

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение
высшего образования «СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ
УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Институт цифрового развития

ОТЧЁТ

по лабораторной работе №1.3

Дисциплина: «Основы кроссплатформенного программирования»

Тема: «Условные операторы и циклы в языке Python»

Выполнила: студентка 1 курса,
группы ИВТ-б-о-21-1
Диченко Дина Алексеевна

Ставрополь 2022

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

Практическая часть:

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

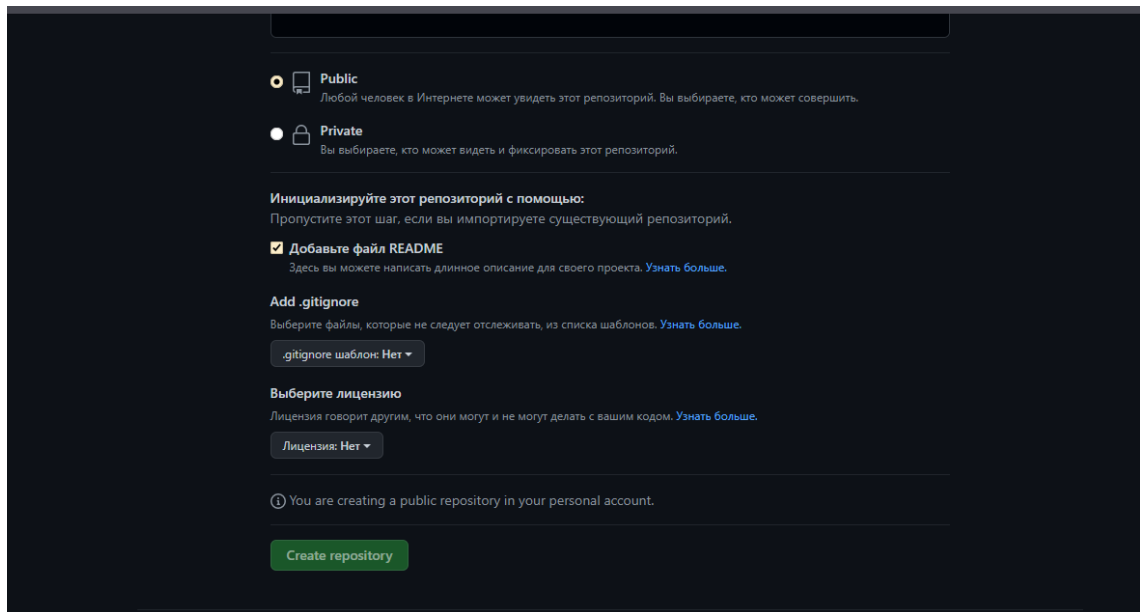


Рисунок 1. Создание репозитория

2. Клонировала репозиторий и создала три файла: 1.txt, 2.txt, 3.txt.

Имя	Дата изменения	Тип
.git	17.05.2022 23:46	Папка с файл
1.txt	20.05.2022 0:10	Текстовый дс
2.txt	20.05.2022 0:11	Текстовый дс
3.txt	20.05.2022 0:11	Текстовый дс
README	17.05.2022 23:46	Файл "MD"

Рисунок 2. Создание трех файлов

3. Проиндексировала первый файл и сделала коммит с комментарием "add 1.txt file". Проиндексировала второй и третий файлы. Перезаписала уже сделанный коммит с новым комментарием "add 2.txt and 3.txt."

```

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git add 1.txt
C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git commit -m "add 1.txt file"
[main 476f838] add 1.txt file
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 1.txt
C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git add 2.txt.
fatal: pathspec '2.txt.' did not match any files
C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git add 2.txt
C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git add 3.txt
C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git commit --amend -m "add 2.txt and 3.txt file"
[main 34665e2] add 2.txt and 3.txt file
Date: Fri May 20 00:36:57 2022 +0300
3 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 1.txt
create mode 100644 2.txt
create mode 100644 3.txt

```

Рисунок 3. Индексация и коммит файлов

4. Создала новую ветку и перешла на нее.

```

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git branch my_first_branch
C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git checkout my_first_branch
Switched to branch 'my_first_branch'

```

Рисунок 4. Создание новой ветки и переход на нее

5. Создала новый файл, закоммитила изменения и перешла на ветку master.




 3	20.05.2022 0:11	Текстовый,
 in_branch	20.05.2022 0:41	Текстовый,
 README	17.05.2022 23:46	Файл "MD"

Рисунок 5. Создание нового файла

```

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git add in_branch.txt
C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git commit -m "add in_branch.txt file"
[my_first_branch 9e3e77a] add in_branch.txt file
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 in_branch.txt

```

Рисунок 6. Коммит нового файла

```

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 1 commit.
(use "git push" to publish your local commits)

```

Рисунок 7. Возвращение на ветку main

6. Создала и перешла на ветку new_branch. Сделала изменения в файле 1.txt, добавила строчку “new row in the 1.txt file”, закоммитила изменения.

```
C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git branch new_branch  
C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git checkout new_branch  
Switched to branch 'new_branch'
```

Рисунок 8. Создание и переход на ветку new_branch.

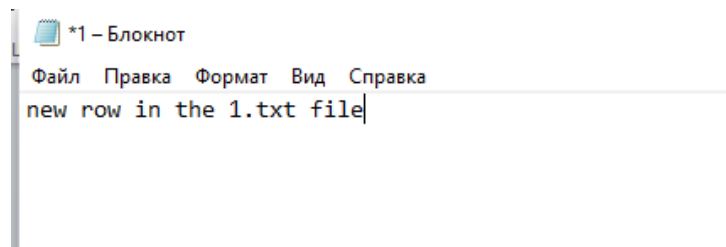


Рисунок 9. Изменение в файле 1.txt

```
C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git add 1.txt  
  
C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git commit -m "changed in 1.txt"  
[new_branch 5b0cebf] changed in 1.txt  
1 file changed, 1 insertion(+)
```

Рисунок 10. Коммит изменений

7. Перейшла на ветку master и слила ветки master и my_first_branch, слила ветки master и new_branch.

```

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git merge my_first_branch
Updating 34665e2..9e3e77a
Fast-forward
 in_branch.txt | 0
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 in_branch.txt

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git merege new_branch
git: 'merege' is not a git command. See 'git --help'.

The most similar command is
    merge

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git merge new_branch
Merge made by the 'recursive' strategy.
 1.txt | 1 +
1 file changed, 1 insertion(+)

```

Рисунок 11. Слияние веток

8. Удалила ветки my_first_branch и new_branch.

```

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git branch -d my_first_branch
Deleted branch my_first_branch (was 9e3e77a).

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git branch -d new_branch
Deleted branch new_branch (was 5b0cebf).

```

Рисунок 12. Удаление веток

9. Создала ветки branch_1 и branch_2.

```

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git branch branch_1
C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git branch branch_2

```

Рисунок 13. Создание веток

10. Перешла на ветку branch_1 и изменила файл 1.txt, изменила файл 3.txt, закоммитила изменения. Перешла на ветку branch_2 и также изменила файл 1.txt, изменила файл 3.txt, закоммитила изменения.

```

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git checkout branch_1
Switched to branch 'branch_1'

```

Рисунок 14. Переход на ветку branch_1

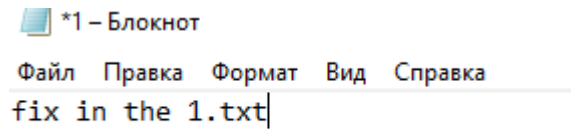


Рисунок 15. Изменение 1.txt

```
C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git add 1.txt
C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git commit -m "change1"
[branch_1 719806e] change1
1 file changed, 1 insertion(+), 1 deletion(-)
```

Рисунок 16. Коммит изменений

```
C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git checkout branch_2
Switched to branch 'branch_2'
```

Рисунок 17. Переход на ветку branch_2

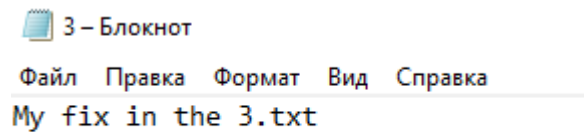


Рисунок 18. Изменение 3.txt

```
C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git add 1.txt
C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git add 3.txt
C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git commit -m "changes"
[branch_2 40edbc5] changes
2 files changed, 2 insertions(+), 1 deletion(-)
```

Рисунок 19. Коммит изменений

11. Слила изменения ветки branch_2 в ветку branch_1. Решила конфликт файла 1.txt в ручном режиме, а конфликт 3.txt используя команду git mergetool с помощью одной из доступных утилит, например Meld.

```
C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>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:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>
```

Рисунок 20. Переход на ветку branch_2

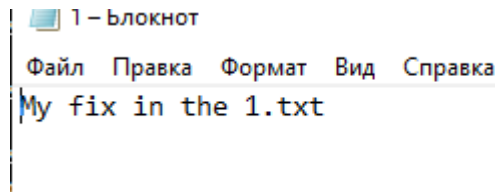


Рисунок 21. Решение конфликта вручную

```
C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git add 1.txt
C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git status
On branch branch_1
You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Changes to be committed:
  modified:   1.txt

Unmerged paths:
  (use "git add <file>..." to mark resolution)
  both modified:   3.txt
```

Рисунок 22. Команда git status

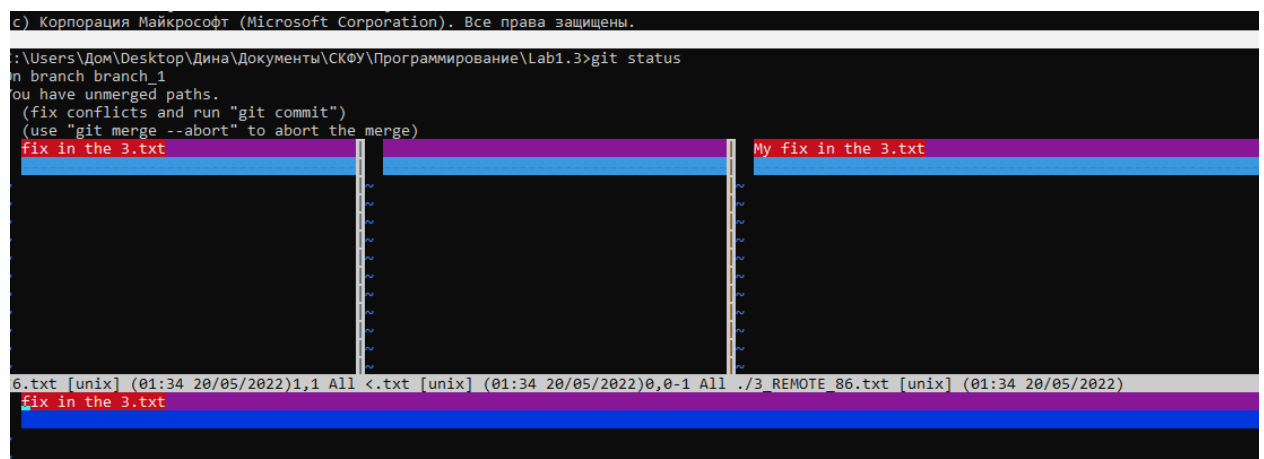


Рисунок 23. Решение конфликта с помощью git mergetool

12. Отправила ветку branch_1 на GitHub.

```

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git push origin branch_1
Enumerating objects: 25, done.
Counting objects: 100% (25/25), done.
Delta compression using up to 2 threads
Compressing objects: 100% (16/16), done.
Writing objects: 100% (24/24), 1.96 KiB | 251.00 KiB/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/DinaDichenko/Lab1.3/pull/new/branch_1
remote:
To https://github.com/DinaDichenko/Lab1.3.git
 * [new branch]      branch_1 -> branch_1

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>

```

Рисунок 24. Отправка ветки branch_1 на GitHub

13. Создала средствами GitHub удаленную ветку branch_3.

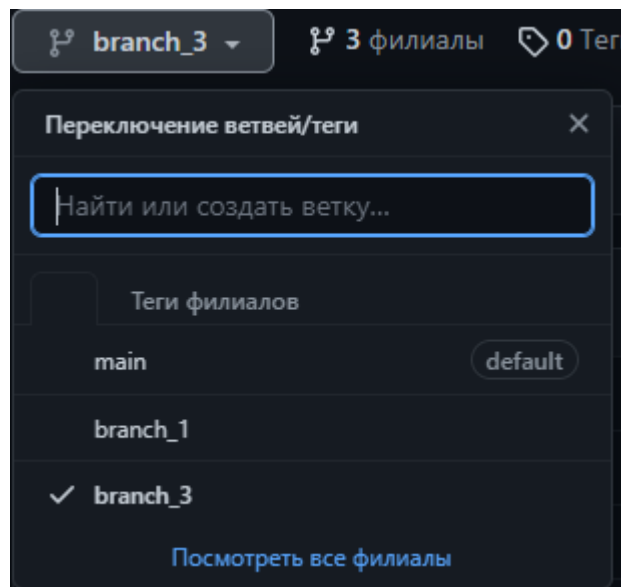


Рисунок 25. Создание ветки branch_3

14. Создала в локальном репозитории ветку отслеживания удаленной ветки branch_3. Перешла на ветку branch_3 и добавила файл файл 2.txt строку "the final fantasy in the 4.txt file".


```

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git checkout --track origin/branch_3
fatal: 'origin/branch_3' is not a commit and a branch 'branch_3' cannot be created from it

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git pull
From https://github.com/DinaDichenko/Lab1.3
* [new branch]      branch_3    -> origin/branch_3
There is no tracking information for the current branch.
Please specify which branch you want to merge with.
See git-pull(1) for details.

    git pull <remote> <branch>

If you wish to set tracking information for this branch you can do so with:

    git branch --set-upstream-to=origin/<branch> branch_1

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>> git checkout branch_3
"checkout" не является внутренней или внешней
командой, исполняемой программой или пакетным файлом.

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git checkout branch_3
Switched to a new branch 'branch_3'
Branch 'branch_3' set up to track remote branch 'branch_3' from 'origin'.

```

Рисунок 26. Создание ветки отслеживания удаленной ветки branch_3

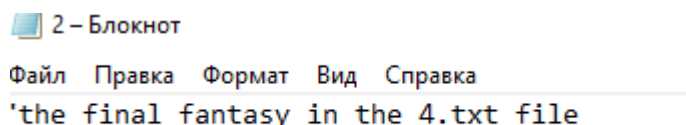


Рисунок 27. Изменение в 2.txt

```

:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git branch
branch_1
branch_2
branch_3
main

:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>
:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git add 2.txt

:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git commit -m "change in 2.txt"
branch_3 33597ed] change in 2.txt
1 file changed, 1 insertion(+)

```

Рисунок 28. Коммит изменений

15. Выполнила перемещение ветки master на ветку branch_2. Отправила изменения веток master и branch_2 на GitHub.

```

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git add .

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git commit -m "main"
[main 31bc9a4] main
 1 file changed, 21 insertions(+)
 create mode 100644 LICENSE

```

Рисунок 29. Отправка изменений

```

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git add .

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git commit -m "2"
[branch_2 70fd44a] 2
 1 file changed, 21 insertions(+)
 create mode 100644 LICENSE

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab1.3>git push origin branch_2
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 2 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 883 bytes | 883.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/DinaDichenko/Lab1.3.git
 40edbc5..70fd44a branch_2 -> branch_2

```

Рисунок 30. Отправка изменений

Ответы на вопросы:

1. Что такое ветка?

Ветка в Git — это простой перемещаемый указатель на один из таких КОММИТОВ.

2. Что такое HEAD?

HEAD — это указатель, задача которого ссылаться на определенный коммит в репозитории. Суть данного указателя можно попытаться объяснить с разных сторон.

3. Способы создания веток.

Чтобы создать новую ветку, необходимо использовать команду `git branch`.

4. Как узнать текущую ветку?

Увидеть текущую ветку можно при помощи простой команды `git log`, которая покажет вам куда указывают указатели веток. Эта опция называется `-decorate`.

5. Как переключаться между ветками?

Для переключения на существующую ветку выполните команду `git checkout`.

6. Что такое удаленная ветка?

Удалённые ссылки — это ссылки (указатели) в ваших удалённых репозиториях, включая ветки, теги и так далее. Полный список удалённых ссылок можно получить с помощью команды `git ls-remote <remote>` или команды `git remote show <remote>` для получения удалённых веток и дополнительной информации.

7. Что такое ветка отслеживания?

Ветки слежения — это ссылки на определённое состояние удалённых веток. Это локальные ветки, которые нельзя перемещать; Git перемещает их автоматически при любой коммуникации с удалённым репозиторием, чтобы гарантировать точное соответствие с ним.

8. Как создать ветку отслеживания?

При клонировании репозитория, как правило, автоматически создаётся ветка `master`, которая следит за `origin/master`. Однако, при желании вы можете настроить отслеживание и других веток — следить за ветками на других серверах или отключить слежение за веткой `master`. Вы только что видели простейший пример, что сделать это можно с помощью команды `git checkout -b <branch> <remote>/<branch>`. Это часто используемая команда, поэтому Git предоставляет сокращённую форму записи в виде флага `-track`.

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

Когда вы хотите поделиться веткой, вам необходимо отправить её на удалённый сервер, где у вас есть права на запись. Ваши локальные ветки автоматически не синхронизируются с удалёнными при отправке — вам нужно явно указать те ветки, которые вы хотите отправить.

Таким образом, вы можете использовать свои личные ветки для работы, которую не хотите показывать, а отправлять только те тематические ветки, над которыми вы хотите работать с кем-то совместно. Если у вас есть ветка `serverfix`, над которой вы хотите работать с кем-то ещё, вы можете отправить её точно так же, как вы отправляли вашу первую ветку. Выполните команду `git push <remote> <branch>`.

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

Для синхронизации ваших изменений с удаленным сервером выполните команду `git fetch <remote>` (в нашем случае `git fetch origin`). Эта команда определяет какому серверу соответствует “origin” (в нашем случае это `git.ourcompany.com`), извлекает оттуда данные, которых у вас ещё нет, и обновляет локальную базу данных, сдвигая указатель `origin/master` на новую позицию.

11. Как удалить локальную и удаленную ветки?

Вы можете удалить ветку на удалённом сервере используя параметр `--delete` для команды `git push`.

12. Изучить модель ветвления `git-flow` (использовать материалы статей <https://www.atlassian.com/ru/git/tutorials/comparing-workflows/gitflow-workflow>, <https://habr.com/ru/post/106912/>). 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 — это устаревшая версия рабочего процесса Git, в свое время ставшая принципиально новой стратегией управления ветками в Git. Это альтернативная модель ветвления Git, в которой используются функциональные ветки и несколько основных веток.

Основными типами веток являются функциональные ветки, ветки выпуска, ветки исправления.

Последовательность действий при работе по модели Gitflow:

- 1) из ветки main создается ветка develop.
- 2) из ветки develop создается ветка release.
- 3) из ветки develop создаются ветки feature.
- 4) когда работа над веткой feature завершается, она сливается в ветку develop.
- 5) когда работа над веткой release завершается, она сливается с ветками develop и main.
- 6) если в ветке main обнаруживается проблема, из main создается ветка hotfix.
- 7) когда работа над веткой hotfix завершается, она сливается с ветками develop и main.

Первая проблема: авторам приходится использовать ветку develop вместо master, поскольку master зарезервирован для кода, который отправляется в продакшен. Существует сложившийся обычай называть рабочую ветвь по умолчанию master, и делать ответвления и слияния с ней. Большинство инструментов по умолчанию используют это название для основной ветки и по умолчанию выводят именно ее, и бывает неудобно постоянно переключаться вручную на другую ветку.

Вторая проблема процесса git flow — сложности, возникающие из-за веток для патчей и для релиза.

Вывод: в результате выполнения работы были исследованы базовые возможности по работе с локальными и удаленными ветками Git.