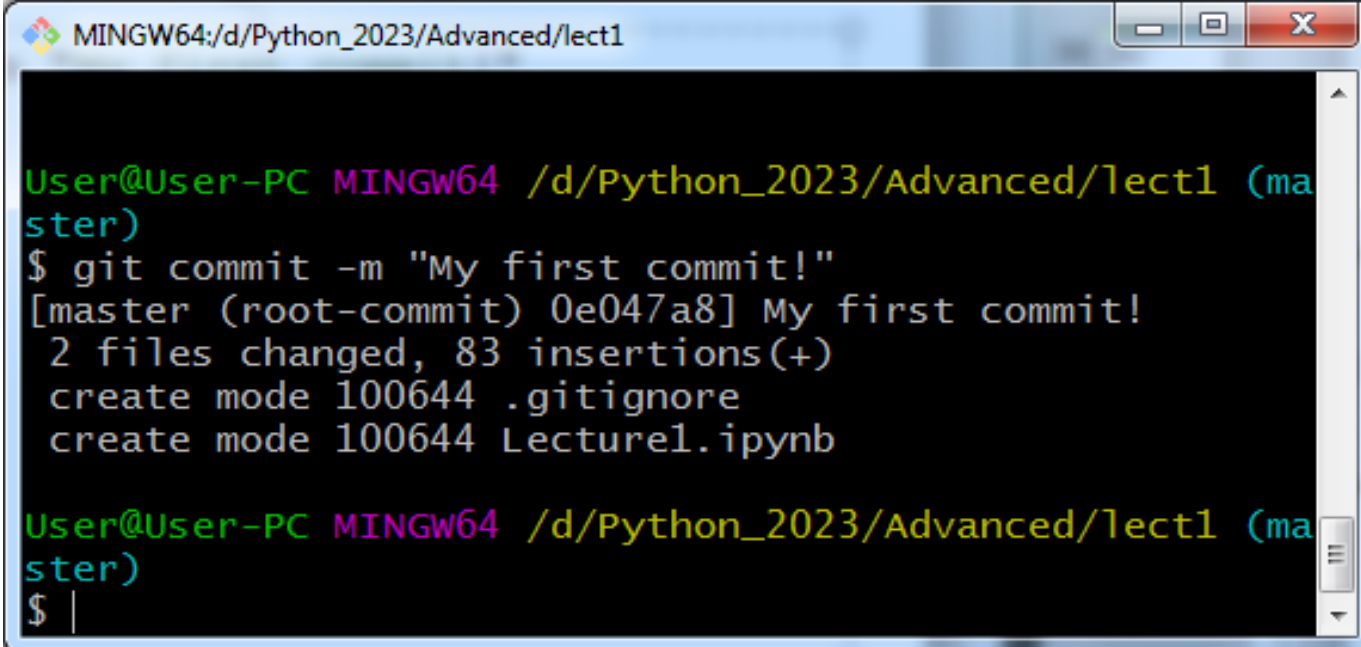
```
git status
```

- Check branch

```
git branch
```

Git. Commit changes

```
git commit -m "My first commit!"
```

A screenshot of a terminal window titled "MINGW64:/d/Python_2023/Advanced/lect1". The terminal shows the execution of the command "git commit -m 'My first commit!'" and its output. The output indicates that the commit was successful on the master branch, with a hash of 0e047a8. It also shows that 2 files were changed, with 83 insertions, and that two new files, ".gitignore" and "Lecture1.ipynb", were created with mode 100644. The prompt "\$" is visible at the bottom, indicating the command prompt is ready for the next input.

```
MINGW64:/d/Python_2023/Advanced/lect1

User@User-PC MINGW64 /d/Python_2023/Advanced/lect1 (ma
ster)
$ git commit -m "My first commit!"
[master (root-commit) 0e047a8] My first commit!
 2 files changed, 83 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 Lecture1.ipynb

User@User-PC MINGW64 /d/Python_2023/Advanced/lect1 (ma
ster)
$ |
```


GitHub. Create a new repo

- Go to <https://github.com/>

The screenshot shows the GitHub 'Create a new repository' page. The page is divided into two main sections: a left sidebar and a main content area.

Left Sidebar:

- Top: Navigation links (back, forward, refresh) and the GitHub logo.
- Search bar: "Search or jump to..."
- Section: "Top Repositories" with a "Find a repository..." input field.
- List of repositories: svniko/Advanced_Python, svniko/AutoParts, svniko/test, svniko/django.
- Green "New" button with a red arrow pointing to it.

Main Content Area:

- Section: "Create a new repository".
- Text: "A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)"
- Form fields:
 - Owner *: svniko (dropdown)
 - Repository name *: Example (text input with a green checkmark and a red arrow pointing to it)
 - Description (optional): (empty text input)
- Visibility options:
 - ☒ Public: Anyone on the internet can see this repository. You choose who can commit.
 - ☐ Private: You choose who can see and commit to this repository.
- Section: "Choose a license".
- Text: "A license tells others what they can and can't do with your code. [Learn more.](#)"
- License dropdown: License: None
- Information: "You are creating a public repository in your personal account."
- Green "Create repository" button with a red arrow pointing to it.

GitHub. Create a new repo

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



Set up in Desktop

or

HTTPS

SSH

<https://github.com/svniko/Example.git>

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

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

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

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

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

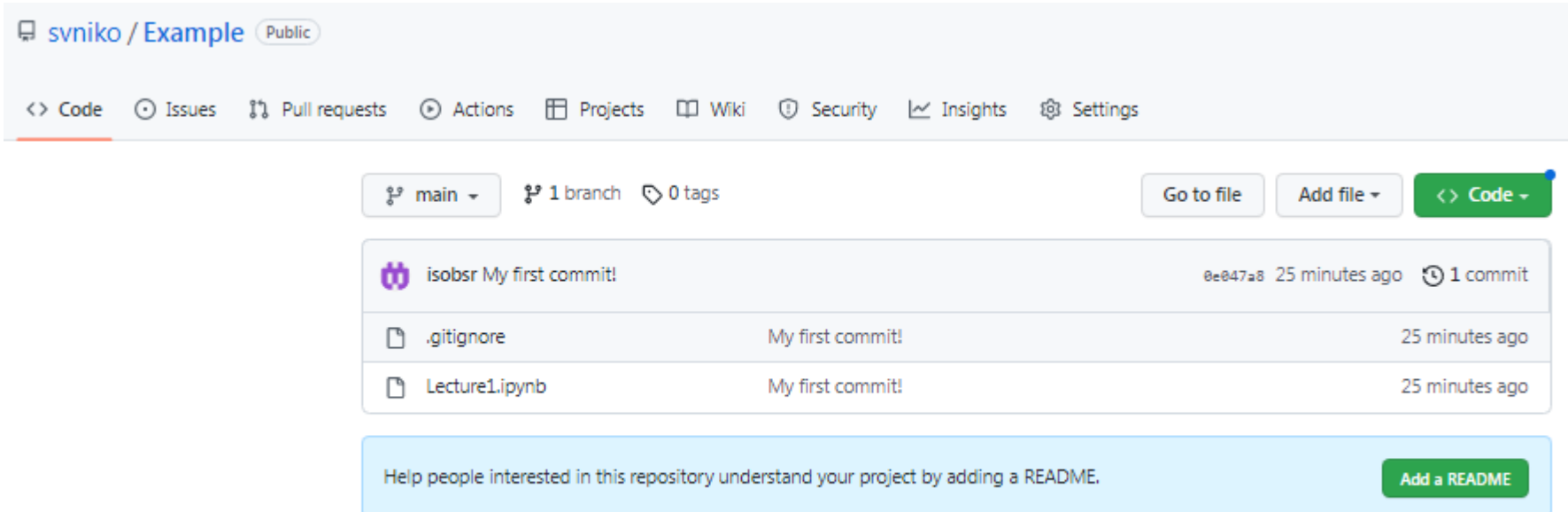
...or import code from another repository

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

[Import code](#)

Push local repo on GitHub

- `git remote add origin <path to your GitHub repo>`
- `git branch -M main`
- `git push -u origin main`



The screenshot shows the GitHub interface for a repository named 'svniko / Example' which is public. The repository has a 'main' branch with 1 branch and 0 tags. The commit history shows a single commit by 'isobsr' titled 'My first commit!' with the hash '0e047a8' made 25 minutes ago. The commit includes two files: '.gitignore' and 'Lecture1.ipynb', both added in the same commit. A prompt at the bottom encourages adding a README file.

svniko / Example Public

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main 1 branch 0 tags Go to file Add file <> Code

isobsr	My first commit!	0e047a8	25 minutes ago	1 commit
.gitignore	My first commit!		25 minutes ago	
Lecture1.ipynb	My first commit!		25 minutes ago	

Help people interested in this repository understand your project by adding a README. Add a README

Pull changes

- Let's create a README.md

svniko / Example Public

<> Code Issues Pull requests Actions Projects Wiki ...

Example / README.md in main

<> Edit file Preview

```
1 # Example
2 ## Created 07.03
3
```

Commit changes

Update README.md





Add an optional extended description...


☒ Commit directly to the main branch.

☐ Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)

Commit changes

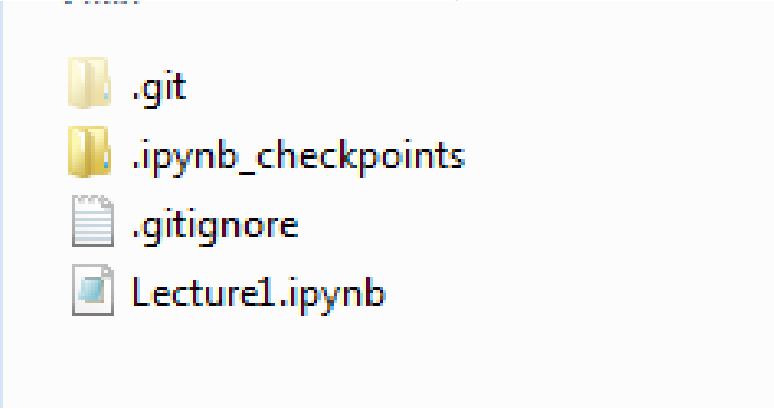
GitHub repo vs local repo

 svniko Update README.md	270405b now ⌚ 3 commits
 .gitignore	My first commit! 34 minutes ago
 Lecture1.ipynb	My first commit! 34 minutes ago
 README.md	Update README.md now

☰ README.md 

Example

Created 07.03

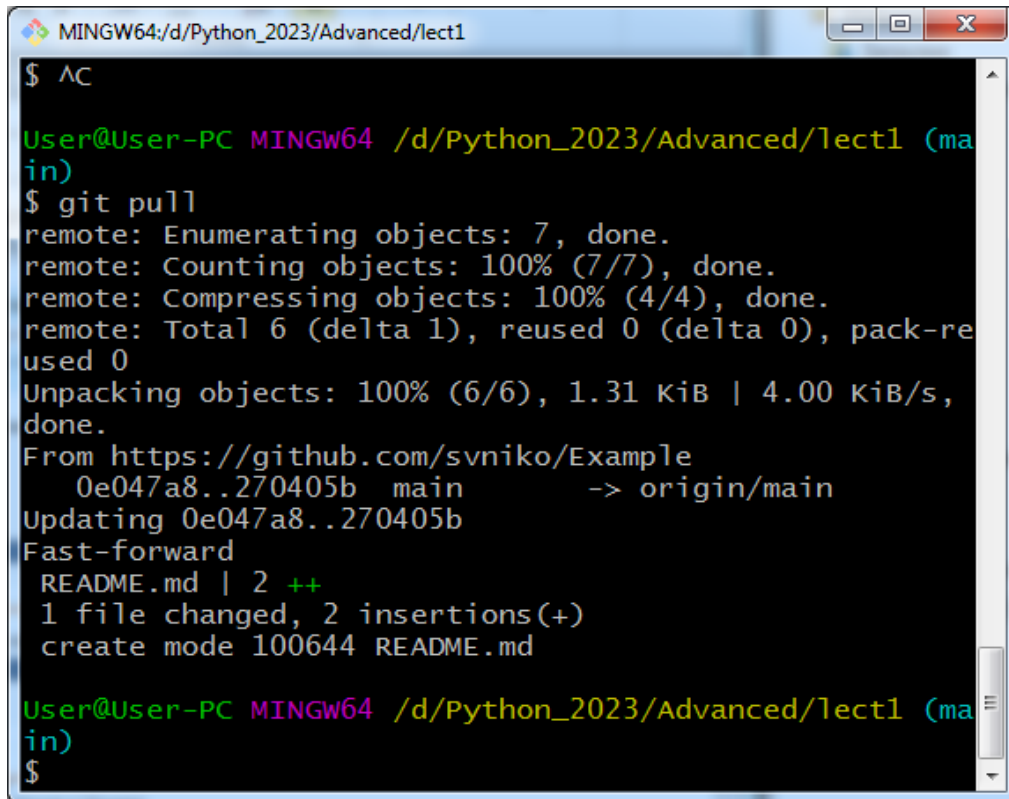


- .git
- .ipynb_checkpoints
- .gitignore
- Lecture1.ipynb

Get data

Gets (fetch) data and **automatically** to merge with the code (master)

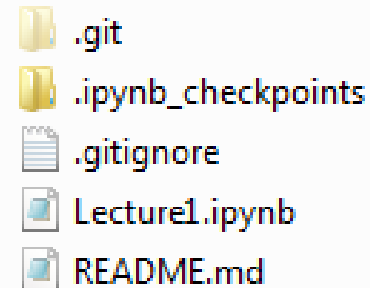
```
git pull
```

A screenshot of a Windows terminal window titled 'MINGW64:/d/Python_2023/Advanced/lect1'. The terminal shows the execution of 'git pull' command. The output indicates that 7 objects were enumerated, 100% of objects were counted, and 100% of objects were compressed. The total size of the update is 6 objects (delta 1), with 0 objects reused. The update is performed via a fast-forward merge from the origin/main branch. The output shows that 1 file (README.md) was changed, with 2 insertions. The terminal text is as follows:

```
$ ^C

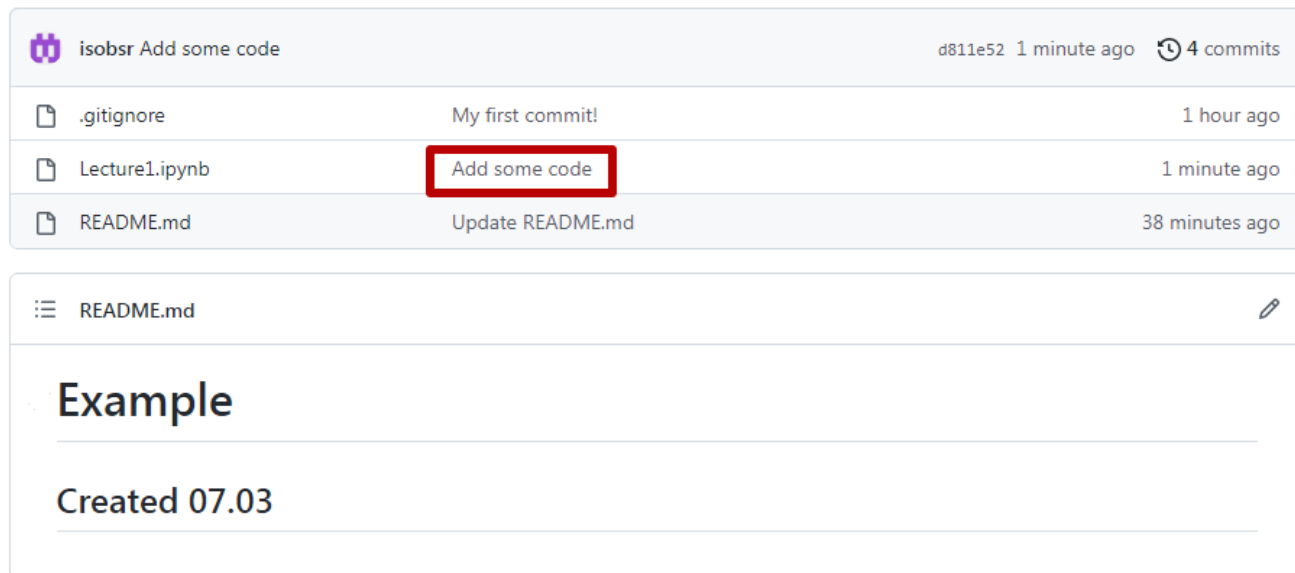
User@User-PC MINGW64 /d/Python_2023/Advanced/lect1 (main)
$ git pull
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 6 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (6/6), 1.31 KiB | 4.00 KiB/s, done.
From https://github.com/svniko/Example
   0e047a8..270405b  main       -> origin/main
Updating 0e047a8..270405b
Fast-forward
 README.md | 2 ++
 1 file changed, 2 insertions(+)
 create mode 100644 README.md

User@User-PC MINGW64 /d/Python_2023/Advanced/lect1 (main)
$
```



After any file modification

1. `git status`
2. `git add .`
3. `git commit -m "<message>"`
4. `git push`



The screenshot displays a GitHub repository interface. At the top, the repository name is 'isobsr Add some code', with the commit hash 'd811e52', the time '1 minute ago', and '4 commits'. Below this is a table of recent commits:

File	Commit Message	Time
.gitignore	My first commit!	1 hour ago
Lecture1.ipynb	Add some code	1 minute ago
README.md	Update README.md	38 minutes ago

The 'Add some code' commit message for 'Lecture1.ipynb' is highlighted with a red box. Below the commit history, the 'README.md' file is shown in a viewer. The file content includes a heading 'Example' and a line 'Created 07.03'.