

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 №2
дисциплины «Программирование на Python»

Выполнил:
Касимов Асхаб Арсенович
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и
вычислительная техника»,
направленность (профиль)
«Программное обеспечение средств
вычислительной техники и
автоматизированных систем», очная
форма обучения

(подпись)

Руководитель практики:
Воронкин Р.А., доцент кафедры
инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

Тема: Исследование возможностей Git для работы с локальными репозиториями

Цель: исследовать базовые возможности системы контроля версий Git для работы с локальными репозиториями.

Порядок выполнения работы:

1. Создал новый репозиторий:

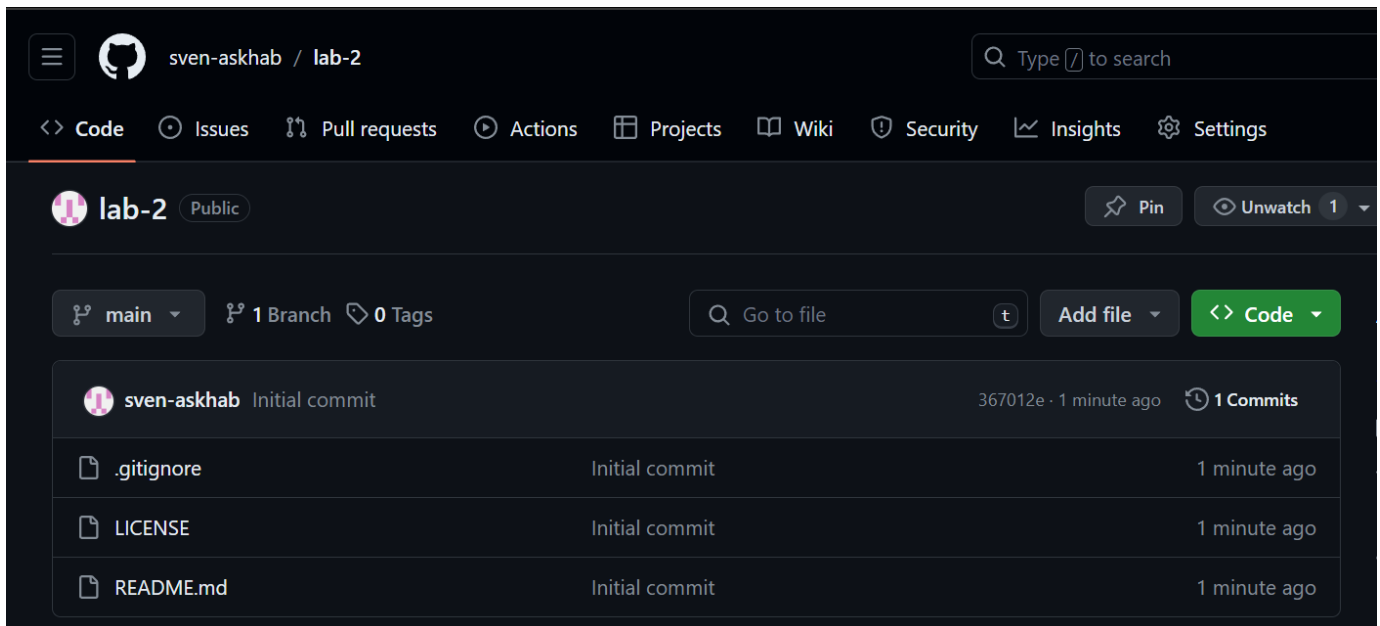


Рисунок 1. Новый репозиторий Lab1.2

2. Проработал примеры лабораторной работы:

```
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheba/Прогр Python
$ git clone https://github.com/schacon/simplegit-progit
Cloning into 'simplegit-progit'...
remote: Enumerating objects: 13, done.
remote: Total 13 (delta 0), reused 0 (delta 0), pack-reused 13
Receiving objects: 100% (13/13), done.
Resolving deltas: 100% (3/3), done.
```

Рисунок 2. Клонирование репозитория

```
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheба/Прорп Python/simplegit-progit (ma
ster)
$ git log
commit ca82a6dff817ec66f44342007202690a93763949 (HEAD -> master, origin/master, orig
in/HEAD)
Author: Scott Chacon <schacon@gmail.com>
Date:   Mon Mar 17 21:52:11 2008 -0700

    changed the verison number

commit 085bb3bcb608e1e8451d4b2432f8ecbe6306e7e7
Author: Scott Chacon <schacon@gmail.com>
Date:   Sat Mar 15 16:40:33 2008 -0700

    removed unnecessary test code

commit a11bef06a3f659402fe7563abf99ad00de2209e6
Author: Scott Chacon <schacon@gmail.com>
Date:   Sat Mar 15 10:31:28 2008 -0700

    first commit
```

Рисунок 3. Результат работы команды git log

```

33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheba/Python/simlegit-progit (master)
$ git log -p -2
commit ca82a6dff817ec66f44342007202690a93763949 (HEAD -> master, origin/master, origin/HEAD)
Author: Scott Chacon <schacon@gmail.com>
Date: Mon Mar 17 21:52:11 2008 -0700

    changed the verison number

diff --git a/Rakefile b/Rakefile
index a874b73..8f94139 100644
--- a/Rakefile
+++ b/Rakefile
@@ -5,7 +5,7 @@ require 'rake/gempackagetask'
spec = Gem::Specification.new do |s|
  s.platform = Gem::Platform::RUBY
  s.name = "simplegit"
- s.version = "0.1.0"
+ s.version = "0.1.1"
  s.author = "Scott Chacon"
  s.email = "schacon@gmail.com"
  s.summary = "A simple gem for using Git in Ruby code."

commit 085bb3bcb608e1e8451d4b2432f8ecbe6306e7e7
Author: Scott Chacon <schacon@gmail.com>
Date: Sat Mar 15 16:40:33 2008 -0700

    removed unnecessary test code

diff --git a/lib/simplegit.rb b/lib/simplegit.rb
index a0a60ae..47c6340 100644
--- a/lib/simplegit.rb
+++ b/lib/simplegit.rb
@@ -18,8 +18,3 @@ class SimpleGit
  end

end

-
- if $0 == __FILE__
-   git = SimpleGit.new
-   puts git.show
- end
\ No newline at end of file

```

Рисунок 4. Результат работы команды git log -p -2

```

33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheba/Python/simlegit-progit (master)
$ git log --stat
commit ca82a6dff817ec66f44342007202690a93763949 (HEAD -> master, origin/master, origin/HEAD)
Author: Scott Chacon <schacon@gmail.com>
Date: Mon Mar 17 21:52:11 2008 -0700

    changed the verison number

Rakefile | 2 +-
1 file changed, 1 insertion(+), 1 deletion(-)

commit 085bb3bcb608e1e8451d4b2432f8ecbe6306e7e7
Author: Scott Chacon <schacon@gmail.com>
Date: Sat Mar 15 16:40:33 2008 -0700

    removed unnecessary test code

lib/simplegit.rb | 5 -----
1 file changed, 5 deletions(-)

commit a11bef06a3f659402fe7563abf99ad00de2209e6
Author: Scott Chacon <schacon@gmail.com>
Date: Sat Mar 15 10:31:28 2008 -0700

    first commit

README | 6 ++++++
Rakefile | 23 ++++++
lib/simplegit.rb | 25 ++++++
3 files changed, 54 insertions(+)

```

Рисунок 5. Результат работы команды git log --stat

```
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheба/Прорп Python/simplegit-progit (master)
$ git log --pretty=oneline
ca82a6dff817ec66f44342007202690a93763949 (HEAD -> master, origin/master, origin/HEAD) changed the verison number
085bb3bcb608e1e8451d4b2432f8ecbe6306e7e7 removed unnecessary test code
a11bef06a3f659402fe7563abf99ad00de2209e6 first commit

33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheба/Прорп Python/simplegit-progit (master)
```

Рисунок 6. Результат работы команды git log --pretty=online

```
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheба/Прорп Python/simplegit-progit (master)
$ git log --pretty=format:"%h - %an, %ar : %s"
ca82a6d - Scott Chacon, 16 years ago : changed the verison number
085bb3b - Scott Chacon, 16 years ago : removed unnecessary test code
a11bef0 - Scott Chacon, 16 years ago : first commit
```

Рисунок 7. Результат команды git log --pretty=format:"%h - %an, %ar : %s"

```
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheба/Прорп Python/simplegit-progit (master)
$ git log --pretty=format:"%h %s" --graph
* ca82a6d changed the verison number
* 085bb3b removed unnecessary test code
* a11bef0 first commit
```

Рисунок 8. Результат работы команды git log --pretty=format:"%h %s" --graph

```
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheба/Прорп Python/simplegit-progit (master)
$ git clone https://github.com/schacon/ticgit
Cloning into 'ticgit'...
remote: Enumerating objects: 1857, done.
remote: Total 1857 (delta 0), reused 0 (delta 0), pack-reused 1857
Receiving objects: 100% (1857/1857), 334.06 KiB | 429.00 KiB/s, done.
Resolving deltas: 100% (837/837), done.

33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheба/Прорп Python/simplegit-progit (master)
$ |
```

Рисунок 9. Клонирование репозитория ticgit

```
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheба/Прорп Python/simplegit-progit/ticgit (master)
$ git remote
origin
```

Рисунок 10. Результат работы команды git remote

```
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheба/Прорп Python/simplegit-progit/ticgit (master)
$ git remote -v
origin https://github.com/schacon/ticgit (fetch)
origin https://github.com/schacon/ticgit (push)
```

Рисунок 11. Результат работы команды git remote -v

```

33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheба/Прорп Python/simplegit-progit/ticgit/ticgit/ticgit (master)
$ git remote add pb https://github.com/paulboone/ticgit

33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheба/Прорп Python/simplegit-progit/ticgit/ticgit/ticgit (master)
$ git remote -v
origin https://github.com/schacon/ticgit (fetch)
origin https://github.com/schacon/ticgit (push)
pb      https://github.com/paulboone/ticgit (fetch)
pb      https://github.com/paulboone/ticgit (push)

33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheба/Прорп Python/simplegit-progit/ticgit/ticgit/ticgit (master)
$

```

Рисунок 12. Результат работы команды git remote add pb

```

33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheба/Прорп Python/ticgit (master)
$ git fetch pb
remote: Enumerating objects: 43, done.
remote: Counting objects: 100% (22/22), done.
remote: Total 43 (delta 22), reused 22 (delta 22), pack-reused 21
Unpacking objects: 100% (43/43), 5.99 KiB | 25.00 KiB/s, done.
From https://github.com/paulboone/ticgit
* [new branch]      master      -> pb/master
* [new branch]      ticgit      -> pb/ticgit

```

fetch

Рисунок 13. Результат работы команды git fetch pb

```

33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheба/Прорп Python/ticgit (master)
$ git remote show origin
* remote origin
  Fetch URL: https://github.com/schacon/ticgit
  Push URL: https://github.com/schacon/ticgit
  HEAD branch: master
  Remote branches:
    master tracked
    ticgit tracked
  Local branch configured for 'git pull':
    master merges with remote master
  Local ref configured for 'git push':
    master pushes to master (up to date)

```

Рисунок 14. Результат работы команды git remote show origin

```

33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheба/Прорп Python/ticgit (master)
$ git remote rename pb paul
Renaming remote references: 100% (2/2), done.

33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheба/Прорп Python/ticgit (master)
$ git remote
origin
paul

```

Рисунок 15. Результат работы команды git remote rename pb paul

```
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/учеба/Прогр Python/ticgit (master)
$ git remote remove paul

33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/учеба/Прогр Python/ticgit (master)
$ git remote
origin
```

Рисунок 16. Результат работы команды git remote remove paul

```
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/учеба/Прогр Python/ticgit (master)
$ git tag

33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/учеба/Прогр Python/ticgit (master)
$ git tag -l "v1.8.5*"

33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/учеба/Прогр Python/ticgit (master)
$ git tag -a v1.4 -m "my version 1.4"

33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/учеба/Прогр Python/ticgit (master)
$ git tag
v1.4
```

Рисунок 17. Создание аннотированного тега

```
tag v1.4
Tagger: sven-askhab <sven.ashab@gmail.com>
Date: Tue Feb 13 05:05:52 2024 +0300

my version 1.4

commit 847256809a3d518cd36b8f81859401416fe8d945 (HEAD -> master, tag: v1.4, origin/master, origin/HEAD)
Author: Jeff Welling <Jeff.Welling@gmail.com>
Date: Tue Apr 26 17:29:17 2011 -0700

    Added note to clarify which is the canonical TicGit-ng repo

diff --git a/README.mkd b/README.mkd
index ab92035..9ea9ff9 100644
--- a/README.mkd
+++ b/README.mkd
@@ -1,3 +1,6 @@
+Note: the original TicGit author has pulled all the TicGit-ng changes into his
+repository, creating a potentially confusing situation. The schacon TicGit repo,
+this one, is not consistently maintained. For up to date TicGit-ng info and code,
+check the canonical TicGit-ng repository at
+https://github.com/jeffwelling/ticgit
```

Рисунок 18. Результат работы команды git show v1.4

Рисунок 19. Удаление ранее созданного тега

3. Клонировал ранее созданные репозиторий:

```
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/учеба/Прогр Python/ticgit (master)
$ git clone https://github.com/sven-askhab/lab-2.git
Cloning into 'lab-2'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
```

Рисунок 20. Клонирование репозитория Lab1.2

4. Дополнил .gitignore:

```
160 #.idea/
161 .vscode
162
```

Рисунок 21. Добавленная строка в .gitignore

5. Добавил в файл README.md информацию о группе и ФИО:

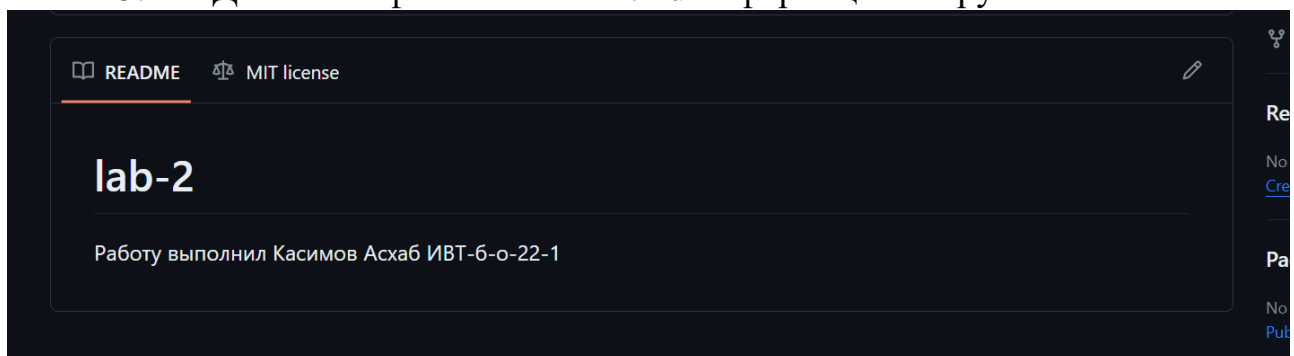


Рисунок 22. Добавление информации в файл README.md

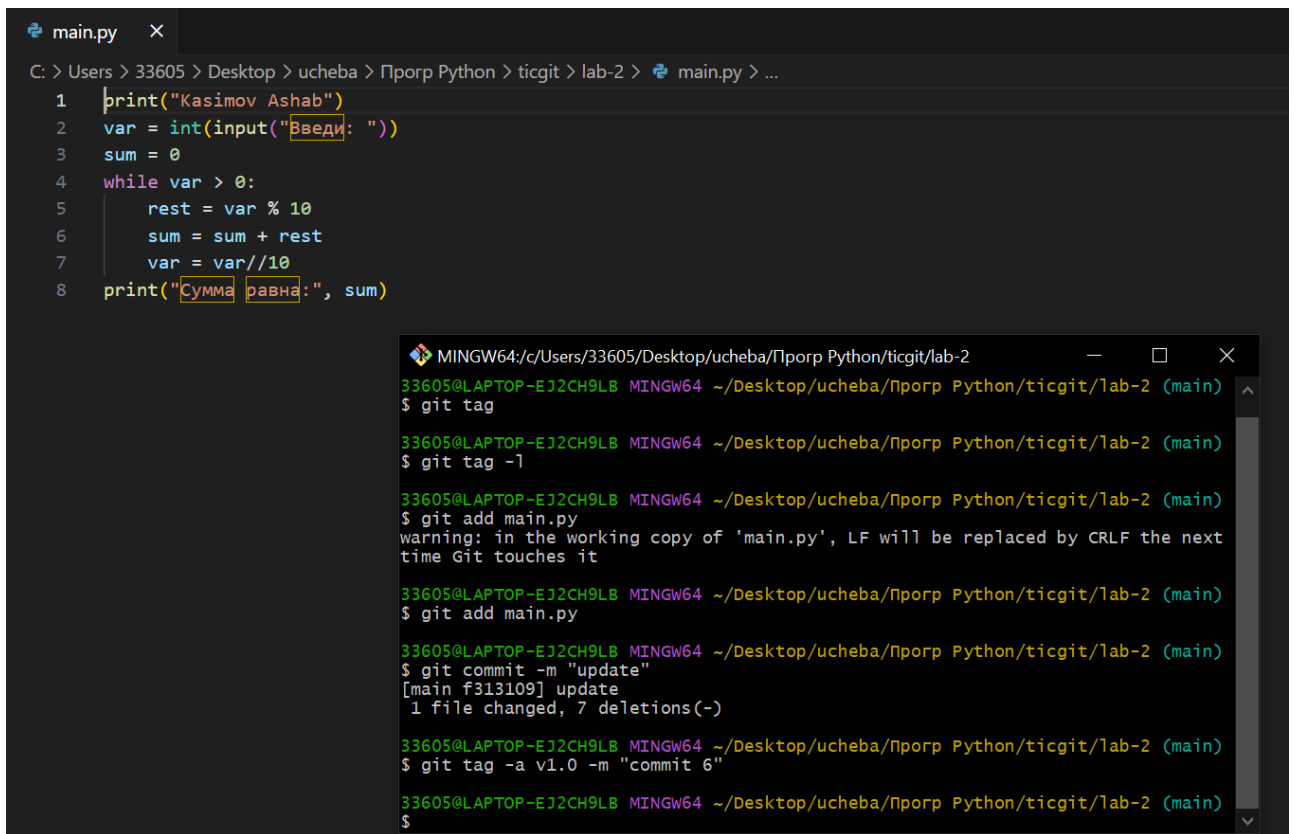
6. Написал небольшую программу на языке Python:

The image shows a code editor with a file named `main.py` open. The code is a Python script that prints a name, takes an input, and calculates the sum of its digits. Below the code editor is a terminal window showing the execution of Git commands to add the file and commit the changes.

```
main.py x ind2.py 6 ●
C: > Users > 33605 > Desktop > ucheba > Пporp Python > ticgit > lab-2 > main.py > ...
1 print("Kasimov Ashab")
2 var = int(input("Введи: "))
3 sum = 0
4 while var > 0:
5     rest = var % 10
6     sum = sum + rest
7     var = var//10
8 print("Сумма равна:", sum)
```

```
MINGW64:/c/Users/33605/Desktop/ucheba/Пporp Python/ticgit/lab-2
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheba/Пporp Python/ticgit/lab-2 (main)
$ git add main.py
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheba/Пporp Python/ticgit/lab-2 (main)
$ git commit -m "update"
[main 46efc17] update
1 file changed, 1 insertion(+), 1 deletion(-)
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheba/Пporp Python/ticgit/lab-2 (main)
$ git add main.py
warning: in the working copy of 'main.py', LF will be replaced by CRLF the next
time Git touches it
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheba/Пporp Python/ticgit/lab-2 (main)
$ git commit -m "commit 4"
[main b66a23d] commit 4
1 file changed, 1 insertion(+), 1 deletion(-)
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheba/Пporp Python/ticgit/lab-2 (main)
$ |
```

Рисунок 23. Первые 4 коммита и код программы



The image shows a code editor window with a file named `main.py` and a terminal window below it. The code in `main.py` is as follows:

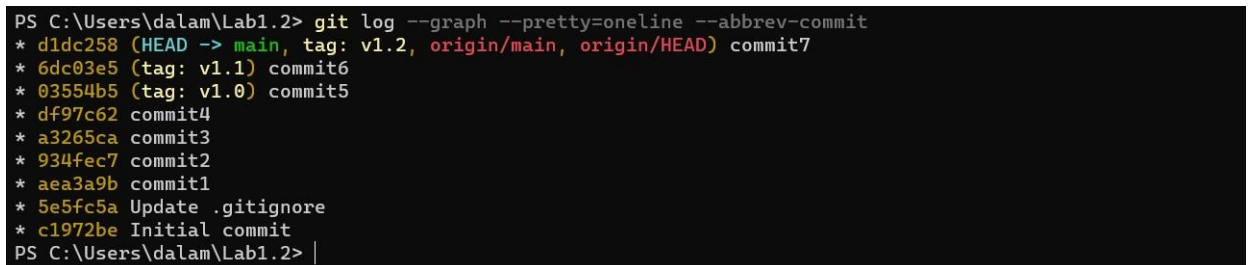
```
1 print("Kasimov Ashab")
2 var = int(input("Введи: "))
3 sum = 0
4 while var > 0:
5     rest = var % 10
6     sum = sum + rest
7     var = var//10
8 print("Сумма равна:", sum)
```

The terminal window shows the following commands and output:

```
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheba/Прорп Python/ticgit/lab-2 (main)
$ git tag
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheba/Прорп Python/ticgit/lab-2 (main)
$ git tag -l
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheba/Прорп Python/ticgit/lab-2 (main)
$ git add main.py
warning: in the working copy of 'main.py', LF will be replaced by CRLF the next
time Git touches it
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheba/Прорп Python/ticgit/lab-2 (main)
$ git add main.py
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheba/Прорп Python/ticgit/lab-2 (main)
$ git commit -m "update"
[main f313109] update
1 file changed, 7 deletions(-)
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheba/Прорп Python/ticgit/lab-2 (main)
$ git tag -a v1.0 -m "commit 6"
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheba/Прорп Python/ticgit/lab-2 (main)
$
```

Рисунок 24. Последние 3 коммита с тегами

7. Просмотрел историю хранилища:



The terminal window shows the output of the command `git log --graph --pretty=oneline --abbrev-commit`:

```
PS C:\Users\dalam\Lab1.2> git log --graph --pretty=oneline --abbrev-commit
* d1dc258 (HEAD -> main, tag: v1.2, origin/main, origin/HEAD) commit7
* 6dc03e5 (tag: v1.1) commit6
* 03554b5 (tag: v1.0) commit5
* df97c62 commit4
* a3265ca commit3
* 934fec7 commit2
* aea3a9b commit1
* 5e5fc5a Update .gitignore
* c1972be Initial commit
PS C:\Users\dalam\Lab1.2> |
```

Рисунок 25. Результат работы команды `git log --graph --pretty=oneline --abbrev-commit`

8. Посмотрел содержимое коммитов:

```

print("Kasimov Ashab")
-var = int(input("Введи: "))
-sum = 0
-while var > 0:
-    rest = var % 10
-    sum = sum + rest
-    var = var//10
-print("Сумма равна:", sum)
\ No newline at end of file

33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheba/Пporp Python/ticgit/lab-2 (main)
$ git show HEAD~1
commit b66a23d68b30f073ef6c4e194751a5f7dc6bf8cc
Author: sven-askhab <sven.ashab@gmail.com>
Date: Tue Feb 13 05:20:06 2024 +0300

    commit 4

diff --git a/main.py b/main.py
index d5edb98..0b1422f 100644
--- a/main.py
+++ b/main.py
@@ -1,5 +1,5 @@
 print("Kasimov Ashab")
-var = int(input("Введи число: "))
+var = int(input("Введи: "))
 sum = 0
 while var > 0:
: ...skipping...
commit b66a23d68b30f073ef6c4e194751a5f7dc6bf8cc
Author: sven-askhab <sven.ashab@gmail.com>
Date: Tue Feb 13 05:20:06 2024 +0300

    commit 4

diff --git a/main.py b/main.py
index d5edb98..0b1422f 100644
--- a/main.py
+++ b/main.py
@@ -1,5 +1,5 @@
 print("Kasimov Ashab")
-var = int(input("Введи число: "))
+var = int(input("Введи: "))
 sum = 0
 while var > 0:
     rest = var % 10

```

Рисунок 26. Результат работы команд git show HEAD и git show HEAD~1

```

33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/учеба/Прогр Python/ticgit/lab-2 (main)
$ git show b66a23d
commit b66a23d68b30f073ef6c4e194751a5f7dc6bf8cc
Author: sven-askhab <sven.ashab@gmail.com>
Date: Tue Feb 13 05:20:06 2024 +0300

    commit 4

diff --git a/main.py b/main.py
index d5edb98..0b1422f 100644
--- a/main.py
+++ b/main.py
@@ -1,5 +1,5 @@
 print("Kasimov Ashab")
:....skipping...
commit b66a23d68b30f073ef6c4e194751a5f7dc6bf8cc
Author: sven-askhab <sven.ashab@gmail.com>
Date: Tue Feb 13 05:20:06 2024 +0300

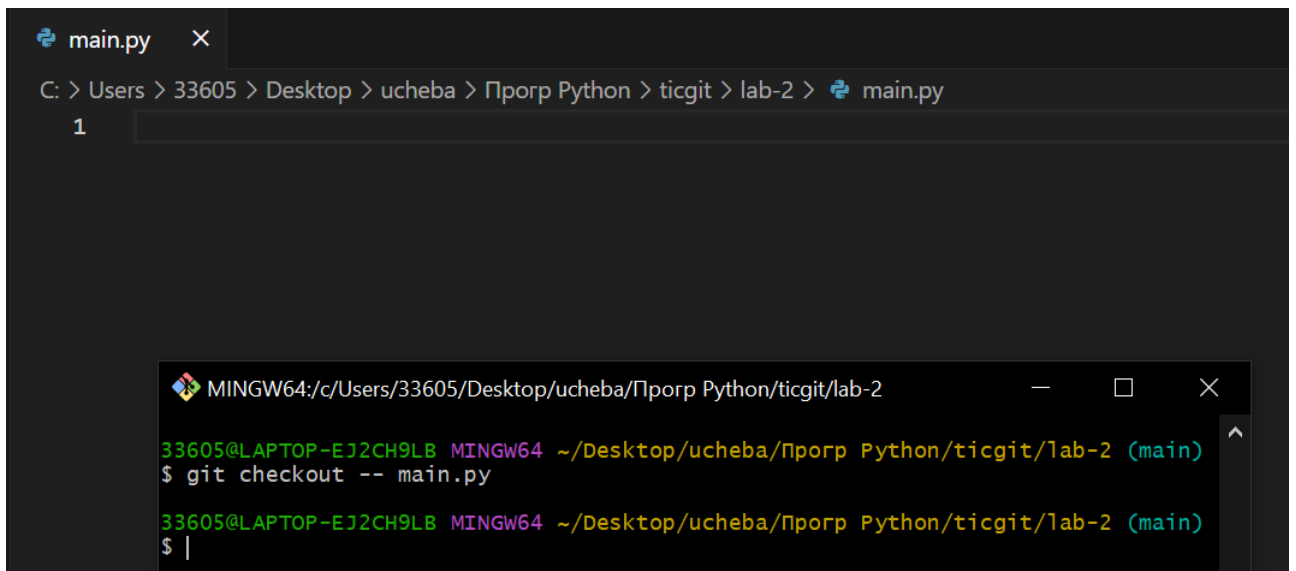
    commit 4

diff --git a/main.py b/main.py
index d5edb98..0b1422f 100644
--- a/main.py
+++ b/main.py
@@ -1,5 +1,5 @@
 print("Kasimov Ashab")
-var = int(input("Введи число: "))
+var = int(input("Введи: "))
sum = 0
while var > 0:
    rest = var % 10

```

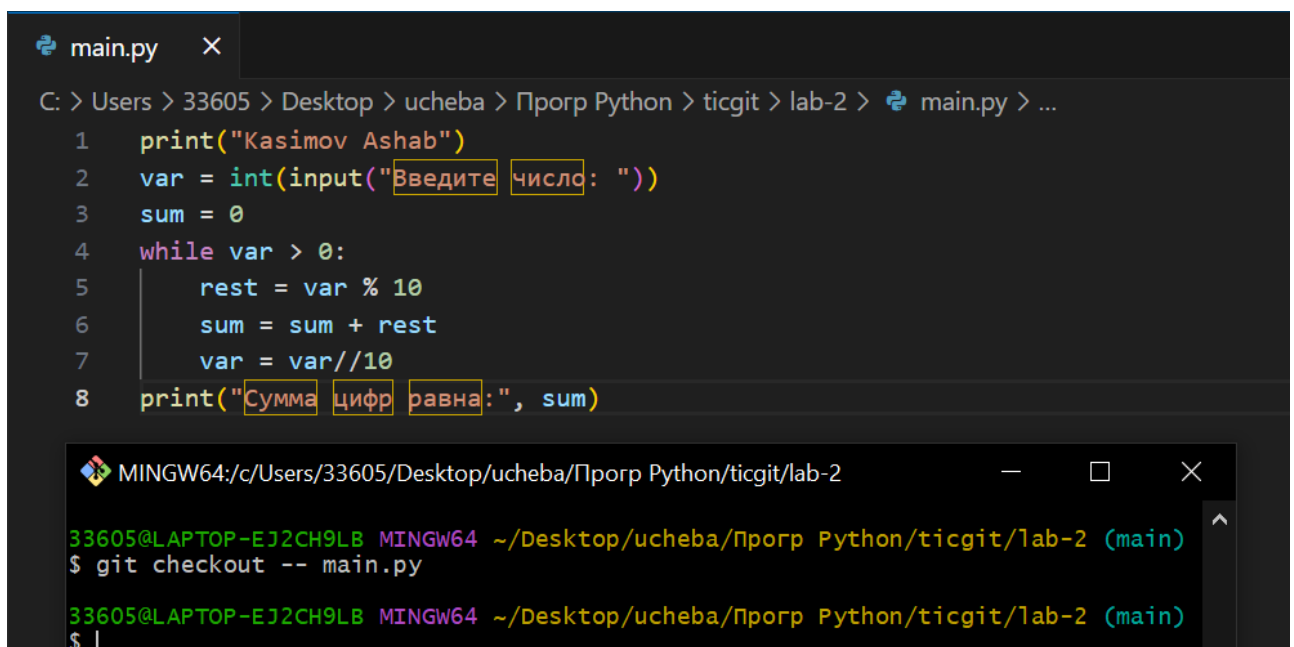
Рисунок 27. Результат работы команды git show b66a23d

9. Удалил код из файла main.py, а затем удалил все несохраненные изменения:



The screenshot shows a code editor with a tab for `main.py`. The file path is `C: > Users > 33605 > Desktop > ucheba > Прорп Python > ticgit > lab-2 > main.py`. The file content is empty, with a line number 1 visible. Below the editor is a terminal window titled `MINGW64:/c/Users/33605/Desktop/ucheba/Прорп Python/ticgit/lab-2`. The terminal shows the command `$ git checkout -- main.py` being executed twice, with the prompt `$` appearing each time.

Рисунок 28. Пустой main.py



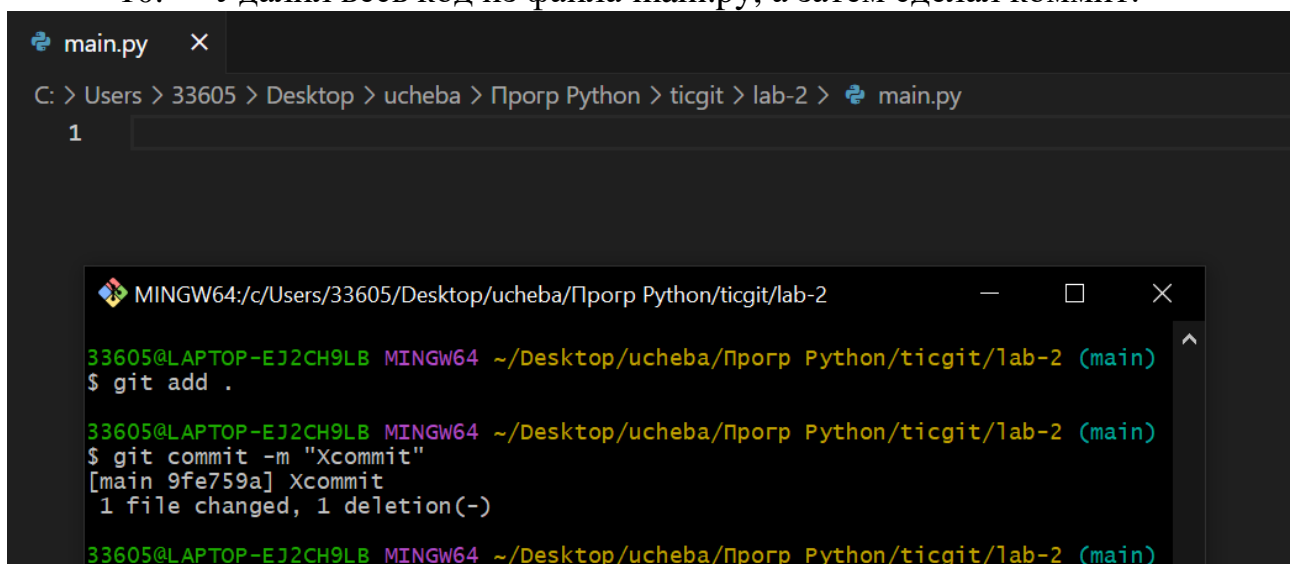
The screenshot shows the same code editor with `main.py` now containing Python code. The code is as follows:

```
1 print("Kasimov Ashab")
2 var = int(input("Введите число: "))
3 sum = 0
4 while var > 0:
5     rest = var % 10
6     sum = sum + rest
7     var = var // 10
8 print("Сумма цифр равна:", sum)
```

Below the editor is a terminal window titled `MINGW64:/c/Users/33605/Desktop/ucheba/Прорп Python/ticgit/lab-2`. The terminal shows the command `$ git checkout -- main.py` being executed twice, with the prompt `$` appearing each time.

Рисунок 29. Код вернулся после команды `git checkout -- prog/main.py`

10. Удалил весь код из файла `main.py`, а затем сделал коммит:

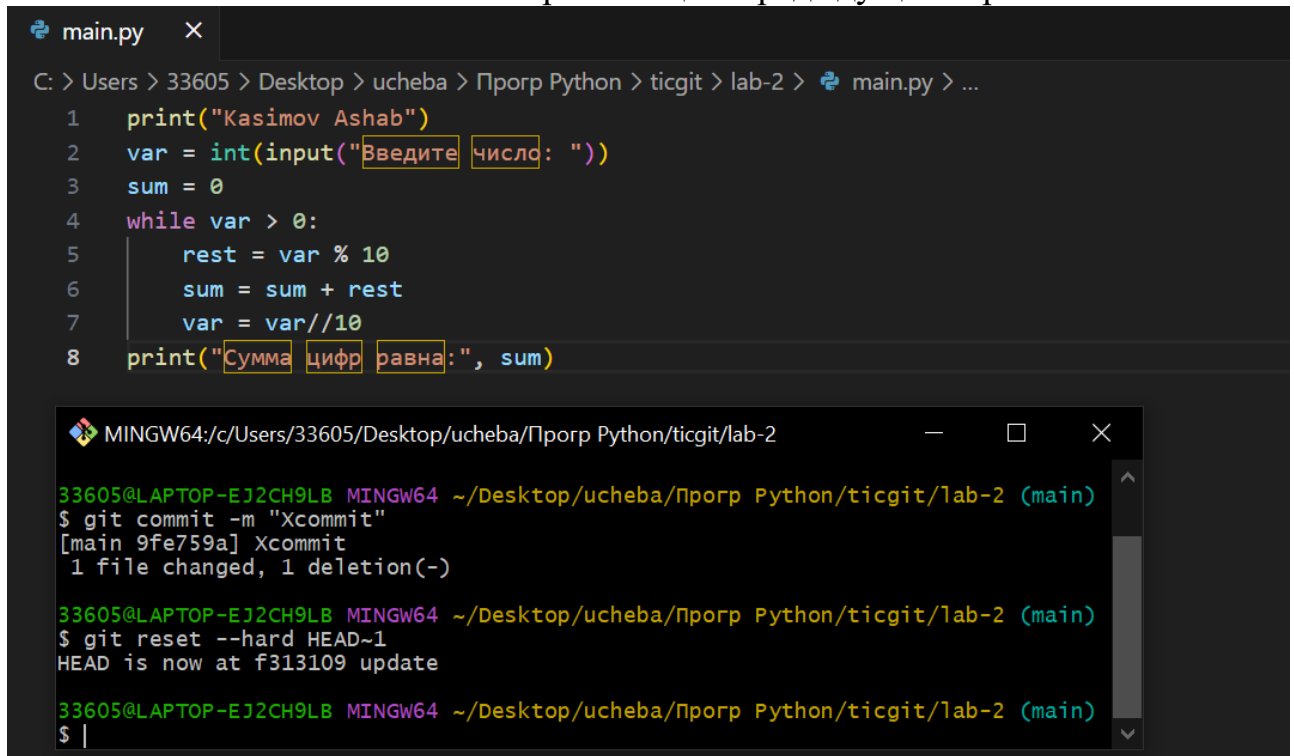


The screenshot shows the same code editor with `main.py` now empty. The file path is `C: > Users > 33605 > Desktop > ucheba > Прорп Python > ticgit > lab-2 > main.py`. Below the editor is a terminal window titled `MINGW64:/c/Users/33605/Desktop/ucheba/Прорп Python/ticgit/lab-2`. The terminal shows the following commands and output:

```
$ git add .
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheba/Прорп Python/ticgit/lab-2 (main)
$ git commit -m "Xcommit"
[main 9fe759a] Xcommit
1 file changed, 1 deletion(-)
```

Рисунок 30. Коммит после удаления кода

11. Откатил состояние хранилище к предыдущей версии коммита:



The image shows a code editor window with a file named `main.py` and a terminal window below it. The code in `main.py` is a Python script that prints "Kasimov Ashab", prompts for a number, and calculates the sum of its digits. The terminal shows the execution of `git commit -m "Xcommit"` and `git reset --hard HEAD~1`.

```
C: > Users > 33605 > Desktop > ucheba > Прорп Python > ticgit > lab-2 > main.py > ...

1  print("Kasimov Ashab")
2  var = int(input("Введите число: "))
3  sum = 0
4  while var > 0:
5      rest = var % 10
6      sum = sum + rest
7      var = var//10
8  print("Сумма цифр равна:", sum)

MINGW64:/c/Users/33605/Desktop/ucheba/Прорп Python/ticgit/lab-2
33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheba/Прорп Python/ticgit/lab-2 (main)
$ git commit -m "Xcommit"
[main 9fe759a] Xcommit
1 file changed, 1 deletion(-)

33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheba/Прорп Python/ticgit/lab-2 (main)
$ git reset --hard HEAD~1
HEAD is now at f313109 update

33605@LAPTOP-EJ2CH9LB MINGW64 ~/Desktop/ucheba/Прорп Python/ticgit/lab-2 (main)
$ |
```

Рисунок 31. Код вернулся после команды `git reset --hard HEAD~1`

Вывод: чтобы удалить не сохраненные коммитом изменения, можно выполнить команду `git checkout -- <имя файла>`, это действие удалит все несохраненные изменения, а чтобы удалить сохраненные коммитом изменения, нужно откатить состояние хранилища к предыдущей версии коммита командой `git reset --hard HEAD~1`, это действие вернет все хранилище к состоянию, которое было зафиксировано в предыдущем коммите. Все изменения, внесенные после этого коммита, будут потеряны.

Ответы на контрольные вопросы:

1. Как выполнить историю коммитов в Git? Какие существуют дополнительные опции для просмотра истории коммитов?

После того, как вы создали несколько коммитов или же клонировали репозиторий с уже существующей историей коммитов, вероятно Вам понадобится возможность посмотреть, что было сделано – историю коммитов. Одним из основных и наиболее мощных инструментов для этого является команда `git log`.

Команда `git log` имеет очень большое количество опций для поиска коммитов по разным критериям. Рассмотрим наиболее популярные из них.

Одним из самых полезных аргументов является `-p` или `--patch`, который показывает разницу (выводит патч), внесенную в каждый коммит.

Если вы хотите увидеть сокращенную статистику для каждого коммита, вы можете использовать опцию `--stat`.

Следующей действительно полезной опцией является `--pretty`. Эта опция меняет формат вывода. Существует несколько встроенных вариантов отображения. Опция `oneline` выводит каждый коммит в одну строку, что может быть очень удобным если вы просматриваете большое количество коммитов. К тому же, опции `short`, `full` и `fuller` делают вывод приблизительно в том же формате, но с меньшим или большим количеством информации соответственно.

Наиболее интересной опцией является `format`, которая позволяет указать

формат для вывода информации.

2. Как ограничить вывод при просмотре истории коммитов?

В дополнение к опциям форматирования вывода, команда `git log` принимает несколько опций для ограничения вывода – опций, с помощью которых можно увидеть определенное подмножество коммитов. Одна из таких опций – это опция `-2`, которая показывает только последние два коммита. В действительности вы можете использовать `-<n>`, где `n` – это любое натуральное число и представляет собой `n` последних коммитов. На практике вы не будете часто использовать эту опцию, потому что `Git` по умолчанию использует постраничный вывод, и вы будете видеть только одну страницу зараз.

Опции для ограничения вывода по времени, такие как `--since` и `--until`, являются очень удобными.

Опция `--author` дает возможность фильтровать по автору коммита, а опция `--grep` искать по ключевым словам в сообщении коммита.

Следующим действительно полезным фильтром является опция `-S`, которая принимает аргумент в виде строки и показывает только те коммиты, в которых изменение в коде повлекло за собой добавление или удаление этой строки.

Последней полезной опцией, которую принимает команда `git log` как фильтр, является путь. Если вы укажете каталог или имя файла, вы ограничите вывод только теми коммитами, в которых были изменения этих файлов. Эта опция всегда указывается последней после двойного тире (`--`), чтобы отделить пути от опций.

3. Как внести изменения в уже сделанный коммит?

Отмена может потребоваться, если вы сделали коммит слишком рано, например, забыв добавить какие-то файлы или комментарий к коммиту. Если вы хотите переделать коммит – внесите необходимые изменения, добавьте их

в индекс и сделайте коммит ещё раз, указав параметр `–amend`.

4. Как отменить индексацию файла в Git?

Использовать `git reset HEAD <file>...` для исключения из индекса.

5. Как отменить изменения в файле?

Использовать `git checkout -- <file>` для возвращения к версии из последнего коммита.

6. Что такое удаленный репозиторий Git?

Удалённые репозитории представляют собой версии вашего проекта, сохранённые в интернете или ещё где-то в сети.

7. Как выполнить просмотр удаленных репозиториях данного локального репозитория?

Для того, чтобы просмотреть список настроенных удалённых репозиториях, вы можете запустить команду `git remote`. Она выведет названия доступных удалённых репозиториях. Если вы клонировали репозиторий, то увидите как минимум `origin` – имя по умолчанию, которое Git даёт серверу, скоторого производилось клонирование.

8. Как добавить удаленный репозиторий для данного локального репозитория?

Для того, чтобы добавить удалённый репозиторий и присвоить ему имя (`shortname`), просто выполните команду `git remote add <shortname> <url>`.

9. Как выполнить отправку/получение изменений с удаленного репозитория?

Для получения данных из удалённых проектов, следует выполнить `git fetch [remote-name]`.

Когда вы хотите поделиться своими наработками, вам необходимо отправить их в удалённый репозиторий. Команда для этого действия простая: `git push <remote-name> <branch-name>`.

10. Как выполнить просмотр удаленного репозитория?

Если хотите получить побольше информации об одном из удалённых репозиториях, вы можете использовать команду `git remote show <remote>`. Она выдаёт URL удалённого репозитория, а также информацию об отслеживаемых ветках.

11. Каково назначение тэгов Git?

Как и большинство СКВ, Git имеет возможность пометить определённые моменты в истории как важные. Как правило, эта функциональность используется для отметки моментов выпуска версий (v1.0, и т. п.). Такие пометки в Git называются тегами.

12. Как осуществляется работа с тэгами Git?

Просмотреть список имеющихся тегов в Git можно очень просто.

Достаточно набрать команду `git tag` (параметры `-l` и `--list` опциональны).

Создание аннотированного тега в Git выполняется легко. Самый простой способ – это указать - а при выполнении команды `tag`.

По умолчанию, команда `git push` не отправляет теги на удалённые сервера. После создания теги нужно отправлять явно на удалённый сервер.

Процесс аналогичен отправке веток — достаточно выполнить команду `git push origin <tagname>`.

Для удаления тега в локальном репозитории достаточно выполнить команду `git tag -d <tagname>`.

Если вы хотите получить версии файлов, на которые указывает тег, то вы можете сделать `git checkout` для тега. Однако, это переведёт репозиторий в состояние «detached HEAD», которое имеет ряд неприятных побочных

эффектов.

13. Самостоятельно изучите назначение флага `--prune` в командах `git fetch` и `git push`. Каково назначение этого флага?

Исходя из описания, предоставленного `git help fetch: --prune` используется для удаления ссылок удаленного отслеживания, которые больше не существуют в удаленном репозитории, а из описания, предоставленного `git help push: --prune` используется для удаления ветвей на удаленном репозитории, для которых нет аналога в локальном репозитории.

Вывод: в результате выполнения работы были исследованы возможности Git для работы с локальными репозиториями.