

**Міністерство освіти і науки України
Національний університет «Львівська політехніка»**



ЗВІТ

Про виконання лабораторної роботи № 0

**На тему: “Налаштування *Git*”
з дисципліни «Моделювання та аналіз ПЗ»**

Лектор:

доцент кафедри ПЗ
Сердюк П.В.

Виконав:

студент. групи ПЗ-22
Місяйло О.О.

Прийняв:

викладач ПЗ
Микуляк А.В.

«___» _____ 2023 р.

Σ = _____

Львів – 2023

Тема: Налаштування Git.

Мета: Освоїти базові навички роботи з Git.

Теоретичні відомості

Git – це розподілена система контролю версій, яка дозволяє відстежувати історію розробки ПЗ і спільно працювати над складними проектами з будь-якої точки світу.

Переваги Git над іншими системами:

Головна перевага Git – в тому, що він дуже швидкий і прозорий. Він зручний для нелінійної розробки і ефективний як для невеликих проектів, так і для великих систем з тисячами учасників.

На відміну від Perforce, CVS та інших, Git зберігає знімки репозиторіїв, а не списки змін в файлах, і внаслідок цього працює набагато швидше.

Git – розподілена система. Якщо сервер з віддаленим репозиторієм вийде з ладу, можна відновити код з локальної копії. Якщо локальна копія постраждає, можна завантажити код із сервера за кілька хвилин.

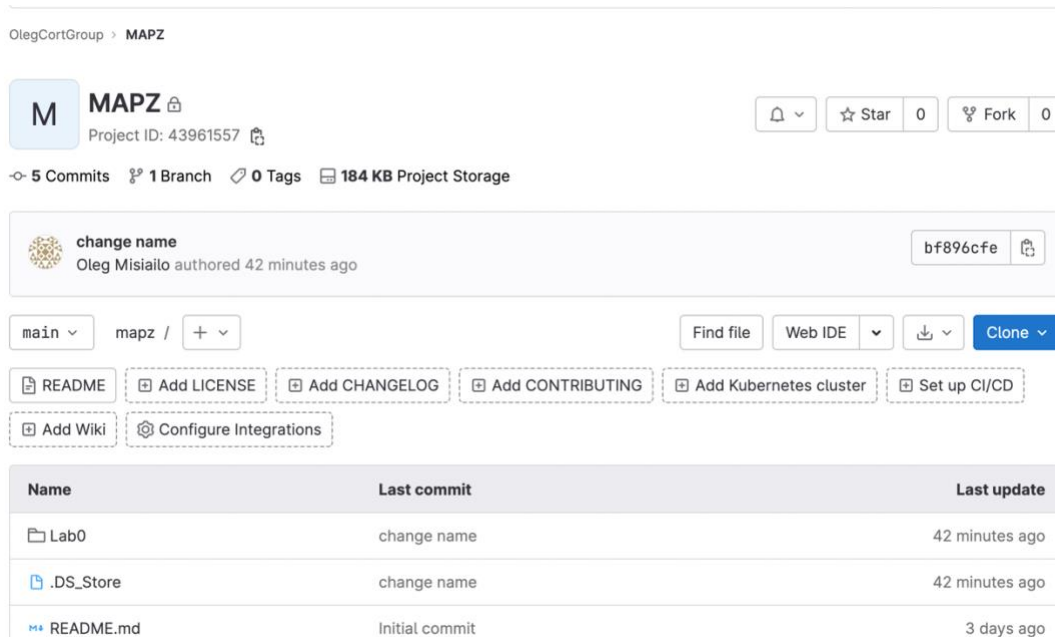
Завдання

1. Переглянути відео
<https://www.youtube.com/watch?v=IO4IGv9TLnQ&list=PLCSO5njdNs1-jYfAkuteDSm1eivMqs6Bw>
2. **Створити аккаунт створювати на GitLab (тільки GitLab - не GitHub, і не жоден інший)**
3. **Створити репозиторій назвати MAPZ (тільки така назва)**
4. Створити папку для лабораторних на комп'ютері. Запустити там git clone для репозиторію
git clone [url репозиторію]
5. Створити структура папок локально на комп'ютері- верхня MAPZ, всередині Lab0, Lab1 ,.....
6. Надати доступ до вашого репозиторію **pavlo_serdyuk**, з правами доступу до коду (низькі права не дозволяють переглядати код)
7. Додати посилання на ваш репозиторійу spreadsheet файл, доступ до кого має забезпечити староста групи
8. Проробити усі дії на сайті <https://learngitbranching.js.org/> до 5 рівня (Advanced не потрібно) і повторити їх у себе локально на комп'ютері
9. Усі дії робити у папці Lab0. Для комітів використовувати текстові файли, щоб зробити новий коміт, достатньо змінити лінійку тексту всередині файлу.
10. Для кожного завдання використовуйте новий текстовий файл.

11. Звіти повинні містити скріншоти з операціями - для цього можна використати лог, що відобразить історію, або будь-яку візуальну надбудову Git - GitKraken, SourceTree, TortoiseGit.
12. Звіти завантажити на ВНС.

Хід виконання

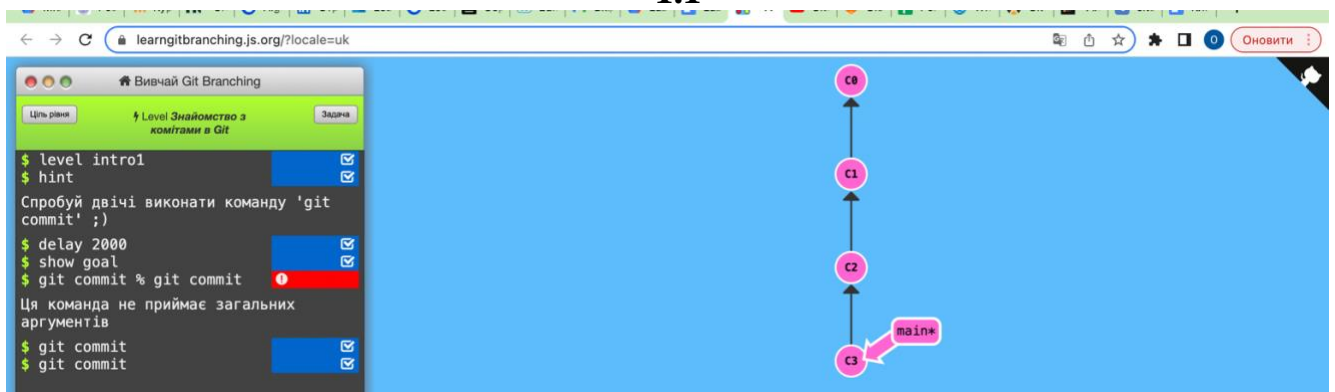
1. На початку виконання я створив акаунт на GitLab та зробив там репозиторій з назвою MAPZ



До папки MAPZ я додав папку lab0.

2. Після цього я проходжу по рівнях та пунктах з сайту [learngitbranching](https://learngitbranching.js.org/) та паралельно виконую ті ж операції на комп'ютері.

1.1



```

[olegmisialo@MacBook-Air-Oleg mapz % git log
commit 520f069f5d6c1672b17c0457225055a840fef7b1 (HEAD -> main, origin/main, origin/HEAD)
Author: OlegCort <olegmisialo@gmail.com>
Date: Sun Mar 5 17:20:13 2023 +0200

    1:1 com2

commit a325a7d56cf6b0bc7cc6d5c005379236aaac0d0e
Author: OlegCort <olegmisialo@gmail.com>
Date: Sun Mar 5 17:19:41 2023 +0200

    1:1 com1

```

1.2

```

$ level intro2
$ hint
Створи нову гілку за допомогою "git
branch [ім'я]" й перейди на неї за
допомогою "git checkout [ім'я]"
$ delay 2000
$ show goal
$ git branch bugFix
$ git checkout bugFix

```

```

graph BT
    c1((c1)) --> c0((c0))
    subgraph Callout [ ]
        direction TB
        main[main]
        bugFix[bugFix*]
    end
    Callout -.-> c1

```

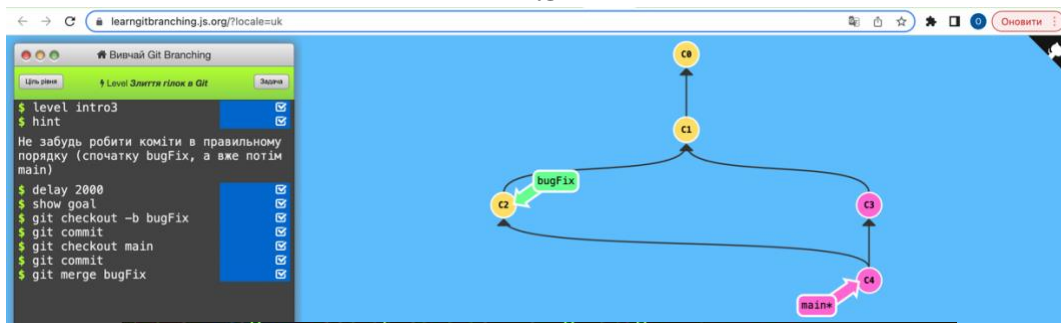
```

[olegmisialo@MacBook-Air-Oleg mapz % git log
commit daf806435a62ab82dc16c13bd327f91fb5c784da (HEAD -> bugFix)
Author: OlegCort <olegmisialo@gmail.com>
Date: Sun Mar 5 17:32:30 2023 +0200

    1:2

```

1.3



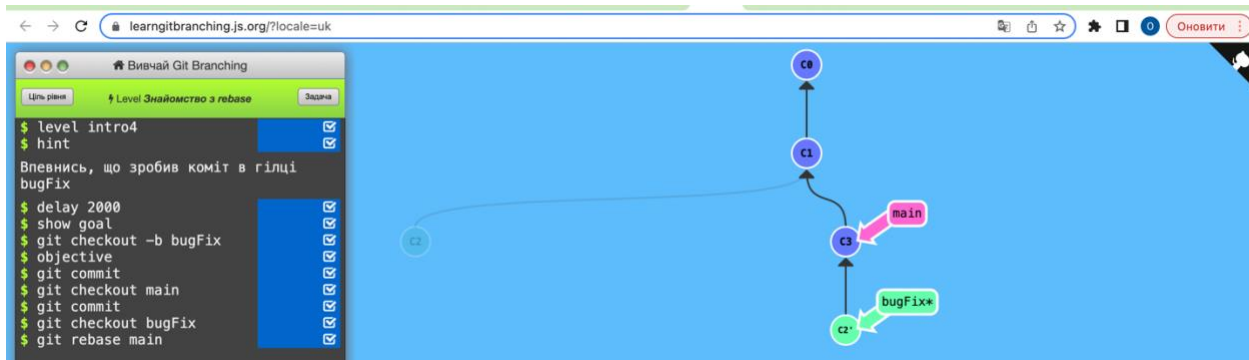
```

olegmisialo@MacBook-Air-Oleg mapz % git checkout bugFix
Already on 'bugFix'
Your branch is up to date with 'origin/bugFix'.
olegmisialo@MacBook-Air-Oleg mapz % touch Lab0/song1_3.txt
olegmisialo@MacBook-Air-Oleg mapz % git add .
olegmisialo@MacBook-Air-Oleg mapz % git commit -m "added 2 lines"
[bugFix 3ae05ba] added 2 lines
1 file changed, 4 insertions(+)
create mode 100644 Lab0/song1_3.txt
olegmisialo@MacBook-Air-Oleg mapz % git main
git: 'main' is not a git command. See 'git --help'.

The most similar command is
    mailinfo
olegmisialo@MacBook-Air-Oleg mapz % git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
olegmisialo@MacBook-Air-Oleg mapz % git merge bugFix

```

1.4



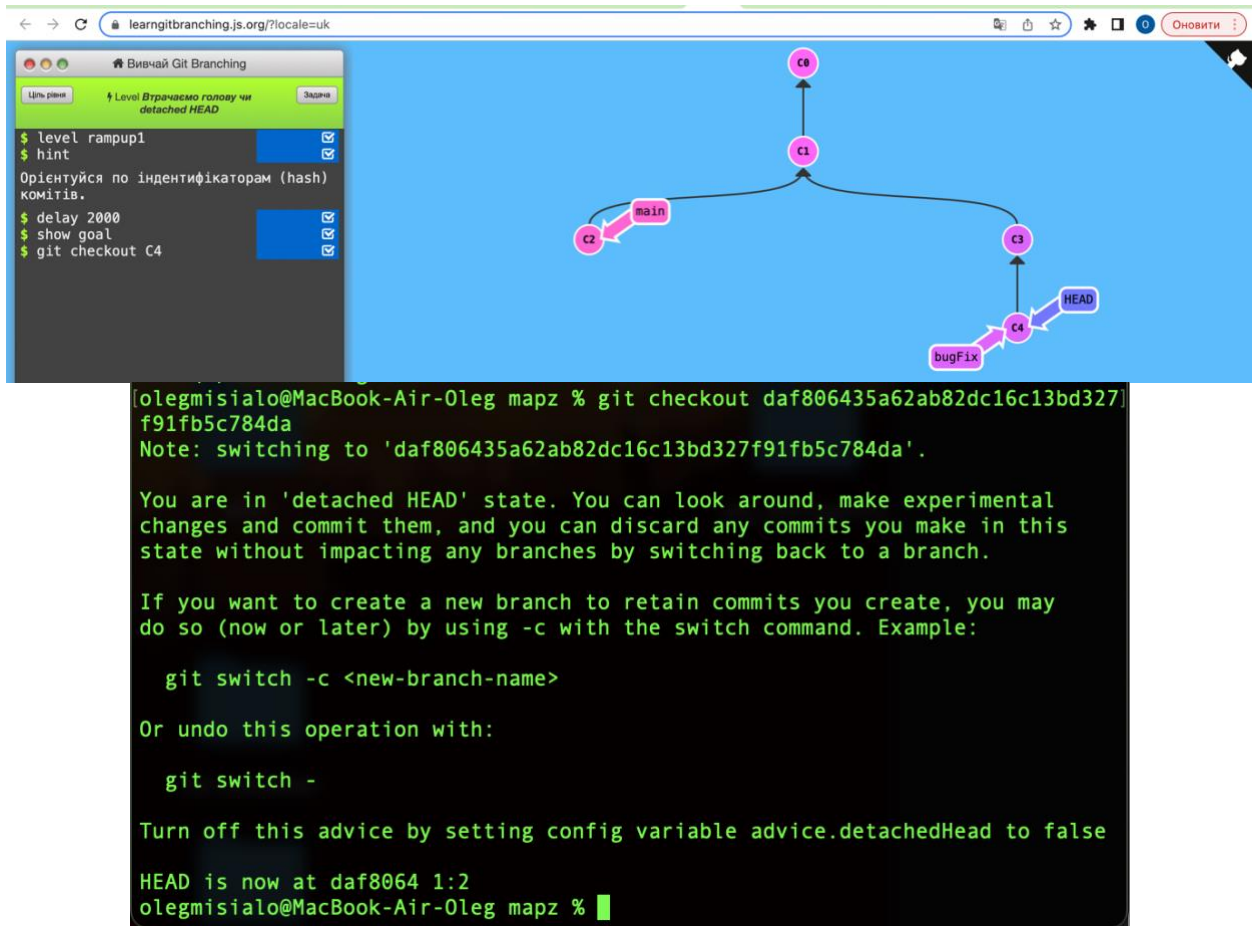
```

olegmisialo@MacBook-Air-Oleg mapz % git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 1 commit.
(use "git push" to publish your local commits)
olegmisialo@MacBook-Air-Oleg mapz % git commit -m "1:4 com from main"
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
(use "git push" to publish your local commits)

nothing to commit, working tree clean
olegmisialo@MacBook-Air-Oleg mapz % git checkout bugFix
Switched to branch 'bugFix'
Your branch is ahead of 'origin/bugFix' by 2 commits.
(use "git push" to publish your local commits)
olegmisialo@MacBook-Air-Oleg mapz % git rebase main
Successfully rebased and updated refs/heads/bugFix.
olegmisialo@MacBook-Air-Oleg mapz % git push

```

2.1



The screenshot shows the 'Вивчай Git Branching' website. On the left, a sidebar contains a list of tasks with checkboxes. The main area displays a Git branching diagram with branches C0, C1, C2, C3, C4, and HEAD. C0 and C1 are at the top, with C2 branching off C1 to the left and C3 branching off C1 to the right. C4 branches off C3, and HEAD points to C4. A 'bugFix' label is next to C4. Below the diagram, a terminal window shows the following commands and output:

```
[olegmisialo@MacBook-Air-Oleg mapz % git checkout daf806435a62ab82dc16c13bd327f91fb5c784da
Note: switching to 'daf806435a62ab82dc16c13bd327f91fb5c784da'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

  git switch -c <new-branch-name>

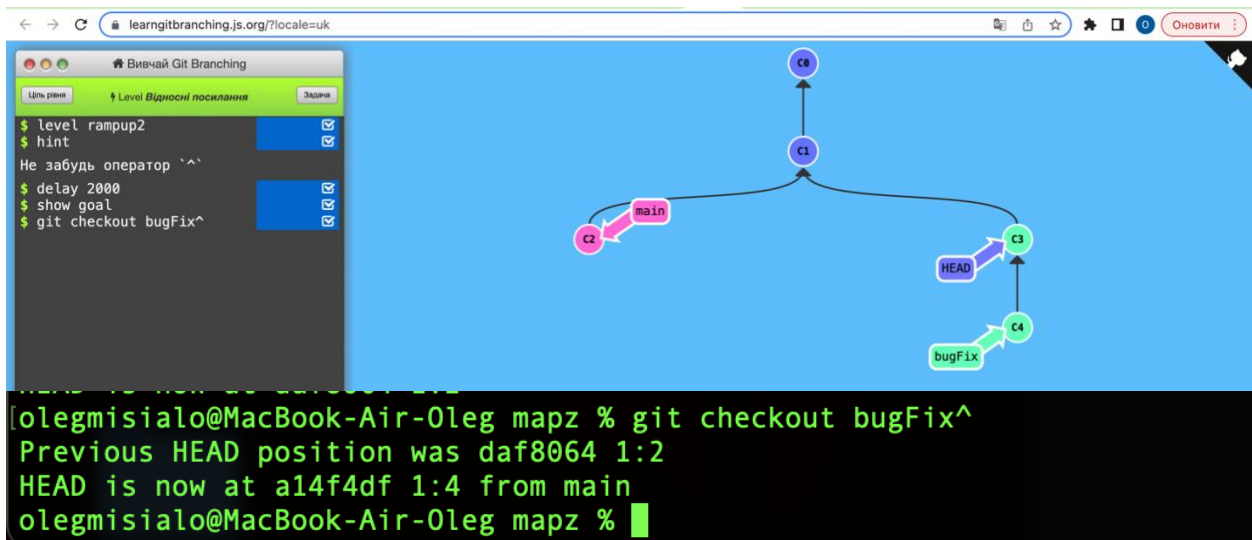
Or undo this operation with:

  git switch -

Turn off this advice by setting config variable advice.detachedHead to false

HEAD is now at daf8064 1:2
olegmisialo@MacBook-Air-Oleg mapz %
```

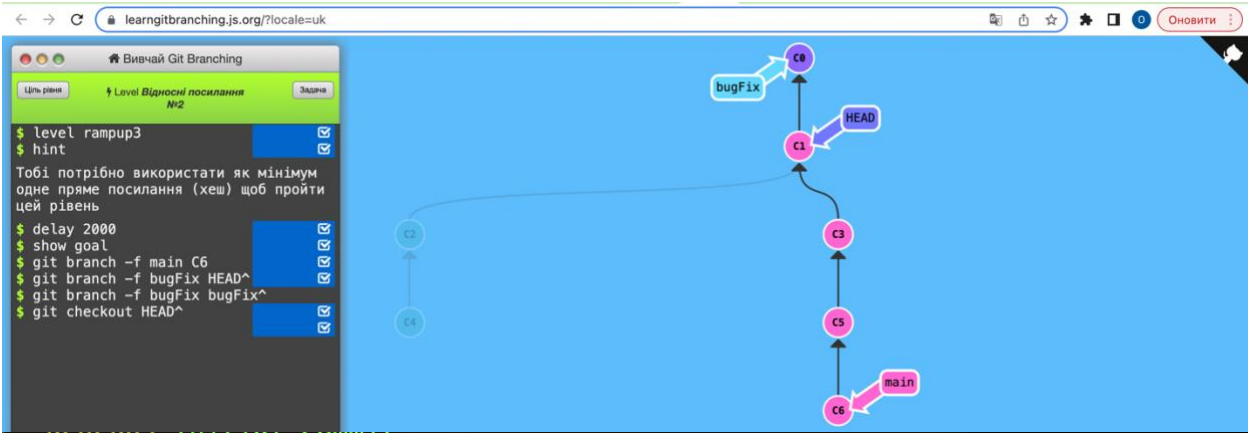
2.2



The screenshot shows the 'Вивчай Git Branching' website. On the left, a sidebar contains a list of tasks with checkboxes. The main area displays a Git branching diagram with branches C0, C1, C2, C3, C4, and HEAD. C0 and C1 are at the top, with C2 branching off C1 to the left and C3 branching off C1 to the right. C4 branches off C3, and HEAD points to C4. A 'bugFix' label is next to C4. Below the diagram, a terminal window shows the following commands and output:

```
[olegmisialo@MacBook-Air-Oleg mapz % git checkout bugFix^
Previous HEAD position was daf8064 1:2
HEAD is now at a14f4df 1:4 from main
olegmisialo@MacBook-Air-Oleg mapz %
```


2.3



```
[olegmisialo@MacBook-Air-Oleg mapz % git branch -f bugFix bugFix~3]
[olegmisialo@MacBook-Air-Oleg mapz % git checkout a14f4df]
Note: switching to 'a14f4df'.
```

You are in 'detached HEAD' state. You can look around, make experimental changes and commit them, and you can discard any commits you make in this state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may do so (now or later) by using `-c` with the switch command. Example:

```
git switch -c <new-branch-name>
```

Or undo this operation with:

```
git switch -
```

Turn off this advice by setting config variable `advice.detachedHead` to `false`

HEAD is now at a14f4df 1:4 from main

```
olegmisialo@MacBook-Air-Oleg mapz % git log --graph --all --oneline --decorate
```

```
* 896d98c (main) 2:3 main2
* c169f9e 2:3 main
| * a581822 (origin/bugFix) 1:4 from bf
|/
* a14f4df (HEAD) 1:4 from main
* 3ae05ba (origin/main, origin/HEAD, bugFix) added 2 lines
* daf8064 1:2
* 520f069 1:1 com2
* a325a7d 1:1 com1
* bf896cf change name
* ac2d3ab Create folder and file
* 59f8437 Delete .DS_Store
* 474d8eb create folder for lab01
* acdcad9 Initial commit
olegmisialo@MacBook-Air-Oleg mapz %
```

2.4

learnitbranching.js.org/?locale=uk

Вивчай Git Branching

Ціль рівня: Level Визначити зміни в Git

Завдання

- \$ level rampup4
- \$ hint
- Зверни увагу на те що revert та reset приймають різні параметри
- \$ delay 2000
- \$ show goal
- \$ git reset local^
- \$ git checkout pushed
- \$ git revert pushed

```
graph TD
    c1((c1)) --> c2((c2))
    c2 --> c3((c3))
    c3 --> c4((c4))
    c1 --> c4
    c2 --> c4
    c3 --> c4
```

olegmisialo@MacBook-Air-Oleg mapz % git log --graph --all --oneline --decorate

```

e
* 1341248 (HEAD -> bugFix) 2:4 com1
| * ef5c715 (main) 2_4 main2
| * eafee50 2_4 main1
|/
* cc40fb6 ///
* 1684c66 2:4 2
* a581822 (origin/bugFix) 1:4 from bf
* a14f4df 1:4 from main
* 3ae05ba (origin/main, origin/HEAD) added 2 lines
* daf8064 1:2
* 520f069 1:1 com2
* a325a7d 1:1 com1
* bf896cf change name
* ac2d3ab Create folder and file
* 59f8437 Delete .DS_Store
* 474d8eb create folder for lab01
* acdcad9 Initial commit

```

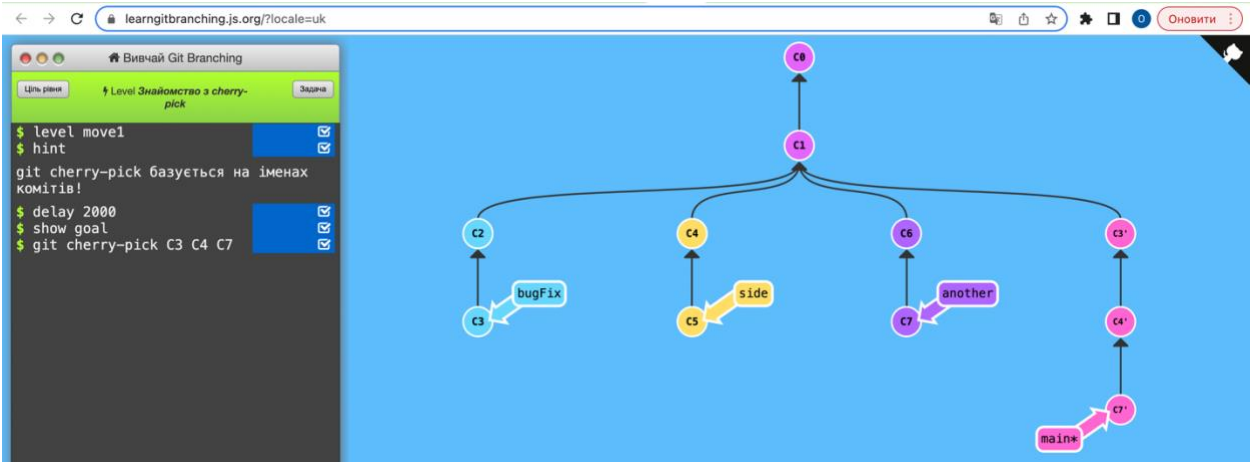
olegmisialo@MacBook-Air-Oleg mapz % git log --graph --all --oneline --decorate

```

* 04e149a (bugFix) 2:4 bf2
* dab1a8b 2:4 bf
| * eafee50 (HEAD -> main) 2_4 main1
|/
* cc40fb6 ///
* 1684c66 2:4 2
* a581822 (origin/bugFix) 1:4 from bf
* a14f4df 1:4 from main
* 3ae05ba (origin/main, origin/HEAD) added 2 lines

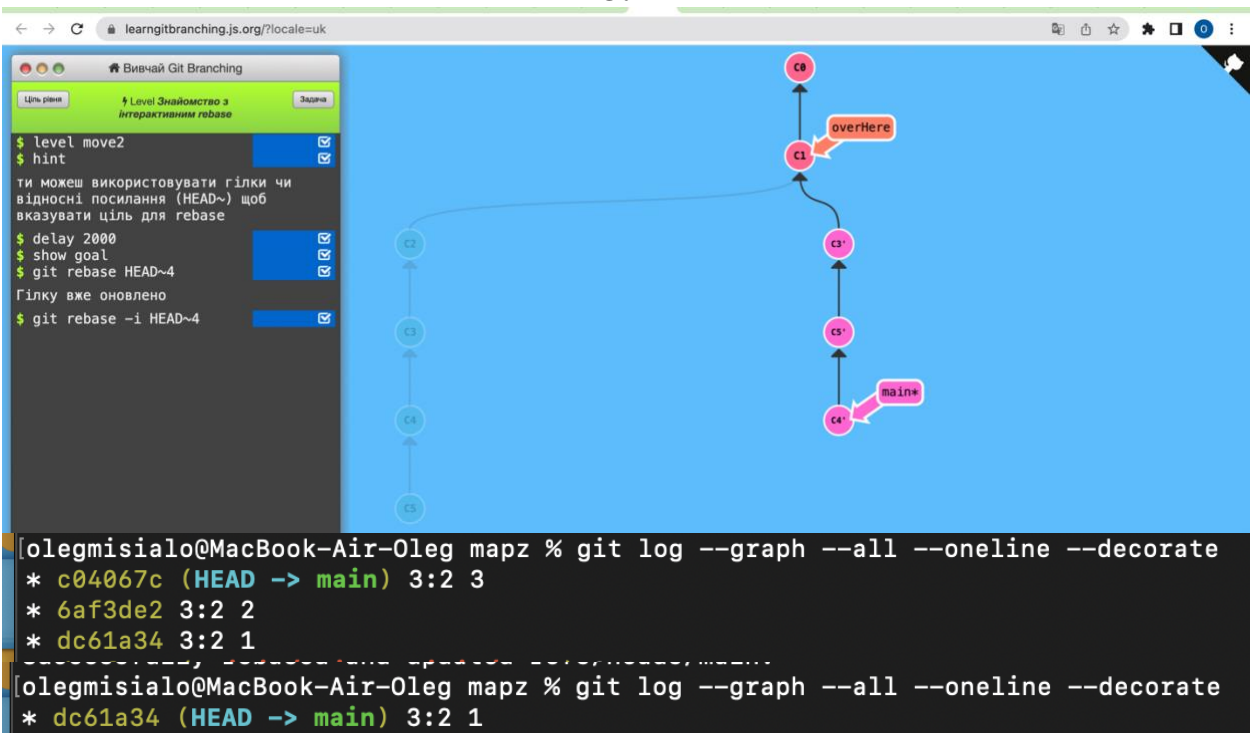
```


3.1



```
(use "git push" to publish your local commits)
[olegmisialo@MacBook-Air-Oleg mapz % git cherry-pick db22b9a
[main d7cf11e] 3:1
Date: Sun Mar 5 21:44:28 2023 +0200
1 file changed, 1 insertion(+)
[olegmisialo@MacBook-Air-Oleg mapz % git log --graph --all --oneline --decorate
* d7cf11e (HEAD -> main) 3:1
| * db22b9a (bugFix) 3:1
|/
```

3.2



4.1

learnitgitbranching.js.org/?locale=uk

Вивчай Git Branching

Ціль рівня: **Level: Виберишмо всього один коміт**

Завдання:

```
$ level mixed1
$ hint
Не забувай, що інтерактивний rebase та
cherry-pick -- це твої друзі!
$ delay 2000
$ show goal
$ git checkout main
$ git cherry-pick C4
```

```
* 97b40a8 (HEAD -> bugFix) 4:1
* db22b9a 3:1
| * dc61a34 (main) 3:2 1
| * d7cf11e (origin/main, origin/HEAD) 3:1
|/
/
```

```
olegmisialo@MacBook-Air-Oleg mapz % git log --graph --all --oneline --decorate
* a7dec6b (HEAD -> main) 3:1 2
* dc61a34 3:2 1
* d7cf11e (origin/main, origin/HEAD) 3:1
| * 97b40a8 (bugFix) 4:1
| * db22b9a 3:1
|/
/
```

4.2

learnitgitbranching.js.org/?locale=uk

Вивчай Git Branching

Ціль рівня: **Level: Жонглюємо комітми**

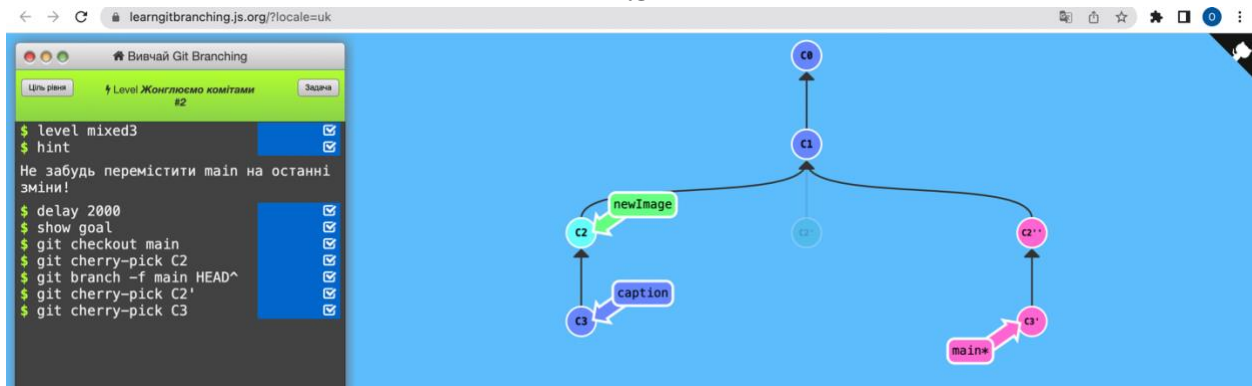
Завдання:

```
$ level mixed2
$ hint
Перша команда має бути git rebase -i
HEAD~2
$ delay 2000
$ show goal
$ git rebase -i HEAD~2
$ objective
$ git commit --amend
$ git rebase -i HEAD~2
$ git branch -f main caption
```

```
* a7dec6b (HEAD -> main) 3:1 2
* dc61a34 3:2 1
| * 3511433 (HEAD -> main) 3:2 1
| * a43c9d9 3:1 2
| * d7cf11e (origin/main, origin/HEAD) 3:1
|/
/
```

```
olegmisialo@MacBook-Air-Oleg mapz % git log --graph --all --oneline --decorate
* a7dec6b (HEAD -> main) 3:1 2
* dc61a34 3:2 1
| * 3511433 (HEAD -> main) 3:2 1
| * a43c9d9 3:1 2
| * d7cf11e (origin/main, origin/HEAD) 3:1
|/
/
```

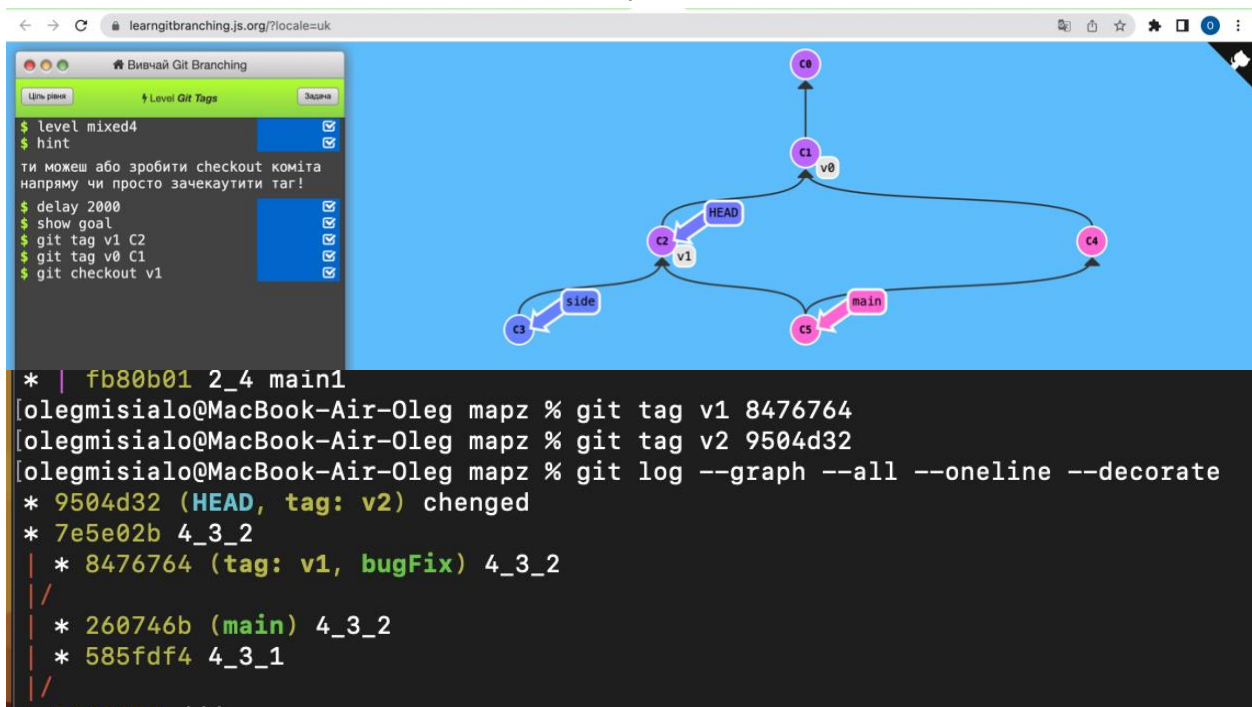
4.3



```
1 file changed, 2 insertions(+), 2 deletions(-)
[olegmisialo@MacBook-Air-Oleg mapz % git log --graph --all --oneline --decorate
* 260746b (HEAD -> main) 4_3_2
* 585fdf4 4_3_1
* 20f84b7 ///
```

```
[olegmisialo@MacBook-Air-Oleg mapz % git cherry-pick --skip
[olegmisialo@MacBook-Air-Oleg mapz % git log --graph --all --oneline --decorate
* 9504d32 (HEAD) changed
* 7e5e02b 4_3_2
| * 8476764 (bugFix) 4_3_2
|/
| * 260746b (main) 4_3_2
| * 585fdf4 4_3_1
|/
* 20f84b7 ///
```

4.4



```
* | fb80b01 2_4 main1
[olegmisialo@MacBook-Air-Oleg mapz % git tag v1 8476764
[olegmisialo@MacBook-Air-Oleg mapz % git tag v2 9504d32
[olegmisialo@MacBook-Air-Oleg mapz % git log --graph --all --oneline --decorate
* 9504d32 (HEAD, tag: v2) changed
* 7e5e02b 4_3_2
| * 8476764 (tag: v1, bugFix) 4_3_2
|/
| * 260746b (main) 4_3_2
| * 585fdf4 4_3_1
|/
* 20f84b7 ///
```

4.5

Вивчай Git Branching

Ціль рівня: Level Git Describe

- \$ level mixed5
- \$ hint
- Просто зроби один коміт в bugFix коли ти будеш готовий іти далі
- \$ delay 2000
- \$ show goal
- \$ git describe C4
- v1_1_gC4
- \$ git describe C6
- v1_2_gC6
- \$ git describe main
- v0_2_gC2
- \$ git commit

v2

```

[olegmisialo@MacBook-Air-Oleg mapz % git describe --tags HEAD
v2-1-g12e72f4
[olegmisialo@MacBook-Air-Oleg mapz % git log --graph --all --oneline --decorate
* 12e72f4 (HEAD) describe
* 9504d32 (tag: v2) changed
* 7e5e02b 4_3_2
* 8476764 (tag: v1, bugFix) 4_3_2
//
* 260746b (main) 4_3_2
* 585fdf4 4_3_1
//

```

Вивчай Git Branching

Основи Віддалені репозиторії

Вступ

Гарно підібране введення в основні команди git

1: Знайомство з комітами в Git

Ідемо далі

Наступна порція абсолютної git-дивини. Сподіваюсь, ви зголовились

1: Втрачаємо голову чи detached HEAD

Переміщуємо роботу туди-сюди

Не соромимось змінювати історію

1: Знайомство з cherry-pick

Всяке

Різні прийоми роботи з Git, хитрощі та поради

1: Вибірємо всього один коміт

Досвідчений рівень

Для хоробрих

Скрін меню сайту

Висновок

Виконуючи дану лабораторну роботу я попрактикувався та покращив навички роботи з Git-ом.