

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №1
дисциплины «Основы кроссплатформенного программирования»
Вариант ____

Выполнил:
Эсеналиев Арсен Мурадинович
1 курс, группа ИТС-б-о-21-1,
11.03.02 «Инфокоммуникационные
технологии и системы связи»,
направленность (профиль)
«Инфокоммуникационные системы и
сети», очная форма обучения

(подпись)

Руководитель практики:
Воронкин Роман Александрович

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2021 г.

Конспект теоретического материала:

...

Порядок выполнения работы:

1. Создаю публичный репозиторий с MIT лицензией

2. Создаю 3 файла .txt

Компьютер > Новый том (D:) > LB1.3				
Имя	Дата изменения	Тип	Размер	
1.txt	28.04.2022 21:18	Текстовый докум...	0 КБ	
2.txt	28.04.2022 21:18	Текстовый докум...	0 КБ	
3.txt	28.04.2022 21:18	Текстовый докум...	0 КБ	
LICENSE	28.04.2022 21:17	Файл	2 КБ	
README.md	28.04.2022 21:17	Файл "MD"	1 КБ	

3. Индексируем 1 файл с комитом

```

Git CMD
D:\>cd LB1.3\

D:\LB1.3>git add 1.txt

D:\LB1.3>git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   1.txt

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

D:\LB1.3>git commit -m "add 1.txt file"
[main 2a958a2] add 1.txt file
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 1.txt

D:\LB1.3>

```

4. Повторяем это с 2 и 3 файлами

```

D:\LB1.3>git add 2.txt
D:\LB1.3>git add 3.txt
D:\LB1.3>git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   2.txt
        new file:   3.txt

D:\LB1.3>git commit -m "add 2.txt and 3.txt"

```

5. Создаем ветку с именем my_first_branch.

```

D:\LB1.3>git branch my_first_branch

```

6. Переходим на созданную ветку и добавляем файл in_branch.

```
D:\LB1.3>git branch my_first_branch

D:\LB1.3>git checkout my_first_branch
Switched to branch 'my_first_branch'

D:\LB1.3>git add in_branch.txt

D:\LB1.3>git commit -m "add in_branch.txt"
[my_first_branch 6200e0f] add in_branch.txt
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 in_branch.txt
```

7. Возвращаемся на основную ветку

```
D:\LB1.3>git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 2 commits.
(use "git push" to publish your local commits)
```

8. Создаем и переходим на ветку new_branch

```
D:\LB1.3>git branch new_branch

D:\LB1.3>git checkout new_branch
Switched to branch 'new_branch'
```

9. Добавляем строчку в файл 1.txt и делаем комит

```
D:\LB1.3>git add 1.txt

D:\LB1.3>git commit -m "1.txt внесены изменения"
[new_branch fd9f23f] 1.txt внесены изменения
1 file changed, 1 insertion(+)

D:\LB1.3>_
```

10. Переходим на ветку master и сливаем ветки master и my_first_branch, после чего сливаем ветки master и new_branch.

```
D:\LB1.3>git commit -m "1.txt внесены изменения"
[new_branch fd9f23f] 1.txt внесены изменения
1 file changed, 1 insertion(+)

D:\LB1.3>git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 2 commits.
(use "git push" to publish your local commits)

D:\LB1.3>git merge my_first_branch
Updating 2355589..6200e0f
Fast-forward
 in_branch.txt | 0
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 in_branch.txt

D:\LB1.3>git merge new_branch
Merge made by the 'ort' strategy.
 1.txt | 1 +
1 file changed, 1 insertion(+)

D:\LB1.3>_
```

11. Удаляем ветки new_branch и my_first_branch

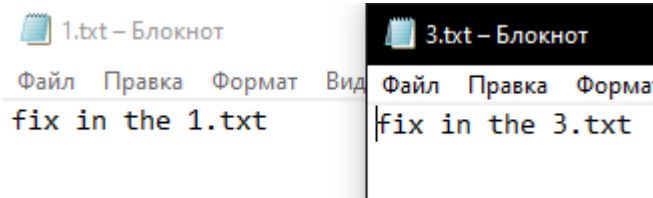
```
D:\LB1.3>git branch -d my_first_branch
Deleted branch my_first_branch (was 6200e0f).

D:\LB1.3>git branch -d new_branch
Deleted branch new_branch (was fd9f23f).
```

12. Создать ветки branch_1 и branch_2.

```
D:\LB1.3>git branch branch_1
D:\LB1.3>git branch branch_2
```

13. Переходим на ветку branch_1 и изменяем файл 1.txt, удаляем все содержимое и добавляем текст “fix in the 1.txt”, изменяем файл 3.txt, удаляем все содержимое и добавляем текст “fix in the 3.txt”, коммитим изменения.



```
D:\LB1.3>git add 1.txt
D:\LB1.3>git add 3.txt

D:\LB1.3>git commit -m "1.txt и 3.txt внесены изменения"
[branch_1 c1dba6b] 1.txt и 3.txt внесены изменения
2 files changed, 2 insertions(+), 1 deletion(-)
```

14. Переходим на ветку branch_2 и также меняем файл 1.txt, удаляем все содержимое и добавляем текст “My fix in the 1.txt”, меняем файл 3.txt, удаляем все содержимое и добавляем текст “My fix in the 3.txt”, коммитим изменения

```
D:\LB1.3>git checkout branch_2
Switched to branch 'branch_2'

D:\LB1.3>git add 1.txt
D:\LB1.3>git add 3.txt

D:\LB1.3>git commit -m "1.txt и 3.txt внесены изменения для ветки branch_2"
[branch_2 f1b3cdc] 1.txt и 3.txt внесены изменения для ветки branch_2
2 files changed, 2 insertions(+), 1 deletion(-)
```

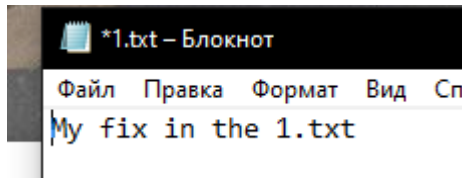
15. D:\LB1.3>

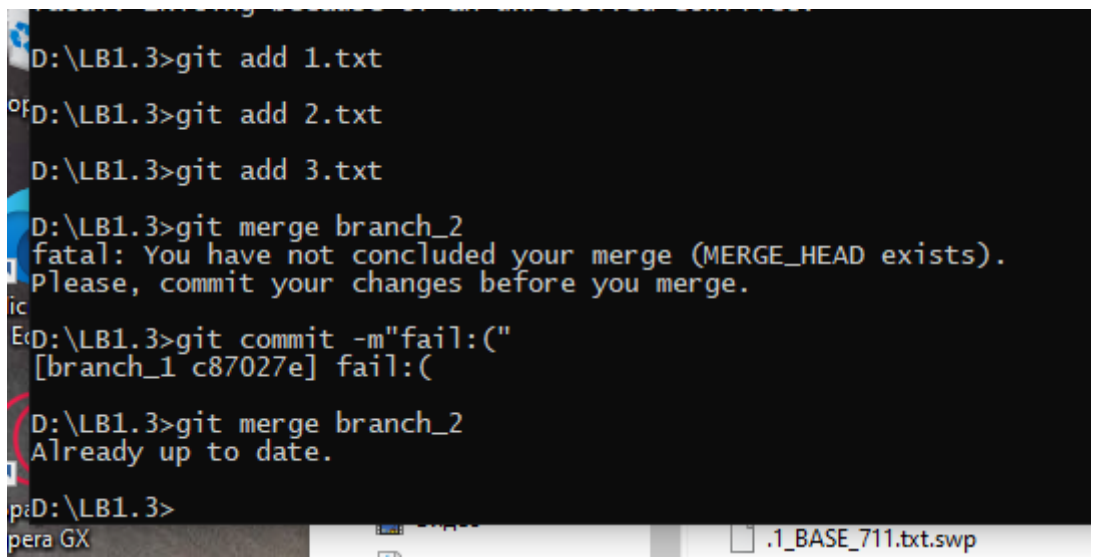
My fix in the 1.txt	My fix in the 3.txt
---------------------	---------------------

16. Слить изменения ветки branch_2 в ветку branch_1.

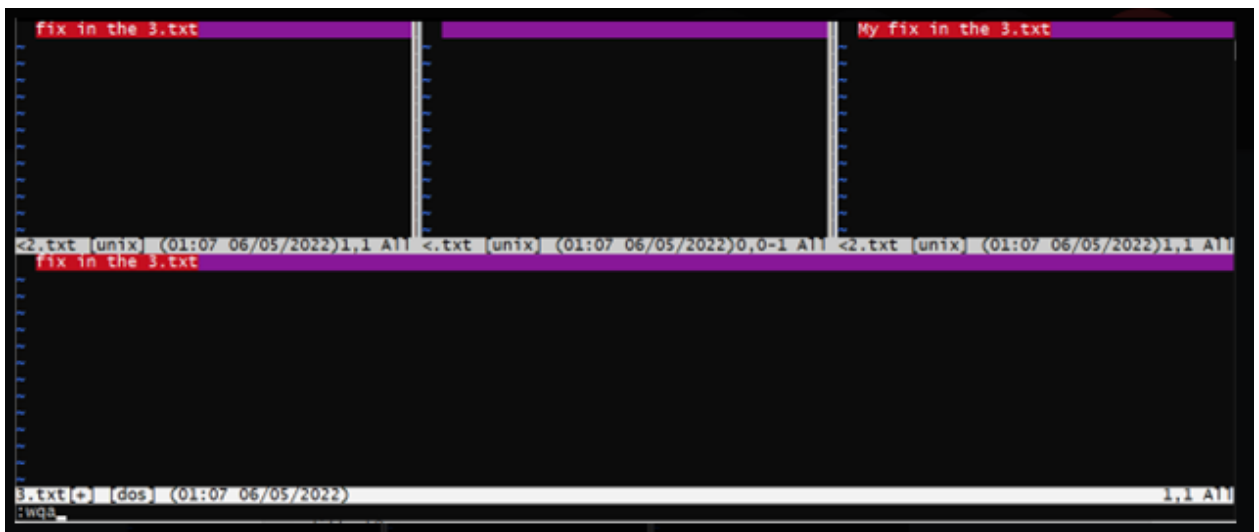
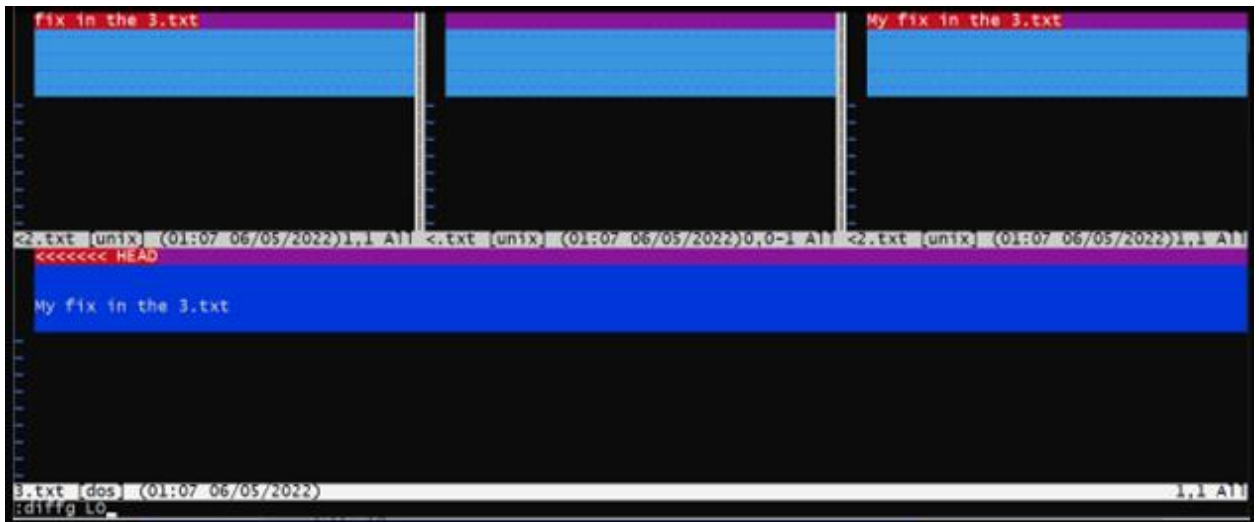
```
D:\LB1.3>git merge branch_2
Auto-merging 1.txt
CONFLICT (content): Merge conflict in 1.txt
Auto-merging 3.txt
CONFLICT (content): Merge conflict in 3.txt
Automatic merge failed; fix conflicts and then commit the result.
D:\LB1.3>
```

17. Решаем конфликт файла 1.txt в ручном режиме, а конфликт 3.txt используя команду git mergetool с помощью одной из доступных утилит, например Meld.

1)  вручную правим

2) 

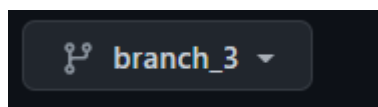
3) Правим через Meld



18. Отправить ветку branch_1 на GitHub.

```
D:\LB1.3>git push --set-upstream origin branch_1
Enumerating objects: 25, done.
Counting objects: 100% (25/25), done.
Delta compression using up to 8 threads
Compressing objects: 100% (16/16), done.
Writing objects: 100% (24/24), 2.10 KiB | 2.10 MiB/s, done.
Total 24 (delta 7), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (7/7), done.
remote:
remote: Create a pull request for 'branch_1' on GitHub by visiting:
remote:   https://github.com/Arsen445/LB1.3/pull/new/branch_1
remote:
To https://github.com/Arsen445/LB1.3.git
 * [new branch]      branch_1 -> branch_1
branch 'branch_1' set up to track 'origin/branch_1'.
```

19. Создать средствами GitHub удаленную ветку branch_3.



20. Создать в локальном репозитории ветку отслеживания удаленной ветки branch_3.

```
D:\LB1.3>git checkout --track origin/branch_3
Switched to a new branch 'branch_3'
branch 'branch_3' set up to track 'origin/branch_3'.

D:\LB1.3>_
```

21.Перейти на ветку branch_3 и добавить в файл файл 2.txt строку "the final fantasy in the 4.txt file".

```
D:\LB1.3>git checkout branch_2
Switched to branch 'branch_2'

D:\LB1.3>git rebase main
Current branch branch_2 is up to date.

D:\LB1.3>git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 5 commits.
(use "git push" to publish your local commits)

D:\LB1.3>git merge branch_2
Updating 5717051..f1b3cdc
Fast-forward
 1.txt | 2 +-
 3.txt | 1 +
 2 files changed, 2 insertions(+), 1 deletion(-)
```

22.Отправить изменения веток master и branch_2 на GitHub.

```
D:\LB1.3>git push
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Arsen445/LB1.3.git
 ca2adec..f1b3cdc main -> main

D:\LB1.3>git checkout main
Already on 'main'
Your branch is up to date with 'origin/main'.

D:\LB1.3>git push
Everything up-to-date
```

Ответы на контрольные вопросы:

1) Что такое ветка?

Это последовательность коммитов, отклоняющаяся от основной линии разработки.

2) Что такое HEAD?

Это указатель, задача которого ссылаться на определённый коммит в репозитории, если точнее: на коммит, который станет родителем для следующего коммита.

3) Способы создания веток?

Новую ветку можно создать командой `git branch <название_ветки>` или на удаленном репозитории `git hub`.

4) Как узнать текущую ветку?

При помощи команды `git branch`.

5) Как переключаться между ветками?

При помощи команды `git checkout <название_ветки>`.

6) Что такое удаленная ветка?

Это ветка, находящаяся на удаленном репозитории. Или ссылка на состояние ветки на удаленном репозитории.

7) Что такое ветка отслеживания?

Это ветка в локальной репозитории, которая напрямую связана с удаленной веткой на удаленном репозитории.

8) Как создать ветку отслеживания?

Командой `git checkout --track origin/<название_ветки>`.

9) Как отправить изменения из локальной ветки в удаленную ветку?

Командой `git push origin <название_ветки>`.

10) В чем отличие команд `git fetch` и `git pull`?

`Git pull` – это сочетание команд `git fetch` (получение изменений с удаленного репозитория) и `git merge` (объединение веток).

11) Как удалить локальную и удаленную ветки?

Используя команду `git branch -d <название_ветки>`. Для удаление удаленной ветки существует команда `git push origin -d <название_ветки>`.