Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота №1

з дисципліни «Технології розроблення програмного забезпечення»

Тема: «Системи контролю версій. Git.»

Виконав:
студент групи ІА-23
Мозоль В.О

Перевірив:
пос. Мягкий М. Ю.

Київ 2024

Тема: Системи контролю версій. Git.

Мета: Ознайомлення з основними принципами роботи системи контролю версій Git, навчитися використовувати базові команди для створення, керування гілками, створення комітів та вирішення конфліктів.

Хід роботи:

Завдання №1 – Створення 4 гілок:

```
User@Lenovo MINGW64 ~/Desktop
$ cd ..

User@Lenovo MINGW64 ~
$ cd /d/gitlab

User@Lenovo MINGW64 /d/gitlab
$ git init
Initialized empty Git repository in D:/gitlab/.git/

User@Lenovo MINGW64 /d/gitlab (main)
$ touch text.txt

User@Lenovo MINGW64 /d/gitlab (main)
$ git status
On branch main

No commits yet

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

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

User@Lenovo MINGW64 /d/gitlab (main)
$ git add .

User@Lenovo MINGW64 /d/gitlab (main)
$ git commit -m "Initial commit"
[main (root-commit) 91e0f54] Initial commit
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 text.txt

User@Lenovo MINGW64 /d/gitlab (main)
$ git status
On branch main
nothing to commit, working tree clean

User@Lenovo MINGW64 /d/gitlab (main)
$ git log
91e0f54 2024-09-20 | Initial commit (HEAD -> main) [feexq]

User@Lenovo MINGW64 /d/gitlab (main)
$ git branch
* main

User@Lenovo MINGW64 /d/gitlab (main)
$ git branch test1
```

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

User@Lenovo MINGW64 /d/gitlab (main)
$ git add .

User@Lenovo MINGW64 /d/gitlab (main)
$ git commit -m "Initial commit"
[main (root-commit) 91e0f54] Initial commit
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 text.txt

User@Lenovo MINGW64 /d/gitlab (main)
$ git status
On branch main
nothing to commit, working tree clean

User@Lenovo MINGW64 /d/gitlab (main)
$ git log
91e0f54 2024-09-20 | Initial commit (HEAD -> main) [feexq]

User@Lenovo MINGW64 /d/gitlab (main)
$ git branch
* main

User@Lenovo MINGW64 /d/gitlab (main)
$ git branch test1

User@Lenovo MINGW64 /d/gitlab (main)
$ git branch
* main
  test1

User@Lenovo MINGW64 /d/gitlab (main)
$ git checkout -b test2
Switched to a new branch 'test2'

User@Lenovo MINGW64 /d/gitlab (test2)
$ git switch main
Switched to branch 'main'

User@Lenovo MINGW64 /d/gitlab (main)
$ git switch -c test3
Switched to a new branch 'test3'

User@Lenovo MINGW64 /d/gitlab (test3)
$ git branch
  main
  test1
  test2
* test3
```

Завдання №2 – Демонстрація роботи merge та rebase з конфліктом та їх вирішення:

Merge:

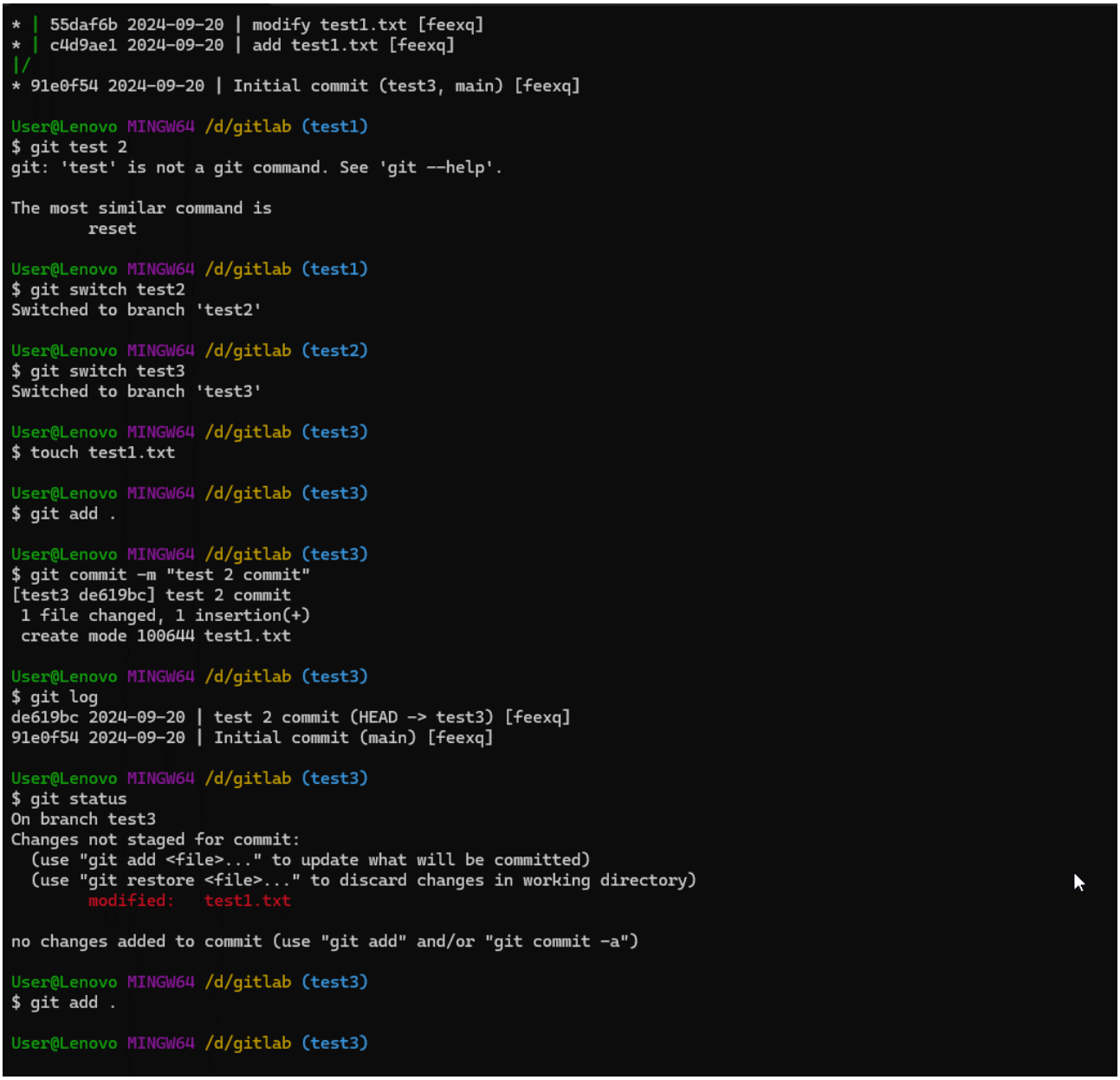
```
User@Lenovo MINGW64 /d/gitlab (main)
$ git switch test1
Switched to branch 'test1'

User@Lenovo MINGW64 /d/gitlab (test1)
$ git status
On branch test1
nothing to commit, working tree clean

User@Lenovo MINGW64 /d/gitlab (test1)
$ touch test1.txt

User@Lenovo MINGW64 /d/gitlab (test1)
$ git status
On branch test1
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        test1.txt

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

User@Lenovo MINGW64 /d/gitlab (test1)
$ git add .

User@Lenovo MINGW64 /d/gitlab (test1)
$ git commit -m "add test1.txt"
[test1 c4d9ae1] add test1.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 test1.txt

User@Lenovo MINGW64 /d/gitlab (test1)
$ git log
c4d9ae1 2024-09-20 | add test1.txt (HEAD -> test1) [feexq]
91e0f54 2024-09-20 | Initial commit (test3, test2, main) [feexq]

User@Lenovo MINGW64 /d/gitlab (test1)
$ git switch test2
Switched to branch 'test2'

User@Lenovo MINGW64 /d/gitlab (test2)
$ git status
On branch test2
nothing to commit, working tree clean

User@Lenovo MINGW64 /d/gitlab (test2)
$ toutch test1.txt
bash: toutch: command not found

User@Lenovo MINGW64 /d/gitlab (test2)
$ touch test1.txt
```

```
User@Lenovo MINGW64 /d/gitlab (test2)
$ git status
On branch test2
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        test1.txt

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

User@Lenovo MINGW64 /d/gitlab (test2)
$ git add .

User@Lenovo MINGW64 /d/gitlab (test2)
$ git commit -m "add test1.txt"
[test2 d8776dc] add test1.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 test1.txt

User@Lenovo MINGW64 /d/gitlab (test2)
$ git switch test1
Switched to branch 'test1'

User@Lenovo MINGW64 /d/gitlab (test1)
$ git status
On branch test1
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:   test1.txt

no changes added to commit (use "git add" and/or "git commit -a")

User@Lenovo MINGW64 /d/gitlab (test1)
$ git add .

User@Lenovo MINGW64 /d/gitlab (test1)
$ git commit -m "modify test1.txt"
[test1 55daf6b] modify test1.txt
 1 file changed, 1 insertion(+)

User@Lenovo MINGW64 /d/gitlab (test1)
$ git merge test2
Auto-merging test1.txt
Merge made by the 'ort' strategy.

User@Lenovo MINGW64 /d/gitlab (test1)
$ git log
23529f5 2024-09-20 | Merge branch 'test2' into test1 (HEAD -> test1) [feexq]
55daf6b 2024-09-20 | modify test1.txt [feexq]
d8776dc 2024-09-20 | add test1.txt (test2) [feexq]
```

```
c4d9ae1 2024-09-20 | add test1.txt [feexq]
91e0f54 2024-09-20 | Initial commit (test3, main) [feexq]

User@Lenovo MINGW64 /d/gitlab (test1)
$ git reset --hard HEAD~1
HEAD is now at 55daf6b modify test1.txt

User@Lenovo MINGW64 /d/gitlab (test1)
$ git log --all --graph
* 55daf6b 2024-09-20 | modify test1.txt (HEAD -> test1) [feexq]
* c4d9ae1 2024-09-20 | add test1.txt [feexq]
| * d8776dc 2024-09-20 | add test1.txt (test2) [feexq]
|/
* 91e0f54 2024-09-20 | Initial commit (test3, main) [feexq]

User@Lenovo MINGW64 /d/gitlab (test1)
$ git log
55daf6b 2024-09-20 | modify test1.txt (HEAD -> test1) [feexq]
c4d9ae1 2024-09-20 | add test1.txt [feexq]
91e0f54 2024-09-20 | Initial commit (test3, main) [feexq]

User@Lenovo MINGW64 /d/gitlab (test1)
$ git switch test 2
fatal: only one reference expected

User@Lenovo MINGW64 /d/gitlab (test1)
$ git switch test2
Switched to branch 'test2'

User@Lenovo MINGW64 /d/gitlab (test2)
$ git add .

User@Lenovo MINGW64 /d/gitlab (test2)
$ git commit -m "modify again test1.txt
> "
[test2 f8e083a] modify again test1.txt
 1 file changed, 1 insertion(+)

User@Lenovo MINGW64 /d/gitlab (test2)
$ git switch test1
Switched to branch 'test1'

User@Lenovo MINGW64 /d/gitlab (test1)
$ git add .

User@Lenovo MINGW64 /d/gitlab (test1)
$ git commit -m "test commit"
[test1 27802b7] test commit
 1 file changed, 4 insertions(+), 1 deletion(-)

User@Lenovo MINGW64 /d/gitlab (test1)
```

```
$ git merge test2
Auto-merging test1.txt
CONFLICT (add/add): Merge conflict in test1.txt
Automatic merge failed; fix conflicts and then commit the result.

User@Lenovo MINGW64 /d/gitlab (test1|MERGING)
$ git status
On branch test1
You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
        both added:      test1.txt

no changes added to commit (use "git add" and/or "git commit -a")

User@Lenovo MINGW64 /d/gitlab (test1|MERGING)
$ git add .

User@Lenovo MINGW64 /d/gitlab (test1|MERGING)
$ git commit -m "Resolve conflict"
[test1 3873181] Resolve conflict

User@Lenovo MINGW64 /d/gitlab (test1)
$ git status
On branch test1
nothing to commit, working tree clean

User@Lenovo MINGW64 /d/gitlab (test1)
$ git log
3873181 2024-09-20 | Resolve conflict (HEAD -> test1) [feexq]
27802b7 2024-09-20 | test commit [feexq]
f8e083a 2024-09-20 | modify again test1.txt (test2) [feexq]
55daf6b 2024-09-20 | modify test1.txt [feexq]
d8776dc 2024-09-20 | add test1.txt [feexq]
c4d9ae1 2024-09-20 | add test1.txt [feexq]
91e0f54 2024-09-20 | Initial commit (test3, main) [feexq]

User@Lenovo MINGW64 /d/gitlab (test1)
$ git log --all --grapgh
fatal: unrecognized argument: --grapgh

User@Lenovo MINGW64 /d/gitlab (test1)
$ git log --all --graph
*   3873181 2024-09-20 | Resolve conflict (HEAD -> test1) [feexq]
|\
| * f8e083a 2024-09-20 | modify again test1.txt (test2) [feexq]
| * d8776dc 2024-09-20 | add test1.txt [feexq]
* | 27802b7 2024-09-20 | test commit [feexq]
```

Rebase:


* | 55daf6b 2024-09-20 | modify test1.txt [feexq]
* | c4d9ae1 2024-09-20 | add test1.txt [feexq]
|/
* 91e0f54 2024-09-20 | Initial commit (test3, main) [feexq]
User@Lenovo MINGW64 /d/gitlab (test1)
$ git test 2
git: 'test' is not a git command. See 'git --help'.
The most similar command is
reset
User@Lenovo MINGW64 /d/gitlab (test1)
$ git switch test2
Switched to branch 'test2'
User@Lenovo MINGW64 /d/gitlab (test2)
$ git switch test3
Switched to branch 'test3'
User@Lenovo MINGW64 /d/gitlab (test3)
$ touch test1.txt
User@Lenovo MINGW64 /d/gitlab (test3)
$ git add .
User@Lenovo MINGW64 /d/gitlab (test3)
$ git commit -m "test 2 commit"
[test3 de619bc] test 2 commit
1 file changed, 1 insertion(+)
create mode 100644 test1.txt
User@Lenovo MINGW64 /d/gitlab (test3)
$ git log
de619bc 2024-09-20 | test 2 commit (HEAD -> test3) [feexq]
91e0f54 2024-09-20 | Initial commit (main) [feexq]
User@Lenovo MINGW64 /d/gitlab (test3)
$ git status
On branch test3
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: test1.txt
no changes added to commit (use "git add" and/or "git commit -a")
User@Lenovo MINGW64 /d/gitlab (test3)
$ git add .
User@Lenovo MINGW64 /d/gitlab (test3)

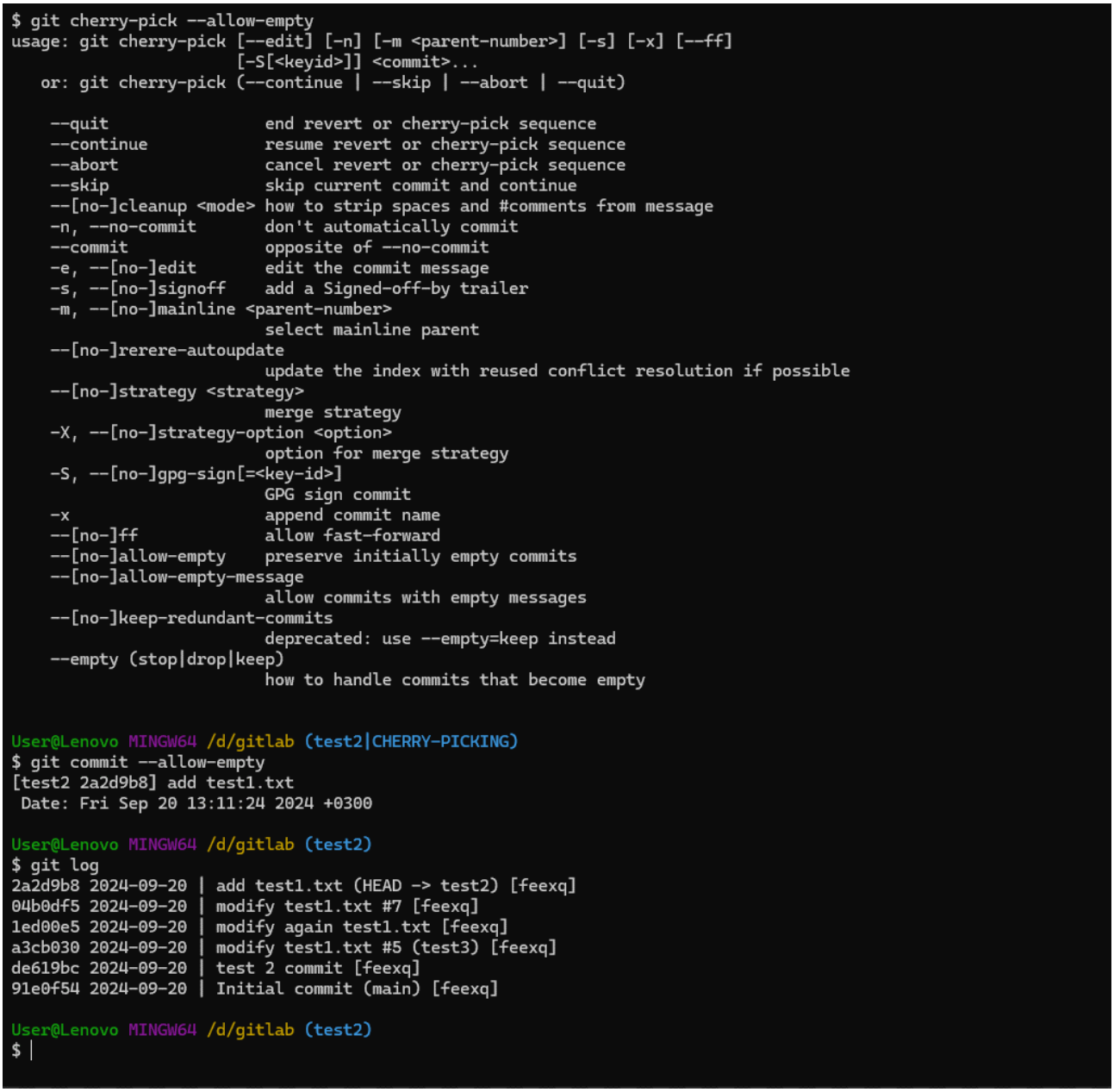
```
$ git commit -m "modify test1.txt #5"
[test3 a3cb030] modify test1.txt #5
 1 file changed, 1 insertion(+)

User@Lenovo MINGW64 /d/gitlab (test3)
$ git status
On branch test3
nothing to commit, working tree clean

User@Lenovo MINGW64 /d/gitlab (test3)
$ git log
a3cb030 2024-09-20 | modify test1.txt #5 (HEAD -> test3) [feexq]
de619bc 2024-09-20 | test 2 commit [feexq]
91e0f54 2024-09-20 | Initial commit (main) [feexq]

User@Lenovo MINGW64 /d/gitlab (test3)
$ git switch test2
Switched to branch 'test2'

User@Lenovo MINGW64 /d/gitlab (test2)
$ git status
On branch test2
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:   test1.txt

no changes added to commit (use "git add" and/or "git commit -a")

User@Lenovo MINGW64 /d/gitlab (test2)
$ git add .

User@Lenovo MINGW64 /d/gitlab (test2)
$ git commit -m "modify test1.txt #7"
[test2 d5fb029] modify test1.txt #7
 1 file changed, 4 insertions(+), 1 deletion(-)

User@Lenovo MINGW64 /d/gitlab (test2)
$ git log
d5fb029 2024-09-20 | modify test1.txt #7 (HEAD -> test2) [feexq]
f8e083a 2024-09-20 | modify again test1.txt [feexq]
d8776dc 2024-09-20 | add test1.txt [feexq]
91e0f54 2024-09-20 | Initial commit (main) [feexq]

User@Lenovo MINGW64 /d/gitlab (test2)
$ git rebase test3
dropping d8776dca7c1bf5244218cc430838ad07a16f41fd add test1.txt -- patch contents already upstream
Auto-merging test1.txt
CONFLICT (content): Merge conflict in test1.txt
error: could not apply f8e083a... modify again test1.txt
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".
hint: Disable this message with "git config advice.mergeConflict false"
Could not apply f8e083a... modify again test1.txt

User@Lenovo MINGW64 /d/gitlab (test2|REBASE 2/3)
$ git status
interactive rebase in progress; onto a3cb030
Last commands done (2 commands done):
   pick d8776dc add test1.txt
   pick f8e083a modify again test1.txt
Next command to do (1 remaining command):
   pick d5fb029 modify test1.txt #7
  (use "git rebase --edit-todo" to view and edit)
You are currently rebasing branch 'test2' on 'a3cb030'.
  (fix conflicts and then run "git rebase --continue")
  (use "git rebase --skip" to skip this patch)
  (use "git rebase --abort" to check out the original branch)

Unmerged paths:
  (use "git restore --staged <file>..." to unstage)
  (use "git add <file>..." to mark resolution)
        both modified:   test1.txt

no changes added to commit (use "git add" and/or "git commit -a")

User@Lenovo MINGW64 /d/gitlab (test2|REBASE 2/3)
$ git rebase -i
fatal: It seems that there is already a rebase-merge directory, and
I wonder if you are in the middle of another rebase.  If that is the
case, please try
        git rebase (--continue | --abort | --skip)
If that is not the case, please
        rm -fr ".git/rebase-merge"
and run me again.  I am stopping in case you still have something
valuable there.


User@Lenovo MINGW64 /d/gitlab (test2|REBASE 2/3)
$ git rebase --continue
test1.txt: needs merge
You must edit all merge conflicts and then
mark them as resolved using git add

User@Lenovo MINGW64 /d/gitlab (test2|REBASE 2/3)
$ git status
interactive rebase in progress; onto a3cb030
Last commands done (2 commands done):
   pick d8776dc add test1.txt
   pick f8e083a modify again test1.txt
```

```
Next command to do (1 remaining command):
   pick d5fb029 modify test1.txt #7
  (use "git rebase --edit-todo" to view and edit)
You are currently rebasing branch 'test2' on 'a3cb030'.
  (fix conflicts and then run "git rebase --continue")
  (use "git rebase --skip" to skip this patch)
  (use "git rebase --abort" to check out the original branch)

Unmerged paths:
  (use "git restore --staged <file>..." to unstage)
  (use "git add <file>..." to mark resolution)
        both modified:   test1.txt

no changes added to commit (use "git add" and/or "git commit -a")

User@Lenovo MINGW64 /d/gitlab (test2|REBASE 2/3)
$ git add .

User@Lenovo MINGW64 /d/gitlab (test2|REBASE 2/3)
$ git rebase --continue
[detached HEAD 1ed00e5] modify again test1.txt
 1 file changed, 3 insertions(+), 1 deletion(-)
Auto-merging test1.txt
CONFLICT (content): Merge conflict in test1.txt
error: could not apply d5fb029... modify test1.txt #7
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".
hint: Disable this message with "git config advice.mergeConflict false"
Could not apply d5fb029... modify test1.txt #7

User@Lenovo MINGW64 /d/gitlab (test2|REBASE 3/3)
$ git add .

User@Lenovo MINGW64 /d/gitlab (test2|REBASE 3/3)
$ git rebase --continue
[detached HEAD 04b0df5] modify test1.txt #7
 1 file changed, 2 insertions(+), 2 deletions(-)
Successfully rebased and updated refs/heads/test2.

User@Lenovo MINGW64 /d/gitlab (test2)
$ git status
On branch test2
nothing to commit, working tree clean

User@Lenovo MINGW64 /d/gitlab (test2)
$ git log
04b0df5 2024-09-20 | modify test1.txt #7 (HEAD -> test2) [feexq]
1ed00e5 2024-09-20 | modify again test1.txt [feexq]
a3cb030 2024-09-20 | modify test1.txt #5 (test3) [feexq]
```

Завдання №3 – Демонстрація роботи cherry-pick:

```
de619bc 2024-09-20 | test 2 commit [feexq]
91e0f54 2024-09-20 | Initial commit (main) [feexq]

User@Lenovo MINGW64 /d/gitlab (test2)
$ git log --all
04b0df5 2024-09-20 | modify test1.txt #7 (HEAD -> test2) [feexq]
1ed00e5 2024-09-20 | modify again test1.txt [feexq]
a3cb030 2024-09-20 | modify test1.txt #5 (test3) [feexq]
de619bc 2024-09-20 | test 2 commit [feexq]
3873181 2024-09-20 | Resolve conflict (test1) [feexq]
27802b7 2024-09-20 | test commit [feexq]
f8e083a 2024-09-20 | modify again test1.txt [feexq]
55daf6b 2024-09-20 | modify test1.txt [feexq]
d8776dc 2024-09-20 | add test1.txt [feexq]
c4d9ae1 2024-09-20 | add test1.txt [feexq]
91e0f54 2024-09-20 | Initial commit (main) [feexq]

User@Lenovo MINGW64 /d/gitlab (test2)
$ git cherry-pick c4d9ae1
Auto-merging test1.txt
On branch test2
You are currently cherry-picking commit c4d9ae1.
  (all conflicts fixed: run "git cherry-pick --continue")
  (use "git cherry-pick --skip" to skip this patch)
  (use "git cherry-pick --abort" to cancel the cherry-pick operation)

nothing to commit, working tree clean
The previous cherry-pick is now empty, possibly due to conflict resolution.
If you wish to commit it anyway, use:

    git commit --allow-empty

Otherwise, please use 'git cherry-pick --skip'

User@Lenovo MINGW64 /d/gitlab (test2|CHERRY-PICKING)
$ git cherry-pick --continue
On branch test2
You are currently cherry-picking commit c4d9ae1.
  (all conflicts fixed: run "git cherry-pick --continue")
  (use "git cherry-pick --skip" to skip this patch)
  (use "git cherry-pick --abort" to cancel the cherry-pick operation)

nothing to commit, working tree clean
The previous cherry-pick is now empty, possibly due to conflict resolution.
If you wish to commit it anyway, use:

    git commit --allow-empty

Otherwise, please use 'git cherry-pick --skip'

User@Lenovo MINGW64 /d/gitlab (test2|CHERRY-PICKING)
```



Висновок: На цій лабораторній роботі було ознайомлення з основними принципами роботи системи контролю версій Git, навчився використовувати базові команди для створення, керування гілками, створення комітів та вирішення конфліктів.

Додаток А: terminal log.txt