Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Технології розроблення програмного забезпечення

Лабораторна робота №1

«Системи контролю версій. Git»

Виконала:
Студентка групи ІА-22
Третякова Ельвіра

Перевірив:
Мягкий Михайло Юрійович

Київ 2024

ЗМІСТ

ХІД РОБОТИ: 3

Крок 1. Ініціалізація репозиторію та перевірка статусу цього репозиторію: 3
Крок 2. Створення порожнього коміту та перегляд журналу комітів: 3
Крок 3. Створення гілок 3-ма різними способами: 3
Крок 4. Перемикання на конкретний коміт у режимі відокремленої HEAD: 4
Крок 5. Створення файлів, перемикання між гілками та коміти змін у відповідні файли: 5
Крок 6. Перегляд журналу комітів у всіх гілках: 7
Крок 7. Злиття branch1 з main: 7
Крок 8. Оновлення гілки branch2 до останньої версії main за допомогою rebase(з вирішенням конфліктів): 8
Крок 9. Перегляд історії комітів: 9

ВИСНОВОК 10

Хід роботи:

Крок 1. Ініціалізація репозиторію та перевірка статусу цього репозиторію:

```
Last login: Sat Oct 12 10:20:14 on ttys000
elatretakova@MacBook-Air-Ela test % git init
Initialized empty Git repository in /Users/elatretakova/Desktop/test/.git/
elatretakova@MacBook-Air-Ela test % git status
On branch main

No commits yet

nothing to commit (create/copy files and use "git add" to track)
```

git init - Ініціалізується новий порожній Git репозиторій у вказаній директорії. Це створює .git папку, де зберігаються всі версії проєкту.

git status - Перевіряється поточний стан репозиторію, зокрема відсутність файлів для відстеження або комітів на гілці main.

Крок 2. Створення порожнього коміту та перегляд журналу комітів:

```
elatretakova@MacBook-Air-Ela test % git commit --allow-empty -m "1 commit"
[main (root-commit) abcc28c] 1 commit
elatretakova@MacBook-Air-Ela test % git log
commit abcc28c68bd347231c9835a62d46a67fe459c201 (HEAD -> main)
Author: Ellviraaaaaa <elvira.tretiakova@gmail.com>
Date:   Sat Oct 12 11:26:04 2024 +0300

    1 commit
elatretakova@MacBook-Air-Ela test % git log --oneline
abcc28c (HEAD -> main) 1 commit
```

git commit --allow-empty -m "1 commit" - Створюється перший порожній коміт з повідомленням "1 commit", що дозволяє зафіксувати стан репозиторію на початку.

Крок 3. Створення гілок 3-ма різними способами:

```
elatretakova@MacBook-Air-Ela test % git branch branch1
elatretakova@MacBook-Air-Ela test % git checkout -b branch2
Switched to a new branch 'branch2'
elatretakova@MacBook-Air-Ela test % git switch main
Switched to branch 'main'
elatretakova@MacBook-Air-Ela test % git switch -c branch3
Switched to a new branch 'branch3'
```

git branch branch1 - Створюється нова гілка branch1 без автоматичного перемикання на неї. Ця команда просто додає нову гілку у поточному репозиторії.

git checkout -b branch2 - Одночасно створюється нова гілка branch2 і виконується перемикання на неї.

git switch main - Перемикання з поточної гілки на основну гілку main. Ця команда повертає вас до основної версії проєкту.

git switch -c branch3 - Створюється нова гілка branch3, після чого автоматично виконується перемикання на неї для подальшої роботи.

Крок 4. Перемикання на конкретний коміт у режимі відокремленої HEAD:

```
elatretakova@MacBook-Air-Ela test % echo 4 > 4.txt
elatretakova@MacBook-Air-Ela test % git add 4.txt
elatretakova@MacBook-Air-Ela test % git commit -m "Add 4.txt"
[branch3 1e896b5] Add 4.txt
 1 file changed, 1 insertion(+)
 create mode 100644 4.txt
elatretakova@MacBook-Air-Ela test % git log --oneline
1e896b5 (HEAD -> branch3) Add 4.txt
abcc28c (main, branch2, branch1) 1 commit
elatretakova@MacBook-Air-Ela test % git log --oneline branch3
1e896b5 (HEAD -> branch3) Add 4.txt
abcc28c (main, branch2, branch1) 1 commit
elatretakova@MacBook-Air-Ela test % echo 5 >> 4.txt
elatretakova@MacBook-Air-Ela test % git status
On branch branch3
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   4.txt

no changes added to commit (use "git add" and/or "git commit -a")
elatretakova@MacBook-Air-Ela test % git commit -am "mode 4.txt"
[branch3 cb70170] mode 4.txt
 1 file changed, 1 insertion(+)
elatretakova@MacBook-Air-Ela test % git log --oneline branch3
cb70170 (HEAD -> branch3) mode 4.txt
1e896b5 Add 4.txt
abcc28c (main, branch2, branch1) 1 commit
elatretakova@MacBook-Air-Ela test % git switch 1e896b5
fatal: a branch is expected, got commit '1e896b5'
hint: If you want to detach HEAD at the commit, try again with the --detach option.
elatretakova@MacBook-Air-Ela test % git switch 1e896b5 --detach
HEAD is now at 1e896b5 Add 4.txt
elatretakova@MacBook-Air-Ela test % git log branch3 --oneline
cb70170 (branch3) mode 4.txt
1e896b5 (HEAD) Add 4.txt
abcc28c (main, branch2, branch1) 1 commit
elatretakova@MacBook-Air-Ela test % git switch cb70170 --detach
Previous HEAD position was 1e896b5 Add 4.txt
HEAD is now at cb70170 mode 4.txt
```

echo 4 > 4.txt - Створюється новий файл “4.txt” з вмістом “4”.

git add 4.txt - Файл “4.txt” додається до індексу для підготовки до коміту.

git commit -m “Add 4.txt” - Комітиться новий файл із повідомленням “Add 4.txt”. Створено новий коміт з хешем “1e896b5”.

git log –oneline - Виводиться короткий список комітів у поточній гілці. Відображається новий коміт “1e896b5”, а також попередній коміт “abcc28c”.

git log --oneline branch3 - Виводиться список комітів гілки branch3, який включає коміт “1e896b5” і попередній “abcc28c”.

echo 5 >> 4.txt - Додається рядок “5” до файлу “4.txt”.

git status - Виводиться поточний статус гілки “branch3”. Змінено файл “4.txt”, але зміни ще не додані до індексу.

git commit -am “mode 4.txt” - Використовується опція “-a”, щоб автоматично додати всі змінені файли до індексу та комітити їх з повідомленням “mode 4.txt”. Створено новий коміт з хешем “cb70170”.

git log --oneline branch3 - Виводиться оновлений список комітів у гілці branch3, включаючи новий коміт “cb70170” і попередній коміт “1e896b5”.

git switch 1e896b5 - Команда повертає помилку, оскільки git switch очікує гілку, а не коміт. Для перемикання на коміт слід використовувати опцію --detach.

git switch 1e896b5 –detach - Перемикається на коміт “1e896b5” без прив'язки до гілки. HEAD стає відокремленою (detached).

git log branch3 –oneline - Виводиться історія комітів гілки branch3, що показує, що поточний HEAD у відокремленому стані на коміті “1e896b5”.

git switch cb70170 –detach - Перемикається на коміт “cb70170” без прив'язки до гілки. HEAD знову відокремлена.

<u>Крок 5. Створення файлів, перемикання між гілками та коміти змін у відповідні файли:</u>

```
elatretakova@MacBook-Air-Ela test % echo 1 > main.txt
elatretakova@MacBook-Air-Ela test % echo 1 > 1.txt
elatretakova@MacBook-Air-Ela test % echo 2 > 2.txt
elatretakova@MacBook-Air-Ela test % echo 3 > 3.txt
elatretakova@MacBook-Air-Ela test % git add .
elatretakova@MacBook-Air-Ela test % git commit 3.txt -m "Add 3.txt"
[detached HEAD 89ba35d] Add 3.txt
 1 file changed, 1 insertion(+)
 create mode 100644 3.txt
elatretakova@MacBook-Air-Ela test % git switch branch2
A       1.txt
A       2.txt
A       main.txt
Warning: you are leaving 1 commit behind, not connected to
any of your branches:

  89ba35d Add 3.txt

If you want to keep it by creating a new branch, this may be a good time
to do so with:

 git branch <new-branch-name> 89ba35d

Switched to branch 'branch2'
elatretakova@MacBook-Air-Ela test % git branch branch3 89ba35d
fatal: a branch named 'branch3' already exists
elatretakova@MacBook-Air-Ela test % git switch branch3
A       1.txt
A       2.txt
A       main.txt
Switched to branch 'branch3'
elatretakova@MacBook-Air-Ela test % git log
commit cb7017045d2f57d6e44440db5eb4d6ffd9377f3c (HEAD -> branch3)
Author: Ellviraaaaaa <elvira.tretiakova@gmail.com>
Date:   Sat Oct 12 11:35:55 2024 +0300

    mode 4.txt

commit 1e896b5d43f7024b56d617aaa1dcc81aad1c32cb
Author: Ellviraaaaaa <elvira.tretiakova@gmail.com>
Date:   Sat Oct 12 11:33:41 2024 +0300

    Add 4.txt

commit abcc28c68bd347231c9835a62d46a67fe459c201 (main, branch2, branch1)
Author: Ellviraaaaaa <elvira.tretiakova@gmail.com>
Date:   Sat Oct 12 11:26:04 2024 +0300
```

```
elatretakova@MacBook-Air-Ela test % echo 3 > 3.txt
elatretakova@MacBook-Air-Ela test % git add 3.txt
elatretakova@MacBook-Air-Ela test % git commit 3.txt -m "Add 3"
[branch3 4f12ab7] Add 3
 1 file changed, 1 insertion(+)
 create mode 100644 3.txt
elatretakova@MacBook-Air-Ela test % git log
commit 4f12ab7bee72655c21e049b4360320012345d42d (HEAD -> branch3)
Author: Ellviraaaaaa <elvira.tretiakova@gmail.com>
Date:   Sat Oct 12 11:47:37 2024 +0300

    Add 3

commit cb7017045d2f57d6e44440db5eb4d6ffd9377f3c
Author: Ellviraaaaaa <elvira.tretiakova@gmail.com>
Date:   Sat Oct 12 11:35:55 2024 +0300

    mode 4.txt

commit 1e896b5d43f7024b56d617aaa1dcc81aad1c32cb
Author: Ellviraaaaaa <elvira.tretiakova@gmail.com>
Date:   Sat Oct 12 11:33:41 2024 +0300

    Add 4.txt

commit abcc28c68bd347231c9835a62d46a67fe459c201 (main, branch2, branch1)
Author: Ellviraaaaaa <elvira.tretiakova@gmail.com>
Date:   Sat Oct 12 11:26:04 2024 +0300

    1 commit
elatretakova@MacBook-Air-Ela test % git switch branch2
A       1.txt
A       2.txt
A       main.txt
Switched to branch 'branch2'
elatretakova@MacBook-Air-Ela test % git commit 2.txt -m "Add 2"
[branch2 d4c7fb5] Add 2
 1 file changed, 1 insertion(+)
 create mode 100644 2.txt
elatretakova@MacBook-Air-Ela test % git switch branch1
A       1.txt
A       main.txt
Switched to branch 'branch1'
elatretakova@MacBook-Air-Ela test % git commit 1.txt -m "Add 1"
[branch1 1a37ecc] Add 1
 1 file changed, 1 insertion(+)
 create mode 100644 1.txt

elatretakova@MacBook-Air-Ela test % git switch main
A       main.txt
Switched to branch 'main'
elatretakova@MacBook-Air-Ela test % git commit main.txt -m "Add main"
[main 4e900af] Add main
 1 file changed, 1 insertion(+)
 create mode 100644 main.txt
```

echo 1 > main.txt

echo 1 > 1.txt

echo 2 > 2.txt

echo 3 > 3.txt

Створюються чотири файли з відповідними значеннями.

git add . - Додаються всі нові файли до індексу для підготовки до коміту.

git commit 3.txt -m “Add 3.txt” - Створено новий коміт для файлу “3.txt” з повідомленням “Add 3.txt” у режимі відокремленої HEAD.

git switch branch2 - Перемикання на гілку branch2. Виводиться попередження про втрату коміту “89ba35d”(для його збереження пропонується створити нову гілку).

git branch branch3 89ba35d - Команда завершилася невдачею, оскільки гілка branch3 вже існує.

git switch branch3 - Успішне перемикання на гілку branch3.

echo 3 > 3.txt

git add 3.txt

git commit 3.txt -m “Add 3”

Змінюється вміст файлу “3.txt” та створюється новий коміт з повідомленням “Add 3”.

git switch branch2

git commit 2.txt -m “Add 2”

Перемикання на гілку branch2 та створення коміту для файлу “2.txt” з повідомленням “Add 2”.

git switch branch1

git commit 1.txt -m “Add 1”

Перемикання на гілку branch1 та створення коміту для файлу “1.txt” з повідомленням “Add 1”.

git switch main

git commit main.txt -m “Add main”

Перемикання на гілку main та створення коміту для файлу “main.txt” з повідомленням “Add main”.

Крок 6. Перегляд журналу комітів у всіх гілках:

```
elatretakova@MacBook-Air-Ela test % git log --all --oneline
4e900af (HEAD -> main) Add main
1a37ecc (branch1) Add 1
d4c7fb5 (branch2) Add 2
4f12ab7 (branch3) Add 3
cb70170 mode 4.txt
1e896b5 Add 4.txt
abcc28c 1 commit
```

Крок 7. Злиття branch1 з main:

```
elatretakova@MacBook-Air-Ela test % git switch branch1
Switched to branch 'branch1'
elatretakova@MacBook-Air-Ela test % git merge main
Merge made by the 'ort' strategy.
 main.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 main.txt
elatretakova@MacBook-Air-Ela test % git switch main
Switched to branch 'main'
elatretakova@MacBook-Air-Ela test % echo a > 2.txt
elatretakova@MacBook-Air-Ela test % echo a > 3.txt
elatretakova@MacBook-Air-Ela test % git add .
elatretakova@MacBook-Air-Ela test % git commit -m "Add 2 3"
[main 91fe868] Add 2 3
 2 files changed, 2 insertions(+)
 create mode 100644 2.txt
 create mode 100644 3.txt
elatretakova@MacBook-Air-Ela test % git swich branch2
git: 'swich' is not a git command. See 'git --help'.

The most similar command is
        switch
```

git switch branch1 - Перемикання на гілку branch1.

git merge main - Злиття гілки main у branch1 з автоматичним застосуванням змін. Файл main.txt було додано до branch1.

git switch main - Перемикання на гілку main.

echo a > 2.txt і echo a > 3.txt - Запис символу “a” у файли 2.txt і 3.txt.

git add . - Додавання змін для коміту.

git commit -m “Add 2 3” - Коміт файлів 2.txt і 3.txt з повідомленням “Add 2 3”.

<u>Крок 8. Оновлення гілки branch2 до останньої версії main за допомогою rebase(з вирішенням конфліктів):</u>

```
elatretakova@MacBook-Air-Ela test % git switch branch2
Switched to branch 'branch2'
elatretakova@MacBook-Air-Ela test % git rebase main
Auto-merging 2.txt
CONFLICT (add/add): Merge conflict in 2.txt
error: could not apply d4c7fb5... Add 2
hint: Resolve all conflicts manually, mark them as resolved with
hint: "git add/rm <conflicted_files>", then run "git rebase --continue".
hint: You can instead skip this commit: run "git rebase --skip".
hint: To abort and get back to the state before "git rebase", run "git rebase --abort".
Could not apply d4c7fb5... Add 2
elatretakova@MacBook-Air-Ela test % git add 2.txt
elatretakova@MacBook-Air-Ela test % git rebase --continue
[detached HEAD 15b7d46] Add 2
 1 file changed, 2 insertions(+)
Successfully rebased and updated refs/heads/branch2.
```

git switch branch2 - Перемикання на гілку branch2.

git rebase main - Спроба перенесення гілки branch2 на гілку main. Виникає конфлікт через зміни у файлі “2.txt”.

git add 2.txt - Після ручного вирішення конфлікту у файлі “2.txt”, файл додається для завершення процесу оновлення.

git rebase --continue - Завершення оновлення після вирішення конфліктів. Оновлення успішно завершено, і гілка branch2 тепер відображає зміни з гілки main.

Крок 8. Cherry-pick з вирішенням конфлікту під час додавання комітів з main на branch3:

git switch branch3 - Перемикання на гілку branch3.

git log main –oneline - Показ комітів, що містяться в гілці main, для вибору необхідних комітів для cherry-pick.

git cherry-pick 4e900af 91fe868 - Перенесення (cherry-pick) комітів 4e900af (Add main) та 91fe868(Add 2 3) з гілки main у гілку branch3. Виникає конфлікт через зміни у файлі “3.txt”.

git add 3.txt - Після ручного вирішення конфлікту у файлі “3.txt”, файл додається для завершення cherry-pick.

git cherry-pick --continue - Завершення cherry-pick після вирішення конфліктів. Коміти успішно перенесені на гілку branch3.

Крок 9. Перегляд історії комітів:

```
elatretakova@MacBook-Air-Ela test % git log --all --oneline
15334f5 (HEAD -> branch3) Add 2 3
c81c238 Add main
15b7d46 (branch2) Add 2
91fe868 (main) Add 2 3
05287a6 (branch1) Merge branch 'main' into branch1
4e900af Add main
1a37ecc Add 1
4f12ab7 Add 3
cb70170 mode 4.txt
1e896b5 Add 4.txt
abcc28c 1 commit
elatretakova@MacBook-Air-Ela test % git log --all --oneline --graph
* 15334f5 (HEAD -> branch3) Add 2 3
* c81c238 Add main
* 4f12ab7 Add 3
* cb70170 mode 4.txt
* 1e896b5 Add 4.txt
| * 15b7d46 (branch2) Add 2
| * 91fe868 (main) Add 2 3
| | *   05287a6 (branch1) Merge branch 'main' into branch1
| | |\
| | |/
| |/|
| * | 4e900af Add main
|/ /
| * 1a37ecc Add 1
|/
* abcc28c 1 commit
elatretakova@MacBook-Air-Ela test %
```

git log --all –oneline - Показує всі коміти в проєкті у скороченому форматі. Всі гілки і коміти видно в хронологічному порядку.

git log --all --oneline --graph - Показує всі коміти в проєкті у вигляді графу, що ілюструє, як коміти і гілки пов’язані між собою. Це дозволяє наочно побачити, де відбувалися злиття гілок або відгалуження.

ВИСНОВОК

В даній лабораторній роботі я ознайомилась з основними командами Git. Вивчила базові операції з гілками в Git, також навчилась вирішувати конфлікти. Використання графічного відображення історії допомогло візуалізувати взаємодію між гілками і комітами.