

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

Институт ИМИТ
Кафедра Инфокоммуникаций

Дисциплина Языки программирования
ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ
Основы ветвления Git

Выполнил:

Боржонов Ростислав Александрович
студент 2 курса, 1 группы
специальности ИТС-б-о-20-1
очной формы обучения

Дата защиты

«___» _____ 20__ г.

Оценка _____

Проверил:

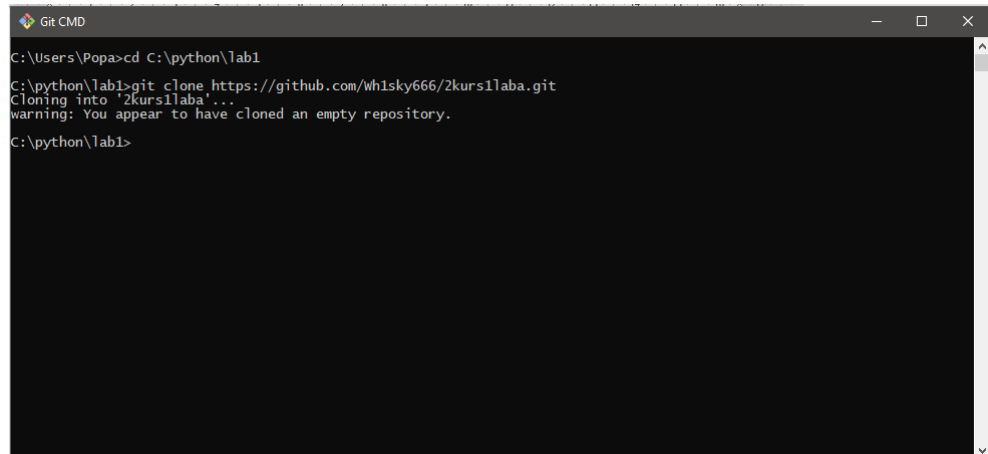
Кандидат технических наук,
Доцент кафедры
инфокоммуникаций.
[Воронкин Роман Александрович](#)

(Подпись)

Ставрополь, 2021г.

Цель работы: исследование базовых возможностей по работе с локальными и удаленными ветками Git.

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



```
Git CMD
C:\Users\Popa>cd C:\python\lab1
C:\python\lab1>git clone https://github.com/whisky666/2kurs1laba.git
Cloning into '2kurs1laba'...
warning: You appear to have cloned an empty repository.
C:\python\lab1>
```

Рисунок 1. Клонирование репозитория.

Создал три файла: 1.txt, 2.txt, 3.txt.

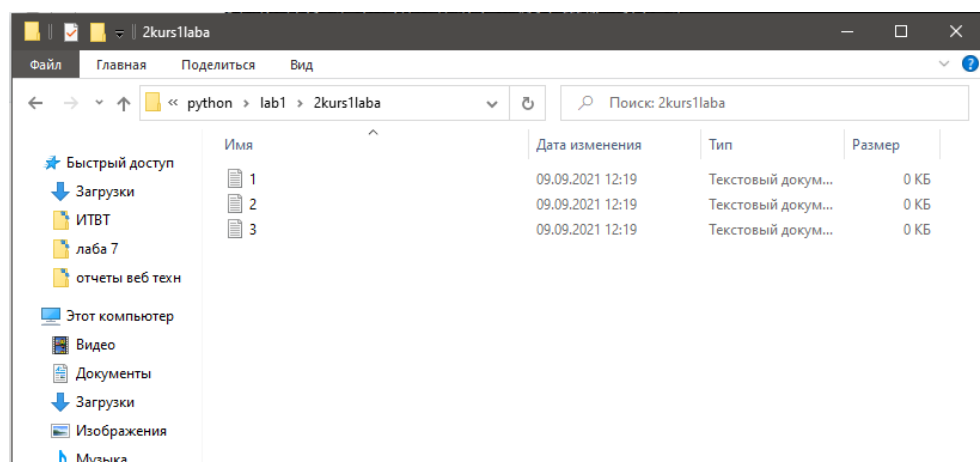
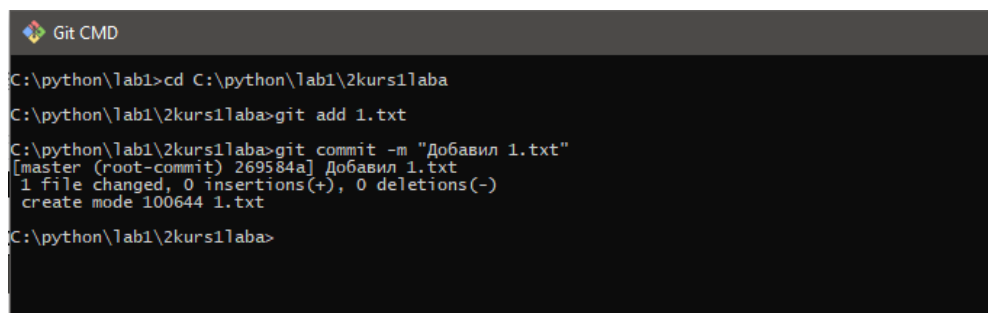


Рисунок 2. Папка с созданными тремя текстовыми документами.

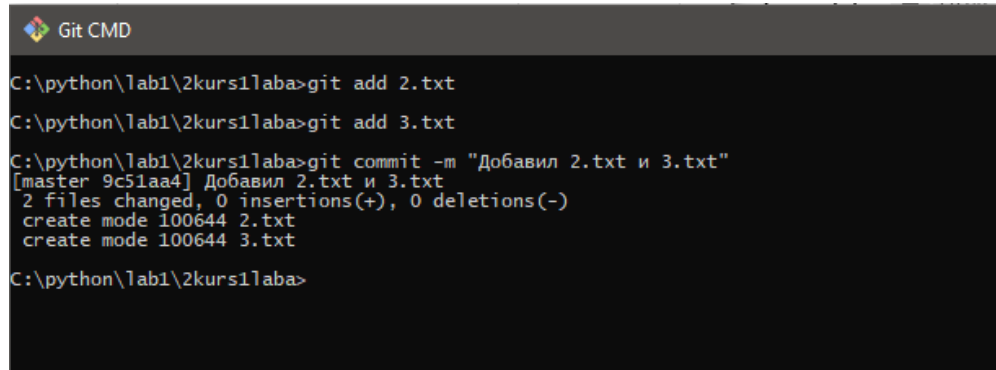
Проиндексировал первый файл и сделать коммит с комментарием "add 1.txt file".



```
Git CMD
C:\python\lab1>cd C:\python\lab1\2kurs1laba
C:\python\lab1\2kurs1laba>git add 1.txt
C:\python\lab1\2kurs1laba>git commit -m "Добавил 1.txt"
[master (root-commit) 269584a] Добавил 1.txt
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 1.txt
C:\python\lab1\2kurs1laba>
```

Рисунок 3. Индексация 1ого файла.

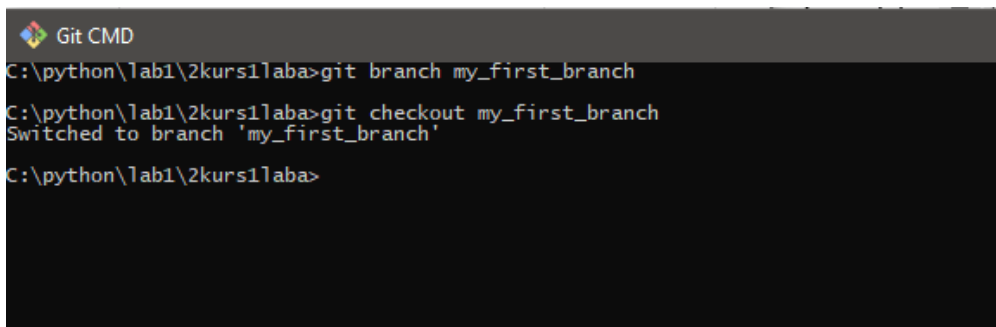
Проиндексировал второй и третий файлы.



```
Git CMD
C:\python\lab1\2kurs11aba>git add 2.txt
C:\python\lab1\2kurs11aba>git add 3.txt
C:\python\lab1\2kurs11aba>git commit -m "Добавил 2.txt и 3.txt"
[master 9c51aa4] Добавил 2.txt и 3.txt
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 2.txt
 create mode 100644 3.txt
C:\python\lab1\2kurs11aba>
```

Рисунок 4. Индексация 2ого и 3его файла.

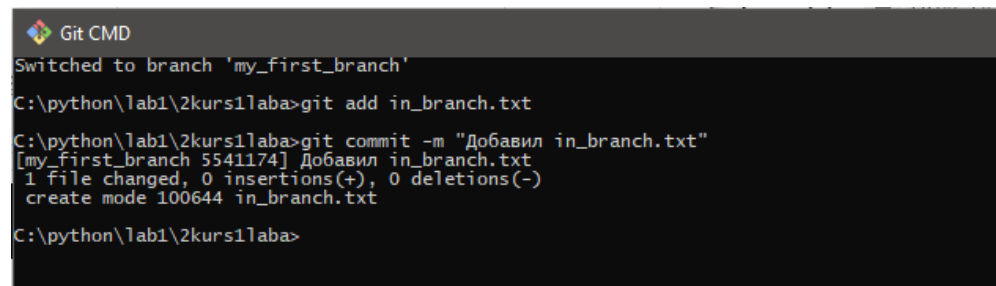
Создал новую ветку my_first_branch.



```
Git CMD
C:\python\lab1\2kurs11aba>git branch my_first_branch
C:\python\lab1\2kurs11aba>git checkout my_first_branch
Switched to branch 'my_first_branch'
C:\python\lab1\2kurs11aba>
```

Рисунок 5. Создание новой ветки.

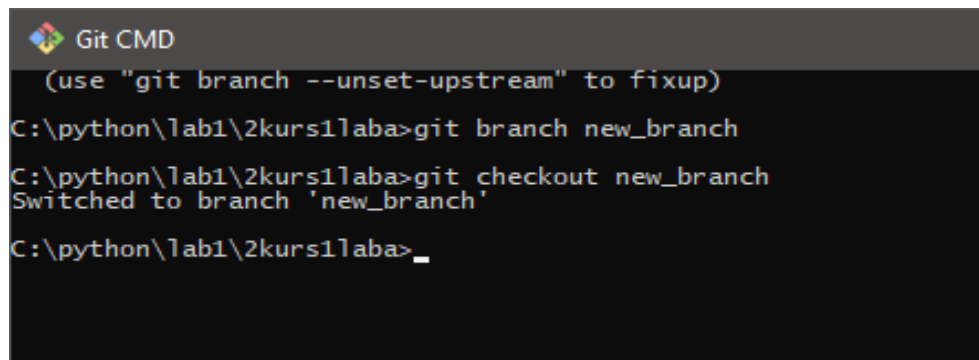
Перешел на ветку и создал новый файл in_branch.txt, закоммитил изменения.



```
Git CMD
Switched to branch 'my_first_branch'
C:\python\lab1\2kurs11aba>git add in_branch.txt
C:\python\lab1\2kurs11aba>git commit -m "Добавил in_branch.txt"
[my_first_branch 5541174] Добавил in_branch.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 in_branch.txt
C:\python\lab1\2kurs11aba>
```

Рисунок 6. Коммит изменений.

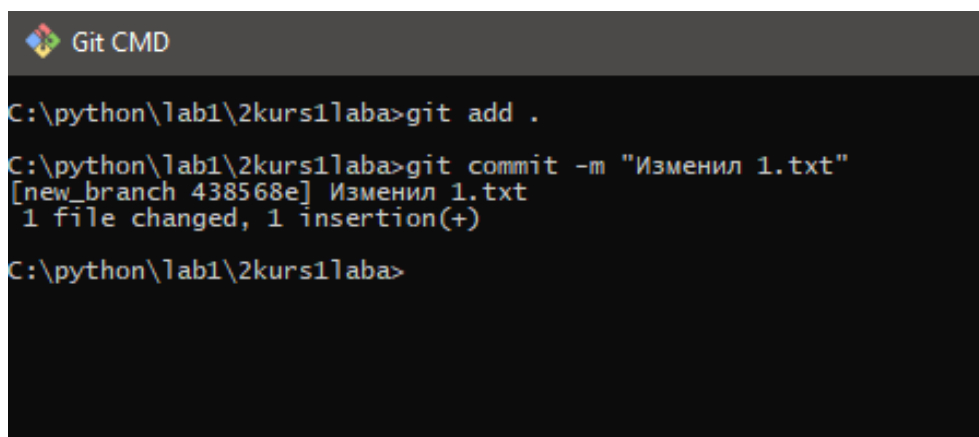
Создал и сразу перешел на ветку new_branch.



```
Git CMD
(use "git branch --unset-upstream" to fixup)
C:\python\lab1\2kurs11aba>git branch new_branch
C:\python\lab1\2kurs11aba>git checkout new_branch
Switched to branch 'new_branch'
C:\python\lab1\2kurs11aba>_
```

Рисунок 7. Создал новую ветку.

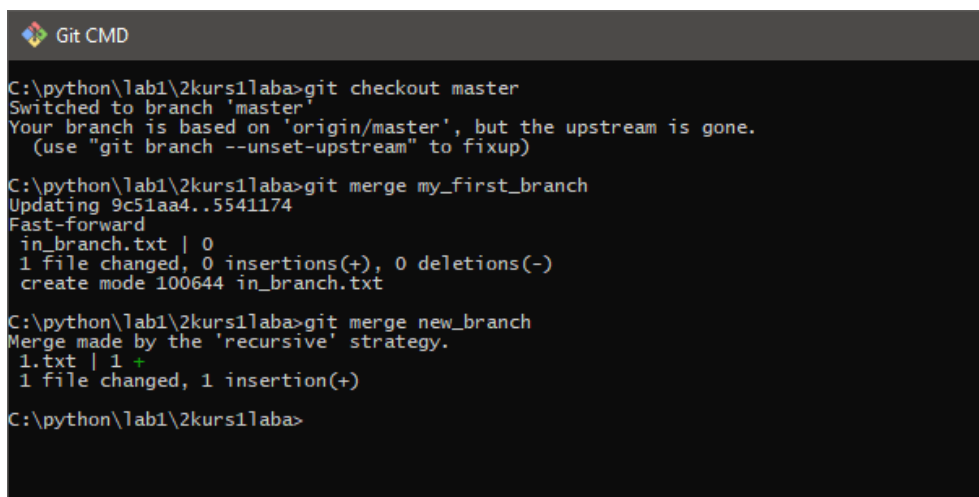
Сделал изменения в файле 1.txt, добавить строчку “new row in the 1.txt file”, закоммитил изменения.



```
Git CMD
C:\python\lab1\2kurs11aba>git add .
C:\python\lab1\2kurs11aba>git commit -m "Изменил 1.txt"
[new_branch 438568e] Изменил 1.txt
1 file changed, 1 insertion(+)
C:\python\lab1\2kurs11aba>
```

Рисунок 8. Коммит изменений.

Перешел на ветку master и слил ветки master и my_first_branch, после чего слил ветки master и new_branch.



```
Git CMD
C:\python\lab1\2kurs11aba>git checkout master
Switched to branch 'master'
Your branch is based on 'origin/master', but the upstream is gone.
(use "git branch --unset-upstream" to fixup)
C:\python\lab1\2kurs11aba>git merge my_first_branch
Updating 9c51aa4..5541174
Fast-forward
 in_branch.txt | 0
1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 in_branch.txt
C:\python\lab1\2kurs11aba>git merge new_branch
Merge made by the 'recursive' strategy.
1.txt | 1 +
1 file changed, 1 insertion(+)
C:\python\lab1\2kurs11aba>
```

Рисунок 9. Слияние веток main, my_first_branch и new_branch.

Удалил ветки my_first_branch и new_branch. Создал ветки branch_1 и branch_2.

```
Git CMD
C:\python\lab1\2kurs1laba>git branch branch_1
C:\python\lab1\2kurs1laba>git branch branch_2
C:\python\lab1\2kurs1laba>git branch
  branch_1
  branch_2
* master
  my_first_branch
  new_branch

C:\python\lab1\2kurs1laba>git branch -d my_first_branch
Deleted branch my_first_branch (was 5541174).

C:\python\lab1\2kurs1laba>git branch -d new_branch
Deleted branch new_branch (was 438568e).

C:\python\lab1\2kurs1laba>git branch
  branch_1
  branch_2
* master

C:\python\lab1\2kurs1laba>
```

Рисунок 10. Удаление веток my_first_branch и new_branch, и создание my_first_branch и new_branch.

Перешел на ветку branch_1 и изменил файл 1.txt, удалил все содержимое и добавил текст “fix in the 1.txt”, изменил файл 3.txt, удалил все содержимое и добавил текст “fix in the 3.txt”, закоммитил изменения.

```
Git CMD
Switched to branch 'branch_1'
C:\python\lab1\2kurs1laba>git add .
C:\python\lab1\2kurs1laba>git commit -m "fix in the 1.txt"
[branch_1 cc2ec1a] fix in the 1.txt
 2 files changed, 2 insertions(+), 1 deletion(-)

C:\python\lab1\2kurs1laba>_
```

Рисунок 11. Коммит изменений.

Перейти на ветку branch_2 и также изменил файл 1.txt, удалил все содержимое и добавил текст “My fix in the 1.txt”, изменить файл 3.txt, удалить все содержимое и добавить текст “My fix in the 3.txt”, закоммитить изменения.

```
Git CMD
Switched to branch 'branch_2'
C:\python\lab1\2kurs11aba>git add .
C:\python\lab1\2kurs11aba>git commit -m "fix in the 1.txt & 3.txt"
"git" не является внутренней или внешней
командой, исполняемой программой или пакетным файлом.
C:\python\lab1\2kurs11aba>git commit -m "fix in the 1.txt & 3.txt"
[branch_2 3e05cd9] fix in the 1.txt & 3.txt
2 files changed, 2 insertions(+), 1 deletion(-)
C:\python\lab1\2kurs11aba>_
```

Рисунок 12. Коммит изменений.

Слил изменения ветки branch_2 в ветку branch_1.

```
Git CMD
C:\python\lab1\2kurs11aba>git checkout branch_1
Switched to branch 'branch_1'
C:\python\lab1\2kurs11aba>git merge branch_2
Auto-merging 3.txt
CONFLICT (content): Merge conflict in 3.txt
Auto-merging 1.txt
CONFLICT (content): Merge conflict in 1.txt
Automatic merge failed; fix conflicts and then commit the result.
C:\python\lab1\2kurs11aba>_
```

Рисунок 13. Слияние ветки branch_2 в ветку branch_1

Решил конфликт файла 1.txt в ручном режиме

```
Git CMD
fatal: Exiting because of an unresolved conflict.
C:\python\lab1\2kurs11aba>git merge branch_2
error: Merging is not possible because you have unmerged files.
hint: Fix them up in the work tree, and then use 'git add/rm <file>'
hint: as appropriate to mark resolution and make a commit.
fatal: Exiting because of an unresolved conflict.
C:\python\lab1\2kurs11aba>git status
On branch branch_1
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 modified:   1.txt
        both modified:   3.txt

no changes added to commit (use "git add" and/or "git commit -a")
C:\python\lab1\2kurs11aba>git add .
C:\python\lab1\2kurs11aba>git commit -m "Решение конфликтов"
[branch_1 ad9f057] Решение конфликтов
C:\python\lab1\2kurs11aba>git merge branch_2
Already up to date.
C:\python\lab1\2kurs11aba>_
```

Рисунок 14. Решение конфликта файлов.

Отправить ветку branch_1 на GitHub.

```
Git CMD
C:\python\lab1\2kurs1laba>git push origin branch_1
Enumerating objects: 24, done.
Counting objects: 100% (24/24), done.
Delta compression using up to 2 threads
Compressing objects: 100% (15/15), done.
Writing objects: 100% (24/24), 2.06 KiB | 421.00 KiB/s, done.
Total 24 (delta 4), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (4/4), done.
To https://github.com/Whisky666/2kurs1laba.git
 * [new branch]      branch_1 -> branch_1

C:\python\lab1\2kurs1laba>
```

Рисунок 15. Отправление ветки branch_1 на GitHub.

Создал средствами GitHub удаленную ветку branch_3.

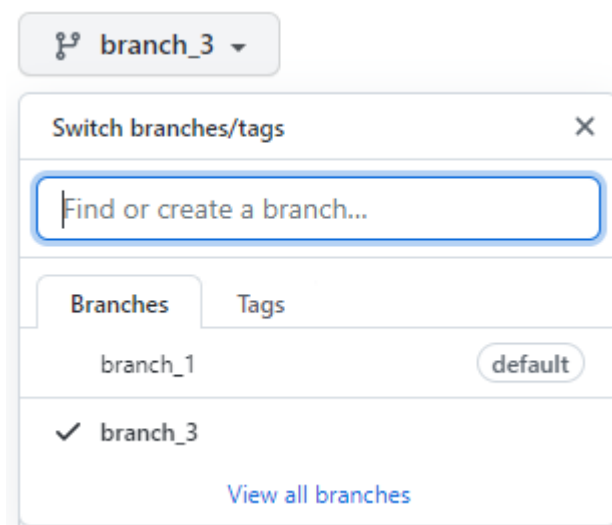


Рисунок 16. Создание удаленной ветки branch_3.

Создал в локальном репозитории ветку отслеживания удаленной ветки branch_3.

```
Git CMD
C:\python\lab1\2kurs1laba>git branch
* branch_1
  master

C:\python\lab1\2kurs1laba>git fetch origin
From https://github.com/Whisky666/2kurs1laba
 * [new branch]      branch_3 -> origin/branch_3

C:\python\lab1\2kurs1laba>git checkout -b branch_3 origin/branch_3
Switched to a new branch 'branch_3'
Branch 'branch_3' set up to track remote branch 'branch_3' from 'origin'.

C:\python\lab1\2kurs1laba>
```

Рисунок 17. Сделал отслеживание branch_3 в локальном репозитории.

Перешел на ветку branch_3 и добавил файл файл 2.txt строку "the final fantasy in the 4.txt file". Выполнил перемещение ветки master на ветку branch_2. Отправил изменения веток master и branch_2 на GitHub.

```
Git CMD
Switched to a new branch 'branch_3'
Branch 'branch_3' set up to track remote branch 'branch_3' from 'origin'.

C:\python\lab1\2kurs11aba>git add .

C:\python\lab1\2kurs11aba>git commit -m "Изменил 2.txt"
[branch_3 76e53d2] Изменил 2.txt
1 file changed, 1 insertion(+)

C:\python\lab1\2kurs11aba>git push origin master
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote: Create a pull request for 'master' on GitHub by visiting:
remote:   https://github.com/Whisky666/2kurs11aba/pull/new/master
remote:
To https://github.com/Whisky666/2kurs11aba.git
 * [new branch]      master -> master

C:\python\lab1\2kurs11aba>git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 2 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 377 bytes | 377.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Whisky666/2kurs11aba.git
 ad9f057..76e53d2  branch_3 -> branch_3

C:\python\lab1\2kurs11aba>git push origin branch_3
Everything up-to-date
```

Рисунок 18. Коммит изменений во втором файле. Отправление веток master и branch_2 на GitHub.

Ответы на контрольные вопросы:

1. Что такое ветка?

Ветка в Git — это простой перемещаемый указатель на один из коммитов. По умолчанию, имя основной ветки в Git — master.

2. Что такое HEAD?

HEAD — это указатель, задача которого ссылаться на определенный коммит в репозитории.

3. Способы создания веток.

Ветки можно создать с помощью команды “git branch имя_ветки”, и чтобы перейти на неё необходимо использовать команду “git checkout имя_ветки”. Можно же создать ветку и сразу перейти на неё с помощью “git checkout -b имя_ветки”.

4. Как узнать текущую ветку?

С помощью команды git branch высветятся все ветки, текущая будет подкрашена и/или будет со знаком *.

5. Как переключаться между ветками?

Чтобы переходить по веткам необходимо использовать команду “git checkout имя_ветки”

6. Что такое удаленная ветка?

Это ветка, находящаяся на удалённом сервере GitHub.

7. Что такое ветка отслеживания?

Ветки слежения — это ссылки на определённое состояние удалённых веток. Это локальные ветки, которые напрямую связаны с удалённой веткой. Если, находясь на ветке слежения, выполнить `git pull`, то Git уже будет знать с какого сервера получать данные и какую ветку использовать для слияния. При клонировании репозитория, как правило, автоматически создаётся ветка `master`, которая следит за `origin/master`.

8. Как создать ветку отслеживания?

С помощью команды `git checkout -b имя_ветки origin/имя_ветки`.

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

С помощью команды `git push имя_ветки`.

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

`Git fetch` лишь показывает изменения веток на сервере, но не копирует их на локальный репозиторий, в отличие от команды `Git pull`.

11. Как удалить локальную и удалённую ветки?

Локальную ветку можно удалить с помощью команды `git branch -d имя_ветки`.

12. Изучить модель ветвления `git-flow`. Какие основные типы веток присутствуют в модели `git-flow`? Как организована работа с ветками в модели `git-flow`? В чем недостатки `git-flow`?

Git Flow описывает несколько веток для разработки, релизов и взаимодействия между ними. Репозиторий содержит 2 главные ветки:

- `master` (стандартная/основная ветка);
- `develop` (ответвляется от основной, в нее добавляются новшества, затем сливается с основной);

Как следствие `master` выступает релизной веткой в этой концепции. В Git Flow мы можем использовать следующие типы веток:

- Feature branches;
- Release branches;
- Hotfix branches;

Минусы `git-flow`:

- `git flow` может замедлять работу;
- релизы сложно делать часто;

– большие функции могут потратить дни на конфликты и форсировать несколько циклов тестирования;

Вывод: исследовали базовые возможности по работе с локальными и удаленными ветками Git.