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

Лабораторна робота *№1*

із дисципліни ***«Технології розробки програмного забезпечення»***

Тема: ***«Системи контроля версій. Розподілена система контролю версій «Git»»***

Виконав:

Студент групи ІА-34

Ястремський Богдан

Перевірив:

асистент кафедри ІСТ

Мягкий Михайло Юрійович

Київ — 2025

Тема: Системи контроля версій. Розподілена система контролю версій «Git».

Мета: Навчитися виконувати основні операції в роботі з децентралізованими системами контролю версій на прикладі роботи з сучасною системою Git.

Короткі теоретичні відомості:

Основна ідея Git, як і будь-якої іншої розподіленої системи контроля версій – кожен розробник має власний репозиторій, куди складаються зміни (версії) файлів, та синхронізація між розробниками виконується за допомогою синхронізації репозиторіїв.

Утиліта git дозволяє працювати з локальними і віддаленими репозиторіями за допомогою власних команд. Одна з таких — git init [repo_name]. Створює порожній безіменний або з ім'ям [repo_name] репозиторій. Зберегти зміни, переглянути їх статус, і закомітити повідомлення можна відповідно за допомогою команд:

- git add [.] – крапка означає всі зміни в поточному каталозі
- git status
- git commit [-m] [repo_message] (пустий або з повідомленням).

Створити гілку можна 2 різними методами:

- git branch [name] – створення гілки без переходу (git checkout [name]) на неї
- git checkout -b [name] – з переходом
- git branch покаже всі гілки в локальному репозиторії

Об'єднати дві гілки можна за допомогою git merge [name] – де name є гілкою з якою ми будемо з'єднувати поточну гілку (master).

Переглянути історію всіх комітів у вигляді графу можна за допомогою git log –all –graph.

Хід роботи:

1. Створити локальний в папці безіменний репозиторій.

```
C:\Users\user>cd Desktop

C:\Users\user\Desktop>mkdir test

C:\Users\user\Desktop>cd test

C:\Users\user\Desktop\test>git init
Initialized empty Git repository in C:/Users/user/Desktop/test/.git/
```

2. Спробувати його закомітити, і впевнитись, що без жодних файлів закомітити не вийде. Тому додаємо файл, зберігаємо зміни і комітимо.

```
C:\Users\user\Desktop\test>git commit -m "Test repo added"
On branch master

Initial commit

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

C:\Users\user\Desktop\test>echo "parent" > parent.txt

C:\Users\user\Desktop\test>git status
On branch master

No commits yet

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

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

C:\Users\user\Desktop\test>git add .

C:\Users\user\Desktop\test>git commit -m "Parent.txt added"
[master (root-commit) 5d91fc0] Parent.txt added
 1 file changed, 1 insertion(+)
 create mode 100644 parent.txt
```

3. Створити нову гілку і переглянути її наявність.

```
C:\Users\user\Desktop\test>git branch branch1

C:\Users\user\Desktop\test>git status
On branch master
nothing to commit, working tree clean

C:\Users\user\Desktop\test>git branch
  branch1
* master
```

4. Створюємо нову гілку іншим способом і перевіряємо граф комітів, де маємо побачити всі гілки в одному коміті HEAD.

```
C:\Users\user\Desktop\test>git checkout -b branch2
Switched to a new branch 'branch2'

C:\Users\user\Desktop\test>git branch
  branch1
* branch2
  master

C:\Users\user\Desktop\test>git log --all --graph
* commit 5d91fc038a923d5c8699ec6099fd82aa291b246e (HEAD -> branch2, master, branch1)
  Author: bohdanyast <yastremskyi.bohdan@lll.kpi.ua>
  Date:   Sat Sep 13 17:50:31 2025 +0300

      Parent.txt added
```

5. Додати файли по різним гілкам і перевіряємо наявність розподілення гілок по комітах.

```
C:\Users\user\Desktop\test>echo "1" > file1.txt

C:\Users\user\Desktop\test>echo "2" > file2.txt

C:\Users\user\Desktop\test>git add file2.txt

C:\Users\user\Desktop\test>git commit -m "file2.txt added"
[branch2 9a69ec5] file2.txt added
 1 file changed, 1 insertion(+)
 create mode 100644 file2.txt

C:\Users\user\Desktop\test>git checkout branch1
Switched to branch 'branch1'

C:\Users\user\Desktop\test>git add file1.txt

C:\Users\user\Desktop\test>git commit -m "file1.txt added"
[branch1 e9b62ad] file1.txt added
 1 file changed, 1 insertion(+)
 create mode 100644 file1.txt
```

```
C:\Users\user\Desktop\test>git log --all --graph
* commit e9b62ad30171ccdfc523e76afd2ef19fc36032d3 (HEAD -> branch1)
| Author: bohdanyast <yastremskyi.bohdan@lll.kpi.ua>
| Date:   Sat Sep 13 17:56:26 2025 +0300
|
|     file1.txt added
|
| * commit 9a69ec54de96a5a6e6d07ae9ed0069a2db2e13f8 (branch2)
|/  Author: bohdanyast <yastremskyi.bohdan@lll.kpi.ua>
|   Date:   Sat Sep 13 17:55:53 2025 +0300
|
|       file2.txt added
|
* commit 5d91fc038a923d5c8699ec6099fd82aa291b246e (master)
  Author: bohdanyast <yastremskyi.bohdan@lll.kpi.ua>
  Date:   Sat Sep 13 17:50:31 2025 +0300

      Parent.txt added
```

6. Створимо конфліктну ситуацію шляхом додавання нового вмісту у file2.txt. Впевнимось під час з’єднання гілок, що помилка саме в ньому. Виправимо зміни у файлі вручну, видаливши символи <, =, >, і замінивши, наприклад на «2A» і впевнимось, що все працює.

```
C:\Users\user\Desktop\test>echo "A" > file2.txt
```

```
C:\Users\user\Desktop\test>git commit -m "added"
[branch1 0cdbefa] added
 1 file changed, 1 insertion(+)
 create mode 100644 file2.txt

C:\Users\user\Desktop\test>git merge branch1
Already up to date.

C:\Users\user\Desktop\test>git merge branch2
Auto-merging file2.txt
CONFLICT (add/add): Merge conflict in file2.txt
Automatic merge failed; fix conflicts and then commit the result.
```

```
C:\Users\user\Desktop\test>git merge --continue
[branch1 cb9cd7a] Merge branch 'branch2' into branch1

C:\Users\user\Desktop\test>git log --all --graph
*   commit cb9cd7a29c8038e3c9f5be88fcc04acc701dc15e (HEAD -> branch1)
|\  Merge: 0cdbefa 9a69ec5
| | Author: bohdanyast <yastremskyi.bohdan@lll.kpi.ua>
| | Date:   Sat Sep 13 18:04:08 2025 +0300
| |
| |     Merge branch 'branch2' into branch1
| |
| * commit 9a69ec54de96a5a6e6d07ae9ed0069a2db2e13f8 (branch2)
| | Author: bohdanyast <yastremskyi.bohdan@lll.kpi.ua>
| | Date:   Sat Sep 13 17:55:53 2025 +0300
| |
| |     file2.txt added
| |
* | commit 0cdbefa128aa37d6f2d9ee5cb1fd5ae4a186c274
| | Author: bohdanyast <yastremskyi.bohdan@lll.kpi.ua>
| | Date:   Sat Sep 13 18:02:39 2025 +0300
| |
| |     added
| |
* | commit e9b62ad30171ccdfc523e76afd2ef19fc36032d3
|/  Author: bohdanyast <yastremskyi.bohdan@lll.kpi.ua>
|   Date:   Sat Sep 13 17:56:26 2025 +0300
|
|       file1.txt added
|
* commit 5d91fc038a923d5c8699ec6099fd82aa291b246e (master)
  Author: bohdanyast <yastremskyi.bohdan@lll.kpi.ua>
  Date:   Sat Sep 13 17:50:31 2025 +0300

      Parent.txt added

C:\Users\user\Desktop\test>
```

Висновок: на даному лабораторному занятті я познайомився з історією розвитку систем контролю версій, які етапи вони пройшли для того щоб стати зручними у використанні. Також я познайомився з утилітою git, яка дозволяє виконувати різноманітні операції з комітами і гілками. Я навчився базовим речам: створювати коміти з повідомленням, створювати нові гілки, обирати робочу гілку, підготовлювати файли до коміту, а також об’єднувати гілки.