

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний аерокосмічний університет ім. М. Є. Жуковського

«Харківський авіаційний інститут»

Факультет радіотехніки, комп'ютерних систем і інфокомунікацій

Кафедра інформаційно-комунікаційних технологій ім. О. О. Зеленського

Практична робота №1

з дисципліни «Технології неперервної інтеграції і розгортання ІС»

Виконав: студент 4 курсу групи № 548
напряму підготовки (спеціальності)
126 Штучний інтелект ті інформаційні
системи

(шифр і назва напряму підготовки (спеціальності))

Косянчук Максим Олександрович
(прізвище й ініціали студента)

Прийняв:
(посада, науковий ступінь, прізвище й ініціали)

Національна шкала: _____

Кількість балів: _____

Оцінка: ECTS _____

Завдання на практичне заняття №1

Основи роботи з Git

Мета: Ознайомитися з основами роботи з системою контролю версій Git, навчитися створювати репозиторій, комітити зміни у файлі, працювати з віддаленими репозиторіями, співпрацювати з іншими розробниками.

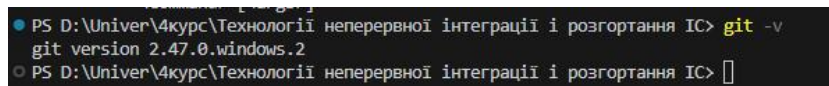
Зміст заняття

1. Встановлення та налаштування Git.
2. Основні команди роботи з Git.
3. Робота з віддаленими гілками.
4. Вирішення конфліктів.

Для кожного завдання (заданого цифрою) у звіті необхідно навести скрін-шот терміналу (або його частини, але обов'язково, щоб було видно рядок запрошення до введення, наприклад, "mivanov@pc: ~\$") з Вашою командою, результатом виконання команди, вмістом файлу (за необхідністю).

Завдання:

1. Встановіть Git на ваш комп'ютер.
 - Завантажте та встановіть останню версію Git з Офіційного сайту - <https://git-scm.com/>.
 - Відкрийте командний рядок (термінал, консоль, Command Prompt).
 - Введіть `git --version`, щоб перевірити, що Git встановлений.



```
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git -v
git version 2.47.0.windows.2
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC>
```

2. Налаштуйте Git.
 - Встановіть своє ім'я та електронну пошту, які будуть використовуватись для підписання комітів
 - Перевірте поточні налаштування:

```

PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git config --list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=openssl
http.sslcainfo=C:/Program Files/Git/mingw64/etc/ssl/certs/ca-bundle.crt
core.autocrlf=true
core.fscache=true
core.symlinks=false
pull.rebase=false
credential.helper=manager
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
core.editor="D:\Programs\Microsoft VS Code\bin\code" --wait
safe.directory=D:/Univer/ML
user.name=Maks
user.email=m.o.kosianchuk@strudent.khai
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC>

```

3. Створіть локальний репозиторій Git.

- Створіть нову папку для проекту на вашому комп'ютері.

```

PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git init
Initialized empty Git repository in D:/Univer/4курс/Технології неперервної інтеграції і розгортання IC/.git/
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC>

```

4. Додайте файли та зробіть коміт.

- Створіть файл у каталозі вашого проекту, наприклад, README.md.
- Додайте зміни у файлі до індексу
- Зафіксуйте зміни у репозиторії

```

PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git add README.md
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git commit -m "Added README.md"
[master (root-commit) 96d4f53] Added README.md
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 README.md
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC>

```

5. Робота з віддаленим репозиторієм.

- Зареєструйте акаунт на GitHub (<https://github.com/>), якщо у вас його ще немає.
- Створіть новий віддалений репозиторій на GitHub.
- Додайте віддалений репозиторій у ваш локальний проект:
- Завантажте коміти з локального репозиторію до віддаленого репозиторію на Github:

```

PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git push -u origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 221 bytes | 221.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/MaksimKosyanchuk/integration.git
* [new branch] master -> master
branch 'master' set up to track 'origin/master'.
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC>

```

6. Співпраця з іншими розробниками.

- Ознайомтеся з такими командами Git, як:
 - `git branch` - для створення гілок для роботи над окремими завданнями.
 - `git checkout` - для перемикання між гілками.
 - `git merge` - для зливання гілок та інтеграції змін із різних гілок.
 - `git pull` - для отримання останніх змін з віддаленого репозиторію та додавання їх до вашого локального репозиторію.

7. Робота з гілками.

- Створіть нову гілку для роботи над однією з функцій вашого проекту:
- Робіть зміни у файлах, додавайте їх до індексу та комітіть, аналогічно до попередніх кроків. Зверніть увагу, що ці зміни будуть додані тільки до поточної гілки.
- Поверніться до основної гілки (`main` або `master`, залежно від версії `git`) та злийте дві гілки разом:
- Тепер зміни з вашої гілки внесені до основної гілки. Ви можете видалити свою робочу гілку, якщо вона вам більше не потрібна:

```
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git checkout -b 7.1
Switched to a new branch '7.1'
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> echo > test.txt

Командлет Write-Output в конвейєре команд в позиції 1
Укажіть значення для наступних параметрів:
InputObject[0]:
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git add test.txt
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git commit -m "Added test.txt"
[7.1 f31a53b] Added test.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 test.txt
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git status
On branch 7.1
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    Practice1/

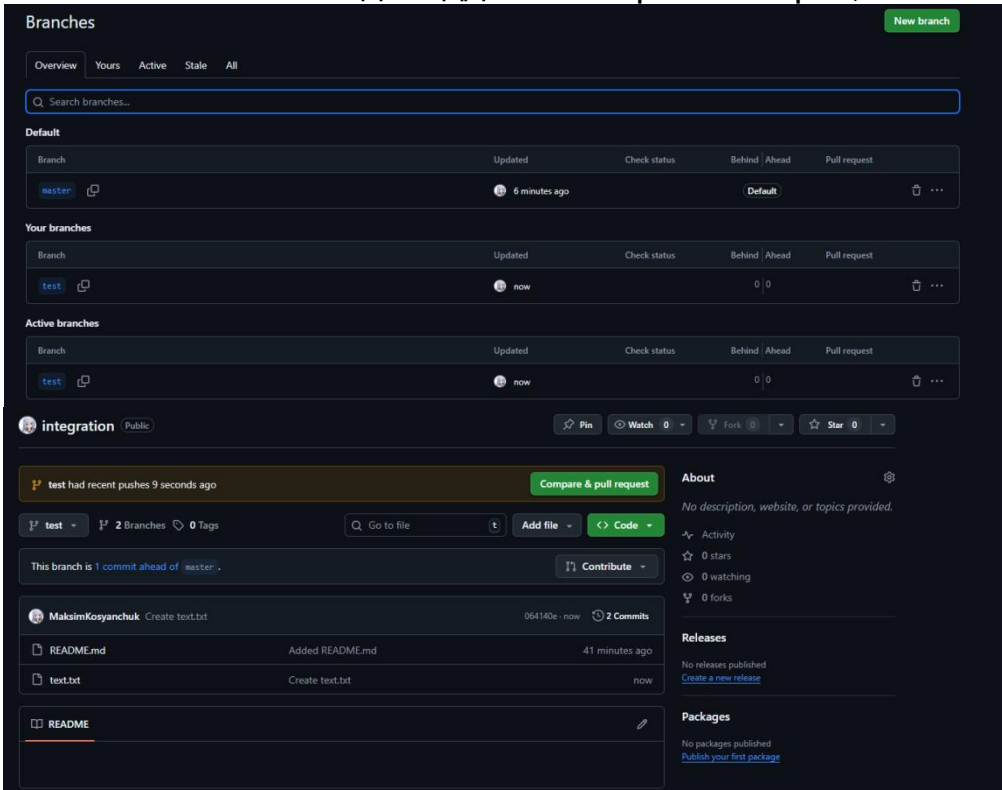
nothing added to commit but untracked files present (use "git add" to track)
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git merge 7.1
Updating 96d4f53..f31a53b
Fast-forward
 test.txt | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 test.txt
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git log
commit f31a53ba89aa4d4f8b413a1494135b7a7e4a8d38 (HEAD -> master, 7.1)
Author: Maks <m.o.kosianchuk@student.khai>
Date:   Mon Sep 22 17:14:49 2025 +0300

    Added test.txt

commit 96d4f5375fac6583f07e53671b48705cb0be5684 (origin/master)
Author: Maks <m.o.kosianchuk@student.khai>
Date:   Mon Sep 22 16:38:29 2025 +0300
```

8. Робота з віддаленими гілками.

- Коли ви працюєте з гілками, неминуче будуть віддалені гілки, які створили ваші колеги.
- Створіть нову гілку в GitHub за допомогою UI.
- Завантажте на GitHub файл на ваш вибір, наприклад test3.txt
- Щоб отримати список всіх віддалених гілок, виконайте:
- Щоб створити локальну копію віддаленої гілки, виконайте:
- Тепер ви можете працювати з цією гілкою локально. Коли ви будете готові внести свої зміни до віддаленого репозиторію, виконайте:



```
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git fetch
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (3/3), 934 bytes | 93.00 KiB/s, done.
From https://github.com/MaksimKosyanchuk/integration
* [new branch] test -> origin/test
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git branch -f 7.1
* master
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git branch -r origin/master origin/test
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git checkout -b test/test
Switched to a new branch 'test/test'
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> 
```

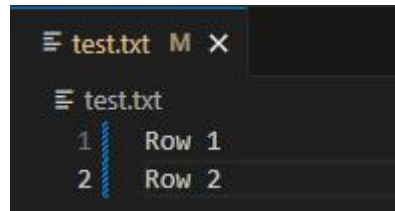
9. Вирішення конфліктів.

Іноді в процесі співпраці виникають конфлікти, коли дві особи вносять зміни до одного блоку коду. Щоб вирішити конфлікт, треба відкрити файл з конфліктом, знайти спірні моменти, зазначені

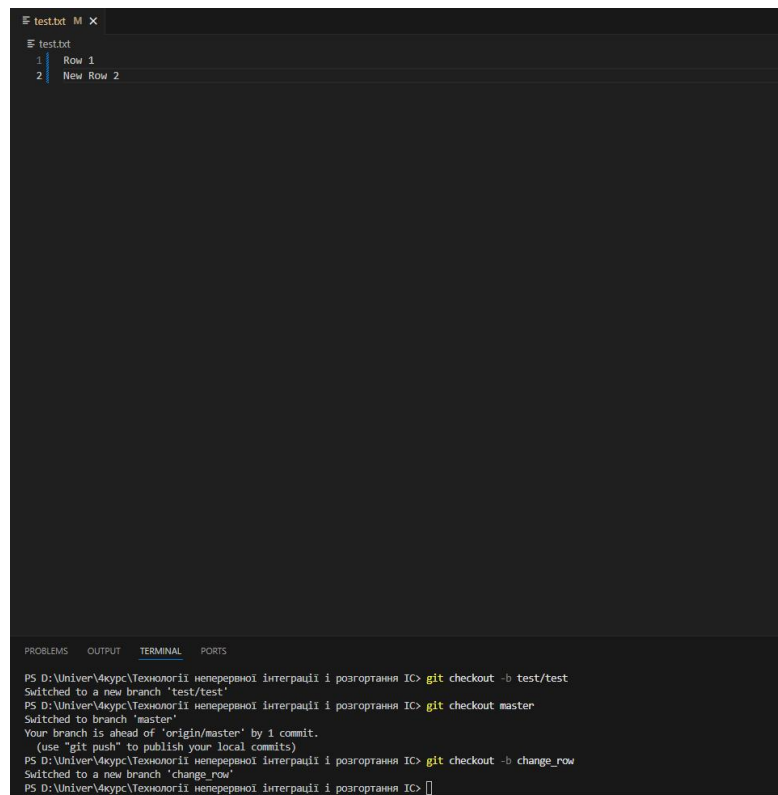
обидвома особами, вирішити, який варіант змін коректний, видалити зайве і зробити коміт.

Далі потрібно зберегти зміни, додати їх до індексу та зробити коміт, аналогічно до попередніх кроків. Коміт повинен містити повідомлення про виправлення конфлікту.

Розглянемо простий практичний приклад створення й вирішення конфлікту в Git.



- Створіть й переключіться на нову гілку feature_branch:
- Зробіть зміни в файлі example.txt. Змініть другий рядок на "Змінений рядок 2".



- Додайте й збережіть зміни у файлі.

```
• PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git add test.txt
• PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git commit -m "Change test.txt"
[change_row c6a4551] Change test.txt
1 file changed, 2 insertions(+)
• PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC>
```

- Поверніться на головну гілку.
- Зробіть іншу зміну в файлі example.txt на головній гілці, змінивши другий рядок на "Новий рядок 2".

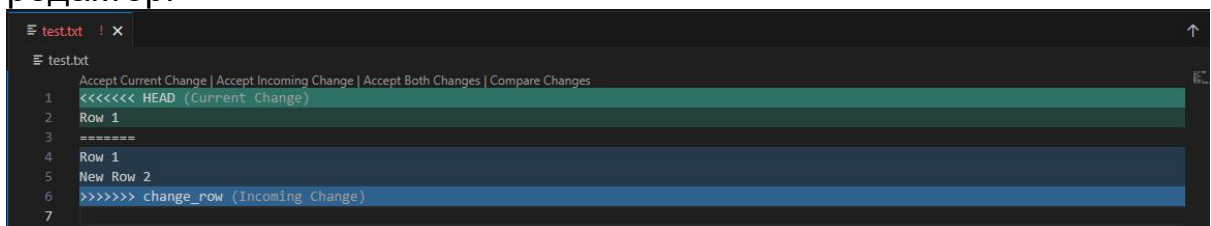
Тепер маємо конфлікт у файлі example.txt, коли спробуємо злити гілку feature_branch.

- Спробуйте злити гілки.

```
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git merge change_row
Auto-merging test.txt
CONFLICT (content): Merge conflict in test.txt
Automatic merge failed; fix conflicts and then commit the result.
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> 
```

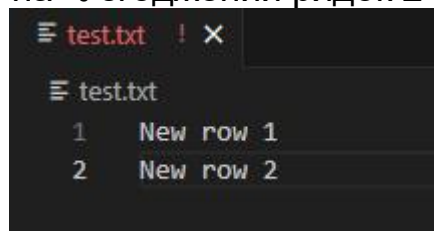
Це повідомлення вказує на конфлікт у файлі "example.txt".

- Відкрийте файл example.txt у текстовому редакторі



```
test.txt
Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
<<<<<< HEAD (Current Change)
1 Row 1
2
3 =====
4 Row 1
5 New Row 2
6 >>>>>> change_row (Incoming Change)
7
```

- Вирішіть конфлікт, домовившись зі своїми колегами, змінивши його на "Узгоджений рядок 2".



```
test.txt
test.txt
1 New row 1
2 New row 2
```

- Збережіть файл та зробіть коміт.

```
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git add test.txt
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> git commit -m "Fixed"
[master 5c73ae8] Fixed
PS D:\Univer\4курс\Технології неперервної інтеграції і розгортання IC> 
```

10. Розуміння Git Workflow.

- Для ефективної роботи з командою дотримуйтесь визначеної стратегії роботи з Git, такими як [GitHub Flow](#) або [Gitflow Workflow](#).
- Ознайомтеся з поняттями pull request, code review для поліпшення роботи вашої команди та якості коду.
- Вивчення основ роботи з Git є дуже важливим для комфортної роботи в команді програмістів. В подальшому, вивчайте додаткові команди та інструменти відповідно до вашої конкретної роботи та завдань.