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 РАБОТЕ №1.3.
дисциплины «Основы кроссплатформенного программирования»

Выполнила:
Дудова Мира Сергеевна
1 курс, группа ИТС-б-о-22-1,
11.03.02 «Инфокоммуникационные технологии и системы связи»,
направленность (профиль) «Инфокоммуникационные системы и сети», очная форма обучения

_____
(подпись)

Руководитель практики:
Воронкин Р.А., канд. тех. наук, доцент,
доцент кафедры инфокоммуникаций

_____
(подпись)

Отчет защищен с оценкой_____Дата защиты_____

Ставрополь, 2023 г.

Тема: основы ветвления Git.

Цель работы: исследовать базовые возможности по работе с локальными и удаленными ветками Git.

Порядок выполнения работы:

1. Создала общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT.

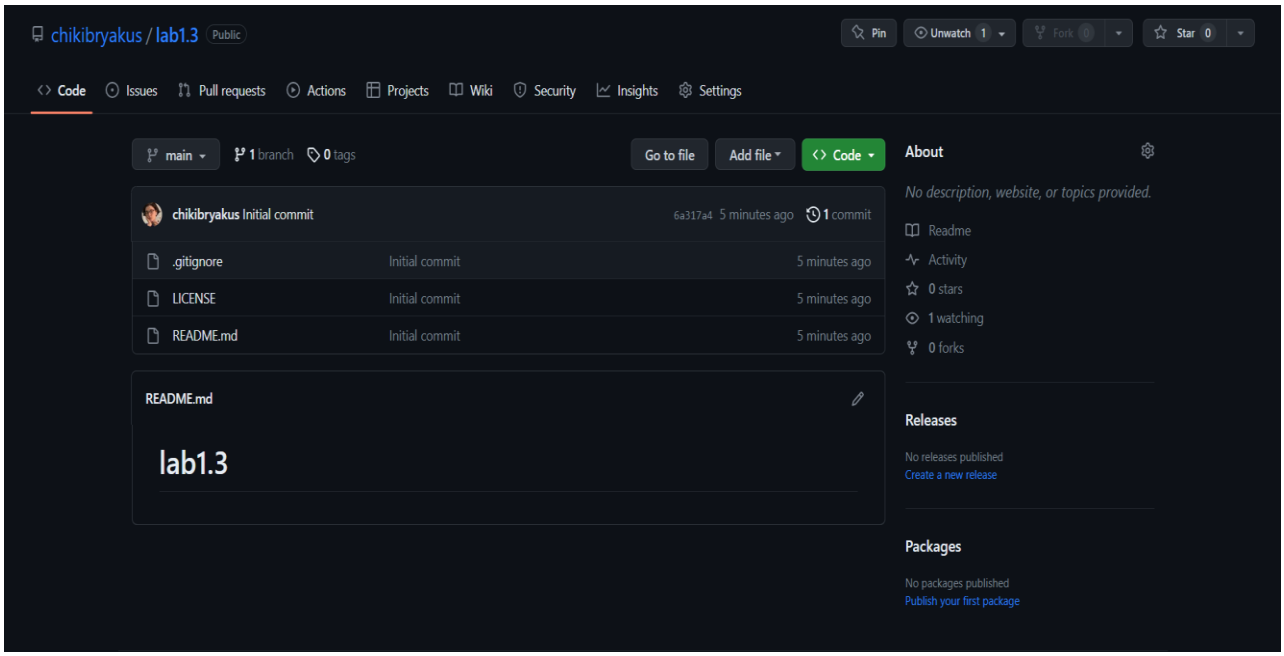

Рис. 1. Новый репозиторий.

2. Проклонировала репозиторий на свой компьютер.

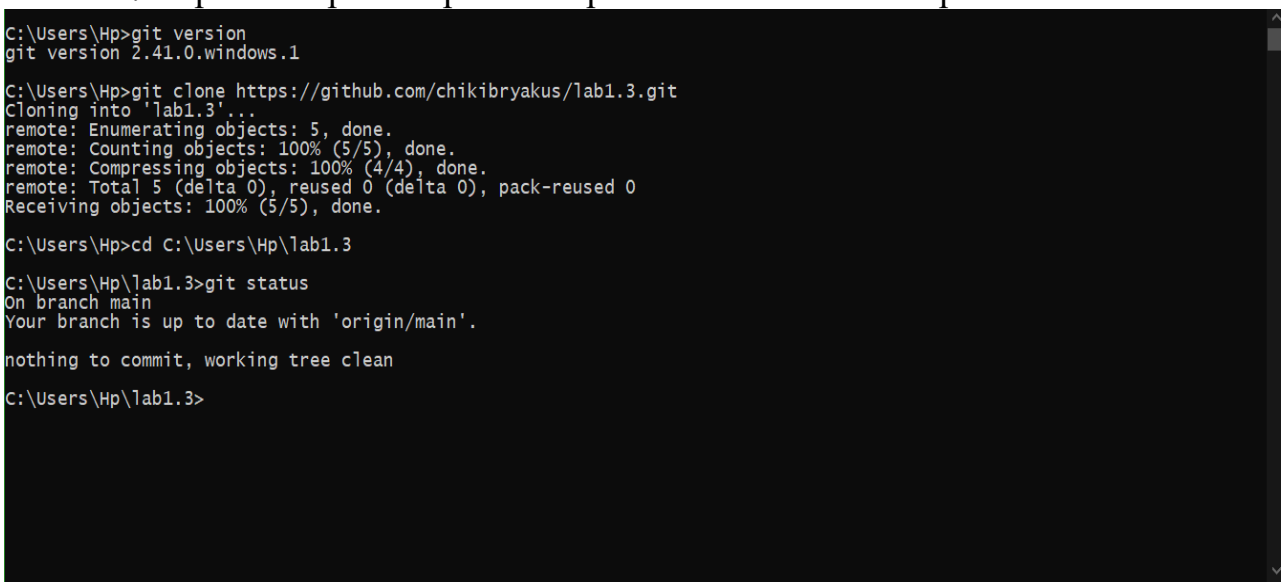

Рис. 2. Клонирование.

3. Добавила 3 новых текстовых файла.

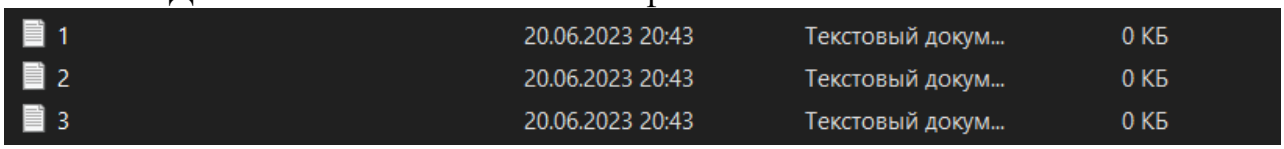

Рис. 3. Новые текстовые файлы.

4. Проиндексировал первый файл и сделать коммит.

```
C:\Users\Hp\lab1.3>git add 1.txt

C:\Users\Hp\lab1.3>git commit -m "1.txt file"
[main b778c06] 1.txt file
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 1.txt

C:\Users\Hp\lab1.3>
```

Рис. 4. Добавление изменений и их фиксация.

5. Проиндексировала второй и третий файлы.

.

```
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
```

Рис. 5. Добавление изменений

6. Перезаписала уже сделанный коммит.

Рис. 6. Редактирование существующего коммита.

```
 3 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 1.txt
 create mode 100644 2.txt
 create mode 100644 3.txt
```

7. Создание новой ветки и переход на нее и создать новый файл in_branch.txt.

```
C:\Users\Hp\lab1.3>git branch my_first_branch

C:\Users\Hp\lab1.3>git checkout my_first_branch
Switched to branch 'my_first_branch'

C:\Users\Hp\lab1.3>git status
On branch my_first_branch
nothing to commit, working tree clean

C:\Users\Hp\lab1.3>git status
On branch my_first_branch
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        in_branch.txt

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

C:\Users\Hp\lab1.3>git add .

C:\Users\Hp\lab1.3>git commit -m "in_branch"
[my_first_branch 419288a] in_branch
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 in_branch.txt

C:\Users\Hp\lab1.3>
```

Рис. 8. Создание файла и фиксация.

8. Перешла вновь на новую ветку main и сразу перешла на ветку new_branch.

```
C:\Users\Hp\lab1.3>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

C:\Users\Hp\lab1.3>git checkout -b new_branch
Switched to a new branch 'new_branch'

C:\Users\Hp\lab1.3>
```

Рис. 10. Создание и сразу переход на ветку.

9. Сделала изменения в файле 1.txt, добавила строчку “new row in the 1.txtfile”, закоммитила изменения.

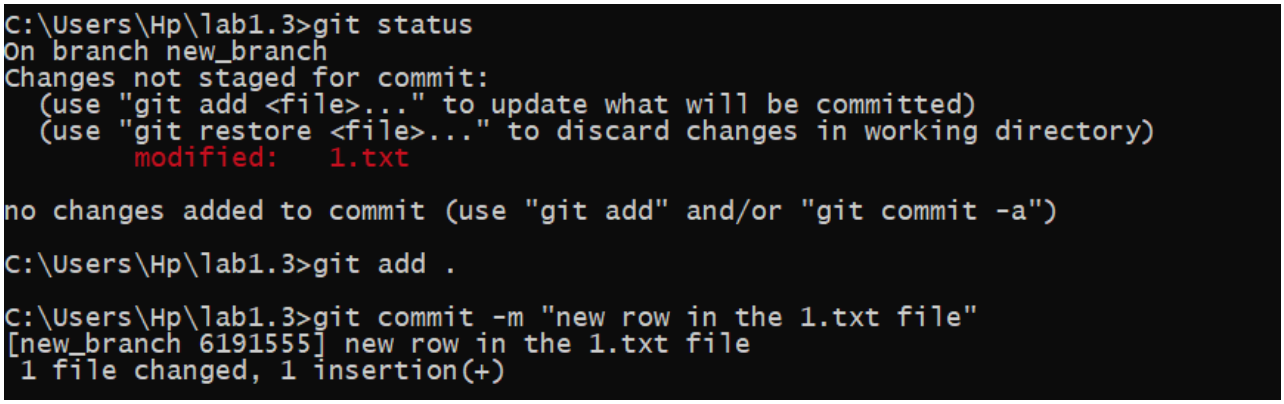

Рис. 11. Изменение в 1.txt фиксация этих изменений.

10. Перешла на ветку main, слила ветки main и my_first_branch, после слила ветки main и new_branch.

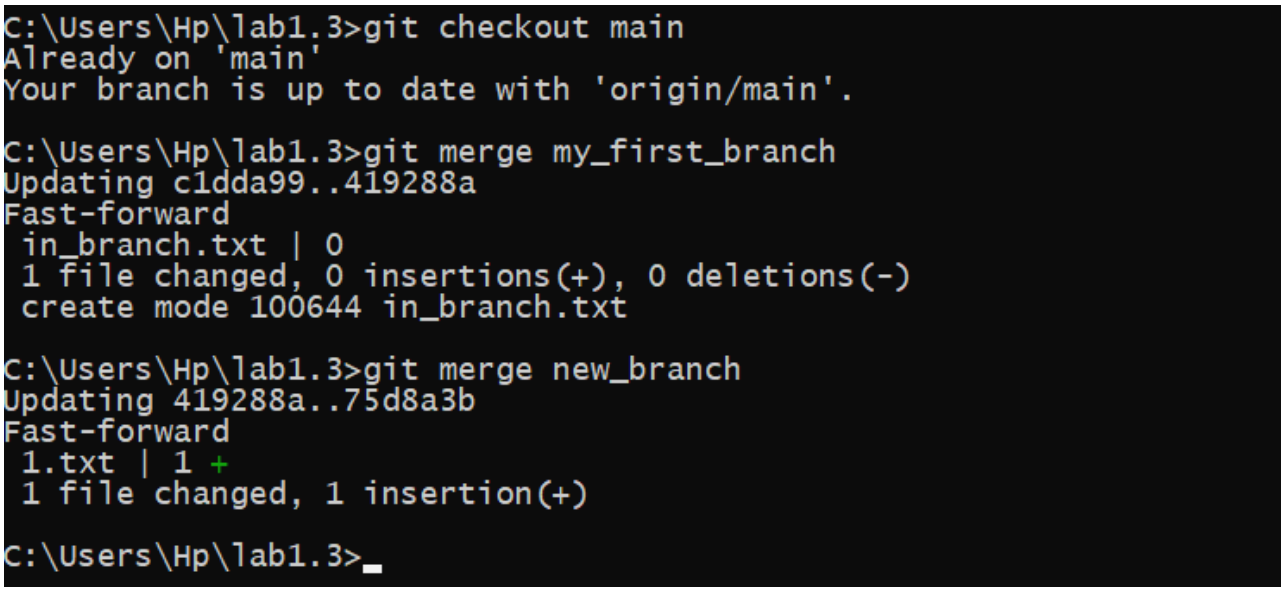

Рис. 12. Слияние веток.

11. Удалила ветки my_first_branch и new_branch.

```
C:\Users\Hp\lab1.3>git branch -d my_first_branch
Deleted branch my_first_branch (was 419288a).

C:\Users\Hp\lab1.3>git branch -d new_branch
Deleted branch new_branch (was 75d8a3b).

C:\Users\Hp\lab1.3>
```

Рис. 13. Удаление веток.

12. Создала ветки branch_1 и branch_2.

```
C:\Users\Hp\lab1.3>git branch branch_1

C:\Users\Hp\lab1.3>git branch branch_2

C:\Users\Hp\lab1.3>git branch
  branch_1
  branch_2
* main
```

Рис. 14. Создание новых веток.

13. Перешла на ветку branch_1 и изменила файл 1.txt, удалил все содержимое и добавила текст “fix in the 1.txt”, изменила файл 3.txt, удалила все содержимое и добавила текст “fix in the 3.txt”, закоммитила изменения.

```
C:\Users\Hp\lab1.3>git checkout branch_1
Switched to branch 'branch_1'

C:\Users\Hp\lab1.3>git add .

C:\Users\Hp\lab1.3>git commit -m "changes in files 1 and 3"
[branch_1 9ac9395] changes in files 1 and 3
 2 files changed, 2 insertions(+), 1 deletion(-)

C:\Users\Hp\lab1.3>
```

Рис. 15. Изменения в первой ветке и во второй ветка и фиксация изменений.

14. Перешла на ветку branch_2 и также изменила файл 1.txt, удалила все содержимое и добавила текст “My fix in the 1.txt”, изменила файл 3.txt, удалила все содержимое и добавила текст “My fix in the 3.txt”, закоммитила изменения.

```
C:\Users\Hp\lab1.3>git status
On branch branch_2
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:   1.txt
        modified:   3.txt

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

C:\Users\Hp\lab1.3>git add .

C:\Users\Hp\lab1.3>git commit -m "changes in files 1 and 3"
[branch_2 4d76ade] changes in files 1 and 3
 2 files changed, 2 insertions(+), 1 deletion(-)
```

Рис. 16. Переход на вторую ветку.

15. Слила изменения ветки branch_2 в ветку branch_1.

```
C:\Users\Hp\lab1.3>git checkout branch_1
Switched to branch 'branch_1'

C:\Users\Hp\lab1.3>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.

C:\Users\Hp\lab1.3>
```

Рис. 17. Слияние веток.

16. Решила конфликт файла 1.txt и 2.txt в ручном режиме.

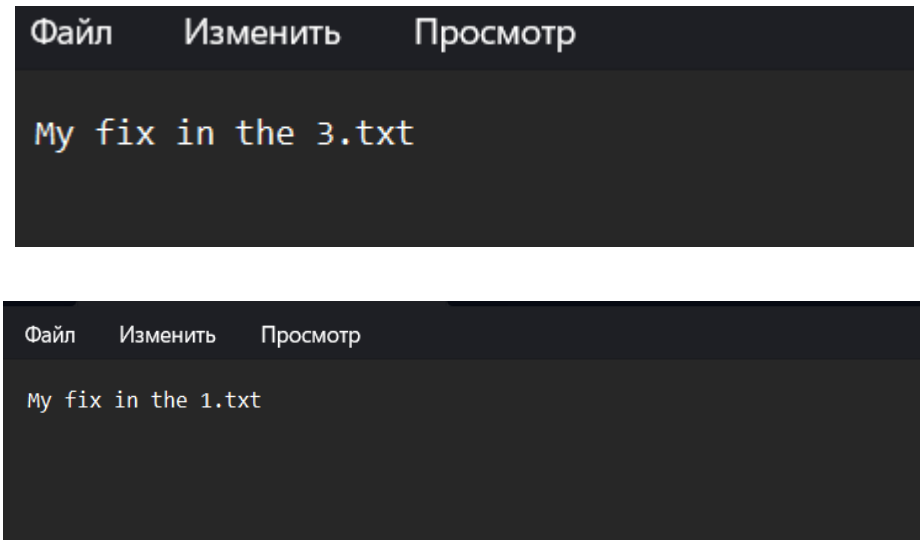

Рис. 18. Решение конфликта вручную.

Рис. 19. Решение конфликта вручную.

17. Отправила branch_1 на удаленный git push origin branch_1.

```
C:\Users\Hp\lab1.3>git commit -m "conflict resolution"
[branch_1 ca08c71] conflict resolution

C:\Users\Hp\lab1.3>git status
On branch branch_1
nothing to commit, working tree clean

C:\Users\Hp\lab1.3>git push origin branch_1
Enumerating objects: 21, done.
Counting objects: 100% (20/20), done.
Delta compression using up to 8 threads
Compressing objects: 100% (13/13), done.
Writing objects: 100% (18/18), 1.55 KiB | 1.55 MiB/s, done.
Total 18 (delta 8), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (8/8), completed with 1 local object.
remote:
remote: Create a pull request for 'branch_1' on GitHub by visiting:
remote:      https://github.com/chikibryakus/lab1.3/pull/new/branch_1
remote:
To https://github.com/chikibryakus/lab1.3.git
 * [new branch]      branch_1 -> branch_1

C:\Users\Hp\lab1.3>
```

Рис. 20. Отправление ветки на удаленный сервер.

18. Создала средствами GitHub удаленную ветку branch_3.

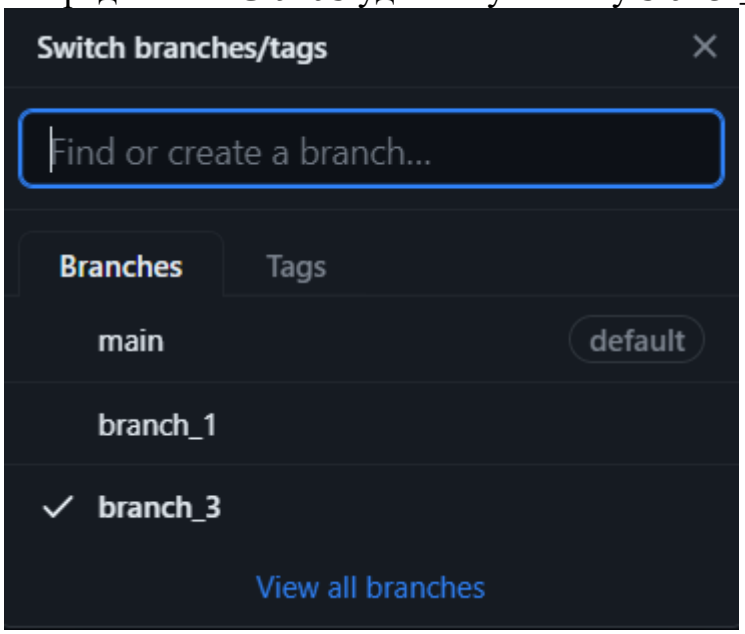

Рис. 21. Создание ветки на GitHub.

19. Создала в локальном репозитории ветку отслеживания удаленной ветки branch_3.

```
C:\Users\Hp\lab1.3>git fetch origin
From https://github.com/chikibryakus/lab1.3
 * [new branch]      branch_3   -> origin/branch_3

C:\Users\Hp\lab1.3>git checkout -b branch_3 origin/branch_3
Switched to a new branch 'branch_3'
branch 'branch_3' set up to track 'origin/branch_3'.
```

Рис. 22. Создание ветки отслеживания.

20. Перешла на ветку branch_3 и добавила файл файл 2.txt строку "thefinal fantasy in the 4.txt file", выполнила перемещение ветки main на ветку branch_2

```
C:\Users\Hp\lab1.3>git status
On branch branch_3
Your branch is up to date with 'origin/branch_3'.

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:   2.txt

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

C:\Users\Hp\lab1.3>git add .

C:\Users\Hp\lab1.3>git commit -m "the final fantasy in the 4.txt file"
[branch_3 a129826] the final fantasy in the 4.txt file
 1 file changed, 1 insertion(+)

C:\Users\Hp\lab1.3>git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 3 commits.
  (use "git push" to publish your local commits)

C:\Users\Hp\lab1.3>git rebase branch_2
Successfully rebased and updated refs/heads/main.

C:\Users\Hp\lab1.3>git checkout branch_2
Switched to branch 'branch_2'

C:\Users\Hp\lab1.3>git merge mainm
merge: mainm - not something we can merge

C:\Users\Hp\lab1.3>git merge main
Already up to date.
```

Рис. 23. Перемещение веток.

21. Отправил изменения веток master и branch_2 на удаленный сервер.

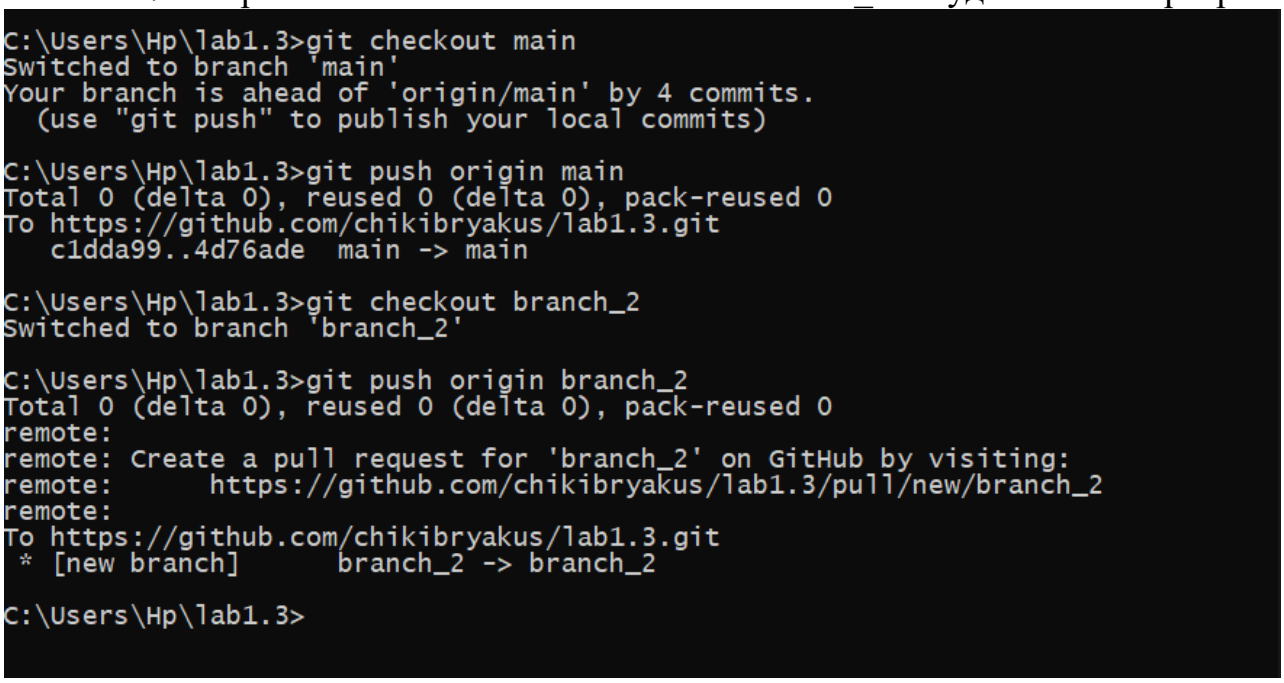

Рис. 24. Отправка изменения веток на GitHub.

Ссылка: https://github.com/chikibryakus/lab1.3

Ответы на контрольные вопросы:

1. Что такое ветка?

Почти каждая система контроля версий (СКВ) в какой-то форме поддерживает ветвление. Используя ветвление, Вы отклоняетесь от основной линии разработки и продолжаете работу независимо от неё, не вмешиваясь в основную линию.

2. Что такое HEAD?

HEAD в Git – это указатель на текущую ссылку ветви, которая, в свою очередь, является указателем на последний сделанный вами коммит или последний коммит, который был извлечен из вашего рабочего каталога.

3. Способы создания веток.

Создать ветку можно с помощью двух команд. Команда, которая просто создает ветку: git branch “name_branch”.

Команда, которая создает ветку и сразу же к ней переходит: git checkout –b “name_branch”.

4. Как узнать текущую ветку?

Текущую ветку можно узнать с помощью команды: git branch.

5. Как переключаться между ветками?

Между ветками можно переключаться с помощью команды: git checkout “name_branch”.

6. Что такое удаленная ветка?

Удаленные ветки - это ссылки на определённое состояние удалённых веток. Это локальные ветки, которые нельзя перемещать.

7. Что такое ветка отслеживания?

Отслеживаемые ветки — это локальные ветки, которые напрямую связаны с удалённой веткой.

Если, находясь на отслеживаемой ветке, вы наберёте git push, Git уже будет знать, на какой сервер и в какую ветку отправлять изменения.

8. Как создать ветку отслеживания?

Ветку отслеживания можно создать с помощью команды: git checkout --track origin/<name_branch>.

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

Отправить изменения из локальной ветки в удаленную можно с помощью команды: git push origin <name_branch>.

10. В чем отличие команд git fetch и git pull ?

Команда git fetch получает с сервера все изменения, которых у вас ещё нет, но не будет изменять состояние вашей рабочей директории. Эта команда просто получает данные и позволяет вам самостоятельно сделать слияние. Тем не менее, существует команда git pull , которая в большинстве случаев является командой git fetch , за которой непосредственно следует команда git merge.

11. Как удалить локальную и удаленную ветки?

Для удаление локальной ветки используется команда: git branch –d <name_branch>

Для удаления удаленной ветки используется команда: git push –delete origin/<name_branch>

12. Какие основные типы веток присутствуют в модели git-flow? Как организованна работа с ветками в модели git-flow? В чем недостатки git-flow?

Существуют следующие типы ветвей:

1) ветви функциональностей;
2) ветви релизов;
3) ветви исправлений.

Ветви функциональностей (feature branches), также называемые иногда тематическими ветвями (topic branches), используются для разработки новых функций, которые должны появиться в текущем или будущем релизах.

Ветви релизов (release branches) используются для подготовки к выпуску новых версий продукта. Они позволяют расставить финальные точки над i перед выпуском новой версии.

Ветви для исправлений (hotfix branches) весьма похожи на ветви релизов (release branches), так как они тоже используются для подготовки новых выпусков продукта, разве лишь незапланированных.

Недостатки git flow: авторам приходится использовать ветку develop вместо master, поскольку master зарезервирован для кода.

Вторая проблема процесса git flow – сложности, возникающие из-за веток для патчей и для релиза. Подобная структура может подойти некоторым организациям, но для абсолютного большинства она просто убийственно излишяя.