

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

Институт цифрового развития  
Кафедра инфокоммуникаций

ОТЧЁТ

по лабораторной работе №1.2

**Дисциплина: «Программирование на Python»**  
**Тема: «Исследование возможностей Git для работы с локальными репозиториями»**

Выполнил:  
Епифанов Алексей Александрович  
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 Роман Александрович

---

(подпись)

Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_  
Ставрополь, 2023 г.

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

Порядок выполнения работы:

1. Создал новый репозиторий:

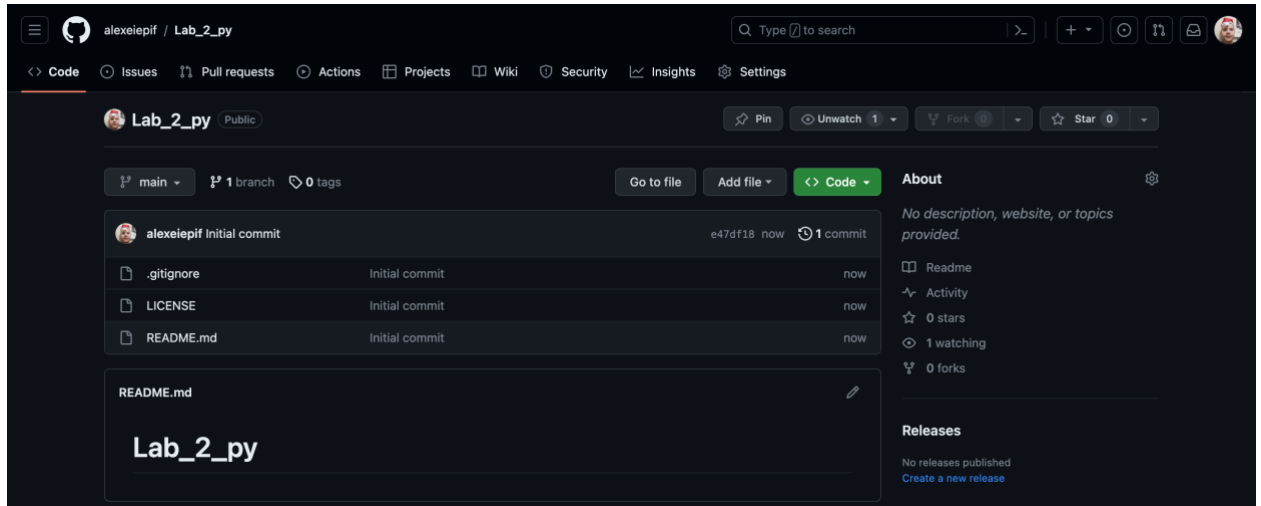


Рисунок 1. Новый репозиторий Lab\_2\_py

2. Проработал примеры лабораторной работы

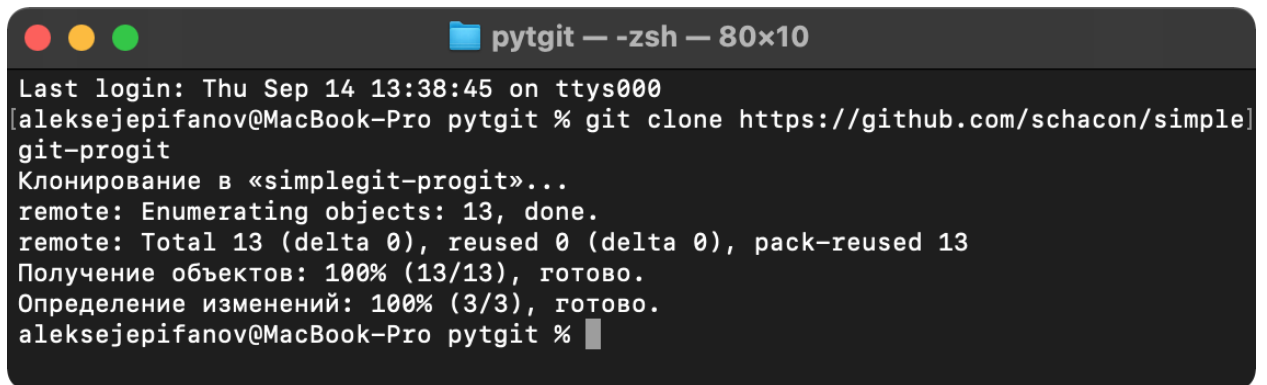


Рисунок 2. Клонирование репозитория

```
simplegit-progit — -zsh — 80x24
Last login: Thu Sep 14 13:52:19 on ttys000
aleksejepifanov@MacBook-Pro simplegit-progit % git log
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

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
aleksejepifanov@MacBook-Pro simplegit-progit %
```

Рисунок 3. Результат работы команды git log

```
simplegit-progit — -zsh — 80x45
Last login: Thu Sep 14 14:03:41 on ttys000
aleksejepifanov@MacBook-Pro simplegit-progit % 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
aleksejepifanov@MacBook-Pro simplegit-progit %
```

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

```
simplegit-progit — -zsh — 80x31
[aleksejepifanov@MacBook-Pro simplegit-progit % 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(+)
aleksejepifanov@MacBook-Pro simplegit-progit %
```

Рисунок 5. Результат работы команды git log --stat

```
simplegit-progit — -zsh — 113x5
aleksejepifanov@MacBook-Pro simplegit-progit % git log --pretty=oneline
ca82a6dff817ec66f44342007202690a93763949 (HEAD -> master, origin/master, origin/HEAD) changed the verison number
085bb3bcb608e1e8451d4b2432f8ecbe6306e7e7 removed unnecessary test code
a11bef06a3f659402fe7563abf99ad00de2209e6 first commit
aleksejepifanov@MacBook-Pro simplegit-progit %
```

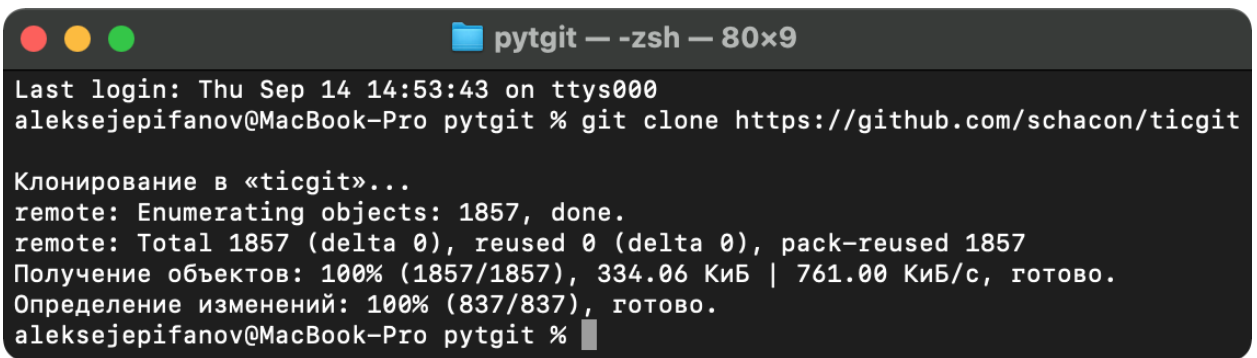
Рисунок 6. Результат работы команды git log --pretty=oneline

```
simplegit-progit — -zsh — 113x5
aleksejepifanov@MacBook-Pro simplegit-progit % git log --pretty=format:"%h - %an, %ar : %s"
ca82a6d - Scott Chacon, 16 лет назад : changed the verison number
085bb3b - Scott Chacon, 16 лет назад : removed unnecessary test code
a11bef0 - Scott Chacon, 16 лет назад : first commit
aleksejepifanov@MacBook-Pro simplegit-progit %
```

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

```
simplegit-progit — -zsh — 113x5
aleksejepifanov@MacBook-Pro simplegit-progit % git log --pretty=format:"%h %s" --graph
* ca82a6d changed the verison number
* 085bb3b removed unnecessary test code
* a11bef0 first commit
aleksejepifanov@MacBook-Pro simplegit-progit %
```

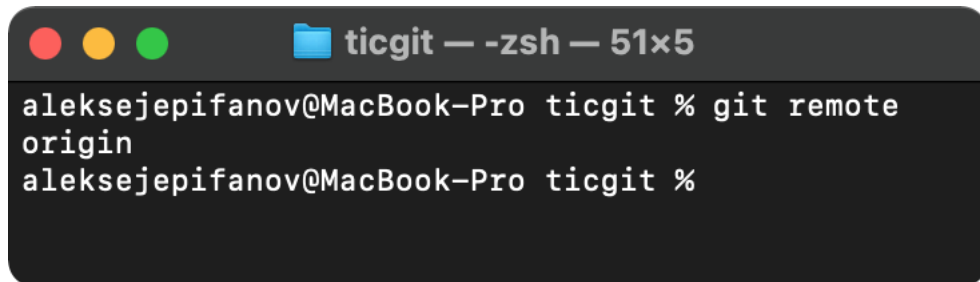
Рисунок 8. Результат работы команды  
git log --pretty=format:"%h %s" --graph



```
Last login: Thu Sep 14 14:53:43 on ttys000
aleksejepifanov@MacBook-Pro pytgit % git clone https://github.com/schacon/ticgit

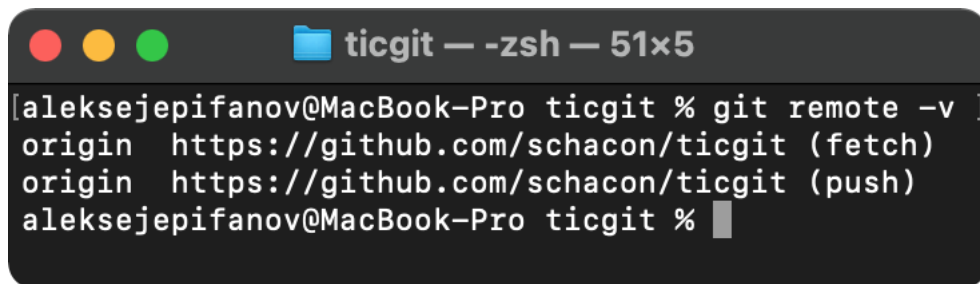
Клонирование в «ticgit»...
remote: Enumerating objects: 1857, done.
remote: Total 1857 (delta 0), reused 0 (delta 0), pack-reused 1857
Получение объектов: 100% (1857/1857), 334.06 КиБ | 761.00 КиБ/с, готово.
Определение изменений: 100% (837/837), готово.
aleksejepifanov@MacBook-Pro pytgit %
```

Рисунок 9. Клонирование репозитория ticgit



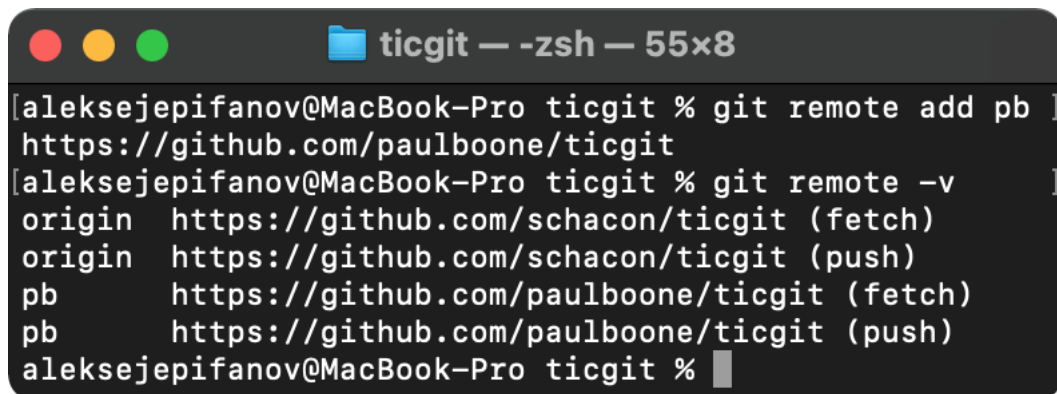
```
aleksejepifanov@MacBook-Pro ticgit % git remote
origin
aleksejepifanov@MacBook-Pro ticgit %
```

Рисунок 10. Результат работы команды git remote



```
[aleksejepifanov@MacBook-Pro ticgit % git remote -v ]
origin https://github.com/schacon/ticgit (fetch)
origin https://github.com/schacon/ticgit (push)
aleksejepifanov@MacBook-Pro ticgit %
```

Рисунок 11. Результат работы команды git remote -v



```
[aleksejepifanov@MacBook-Pro ticgit % git remote add pb ]
https://github.com/paulboone/ticgit
[aleksejepifanov@MacBook-Pro ticgit % 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)
aleksejepifanov@MacBook-Pro ticgit %
```

Рисунок 12. Результат работы команды

git remote add pb https://github.com/paulboone/ticgit

```
ticgit — -zsh — 72x9
[aleksejefifanov@MacBook-Pro ticgit % 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
Распаковка объектов: 100% (43/43), 5.99 КиБ | 139.00 КиБ/с, готово.
Из https://github.com/paulboone/ticgit
* [новая ветка]      master    -> pb/master
* [новая ветка]      ticgit     -> pb/ticgit
aleksejefifanov@MacBook-Pro ticgit %
```

Рисунок 13. Результат работы команды git fetch pb

```
ticgit — -zsh — 80x14
Last login: Sun Sep 17 18:48:15 on ttys004
[aleksejefifanov@MacBook-Pro ticgit % git remote show origin
* внешний репозиторий origin
  URL для извлечения: https://github.com/schacon/ticgit
  URL для отправки: https://github.com/schacon/ticgit
  HEAD ветка: master
  Внешние ветки:
    master отслеживается
    ticgit отслеживается
  Локальная ветка, настроенная для «git pull»:
    master будет слита с внешней веткой master
  Локальная ссылка, настроенная для «git push»:
    master будет отправлена в master (уже актуальна)
aleksejefifanov@MacBook-Pro ticgit %
```

Рисунок 14. Результат работы команды git remote show origin

```
ticgit — -zsh — 80x6
[aleksejefifanov@MacBook-Pro ticgit % git remote rename pb paul
Renaming remote references: 100% (2/2), готово.
[aleksejefifanov@MacBook-Pro ticgit % git remote
origin
paul
aleksejefifanov@MacBook-Pro ticgit %
```

Рисунок 15. Результат работы команды git remote rename pb paul

```
ticgit — -zsh — 80x5
[aleksejefifanov@MacBook-Pro ticgit % git remote remove paul
[aleksejefifanov@MacBook-Pro ticgit % git remote
origin
aleksejefifanov@MacBook-Pro ticgit %
```

Рисунок 16. Результат работы команды git remote remove paul

```
ticgit — -zsh — 80x8
Last login: Tue Sep 19 18:10:29 on ttys000
aleksejepifanov@MacBook-Pro ticgit % git tag
aleksejepifanov@MacBook-Pro ticgit % git tag -l "v1.8.5*"
aleksejepifanov@MacBook-Pro ticgit % git tag -a v1.4 -m "my version 1.4"
aleksejepifanov@MacBook-Pro ticgit % git tag
v1.4
aleksejepifanov@MacBook-Pro ticgit %
```

Рисунок 17. Создал аннотированный тег

```
ticgit — -zsh — 124x28
Last login: Tue Sep 19 18:38:12 on ttys000
aleksejepifanov@MacBook-Pro ticgit % git show v1.4
tag v1.4
Tagger: alexeiepip <aled2093746@gmail.com>
Date: Tue Sep 19 18:34:37 2023 +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
+
## TicGit-ng ##

This project provides a ticketing system built on Git that is kept in a
aleksejepifanov@MacBook-Pro ticgit %
```

Рисунок 18. Результат работы команды git show v1.4

```
ticgit — -zsh — 52x5
aleksejepifanov@MacBook-Pro ticgit % git tag -d v1.4

Метка «v1.4» удалена (была 11193ca)
aleksejepifanov@MacBook-Pro ticgit % git tag
aleksejepifanov@MacBook-Pro ticgit %
```

Рисунок 19. Удаление ранее созданного тега

### 3. Клонировал ранее созданный репозиторий

```
pytgit — -zsh — 80x11
Last login: Tue Sep 19 19:43:34 on ttys000
aleksejepifanov@MacBook-Pro pytgit % git clone https://github.com/alexeiepip/Lab_2_py.git
Клонирование в «Lab_2_py»...
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 8 (delta 1), reused 0 (delta 0), pack-reused 0
Получение объектов: 100% (8/8), готово.
Определение изменений: 100% (1/1), готово.
aleksejepifanov@MacBook-Pro pytgit %
```

Рисунок 20. Клонирование репозитория Lab\_2\_py



#### 4. Дополнил .gitignore

```
#.idea/  
**/.DS_Store  
.vscode
```

Рисунок 21. Добавленные строки .gitignore

#### 5. Написал небольшую программу на python

The screenshot shows a terminal window on the left and a code editor on the right. The terminal displays the output of four git commits, each adding a new file and pushing it to a remote repository. The code editor shows a Python program named 'program.py' that uses numpy to generate a 10x10 matrix and print it.

```
Last login: Tue Sep 19 23:01:51 on tty000  
aleksejefifanov@MacBook-Pro Lab_2_py % git commit -am "commit1"  
[main f6d78bb] commit1  
1 file changed, 8 deletions(-)  
aleksejefifanov@MacBook-Pro Lab_2_py % git push  
Перечисление объектов: 7, готово.  
Подсчет объектов: 100% (7/7), готово.  
При сжатии изменений используется до 8 потоков  
Сжатие объектов: 100% (2/2), готово.  
Запись объектов: 100% (4/4), 316 байтов | 316.00 КиБ/с, готово.  
Всего 4 (изменений 1), повторно использовано 0 (изменений 0), повторно использовано пакетов 0  
remote: Resolving deltas: 100% (1/1), completed with 1 local object.  
To https://github.com/aleksejefif/Lab_2_py.git  
a3a5afa..f6d78bb main -> main  
aleksejefifanov@MacBook-Pro Lab_2_py % git commit -am "commit2"  
[main 9db84a8] commit2  
1 file changed, 1 insertion(+)  
aleksejefifanov@MacBook-Pro Lab_2_py % git push  
Перечисление объектов: 7, готово.  
Подсчет объектов: 100% (7/7), готово.  
При сжатии изменений используется до 8 потоков  
Сжатие объектов: 100% (2/2), готово.  
Запись объектов: 100% (4/4), 337 байтов | 337.00 КиБ/с, готово.  
Всего 4 (изменений 1), повторно использовано 0 (изменений 0), повторно использовано пакетов 0  
remote: Resolving deltas: 100% (1/1), completed with 1 local object.  
To https://github.com/aleksejefif/Lab_2_py.git  
f5d78bb..9db84a8 main -> main  
aleksejefifanov@MacBook-Pro Lab_2_py % git commit -am "commit3"  
[main 62066d4] commit3  
1 file changed, 7 insertions(+)  
aleksejefifanov@MacBook-Pro Lab_2_py % git push  
Перечисление объектов: 7, готово.  
Подсчет объектов: 100% (7/7), готово.  
При сжатии изменений используется до 8 потоков  
Сжатие объектов: 100% (3/3), готово.  
Запись объектов: 100% (4/4), 401 байт | 401.00 КиБ/с, готово.  
Всего 4 (изменений 2), повторно использовано 0 (изменений 0), повторно использовано пакетов 0  
remote: Resolving deltas: 100% (1/1), completed with 1 local object.  
To https://github.com/aleksejefif/Lab_2_py.git  
9db84a8..62066d4 main -> main  
aleksejefifanov@MacBook-Pro Lab_2_py % git commit -am "commit4"  
[main 53b81a9] commit4  
1 file changed, 2 insertions(+), 1 deletion(-)  
aleksejefifanov@MacBook-Pro Lab_2_py % git push  
Перечисление объектов: 7, готово.  
Подсчет объектов: 100% (7/7), готово.  
При сжатии изменений используется до 8 потоков  
Сжатие объектов: 100% (3/3), готово.  
Запись объектов: 100% (4/4), 359 байтов | 359.00 КиБ/с, готово.  
Всего 4 (изменений 2), повторно использовано 0 (изменений 0), повторно использовано пакетов 0  
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.  
To https://github.com/aleksejefif/Lab_2_py.git  
62066d4..53b81a9 main -> main  
aleksejefifanov@MacBook-Pro Lab_2_py %
```

```
1 import numpy as nmp  
2 print("hello ivt")  
3 print("it is my program")  
4 a = []  
5 for i in range(1,11):  
6     b = []  
7     for j in range(1,11):  
8         b.append(i*j)  
9     a.append(b)  
10 print(nmp.matrix(a))  
11
```

Run program x

```
/usr/local/bin/python3 /Users/aleksejefifanov/Desktop/нары_3_сем/pytgt  
hello ivt  
it is my program  
[[ 1  2  3  4  5  6  7  8  9 10]  
 [ 2  4  6  8 10 12 14 16 18 20]  
 [ 3  6  9 12 15 18 21 24 27 30]  
 [ 4  8 12 16 20 24 28 32 36 40]  
 [ 5 10 15 20 25 30 35 40 45 50]  
 [ 6 12 18 24 30 36 42 48 54 60]  
 [ 7 14 21 28 35 42 49 56 63 70]  
 [ 8 16 24 32 40 48 56 64 72 80]  
 [ 9 18 27 36 45 54 63 72 81 90]  
 [10 20 30 40 50 60 70 80 90 100]]
```

Process finished with exit code 0

Рисунок 22. Первые 4 коммита и программа program

The screenshot shows a terminal window on the left and a code editor on the right. The terminal displays the output of three git commits, each adding a new file and pushing it to a remote repository. The code editor shows the updated Python program 'program.py' which now includes a print statement for the multiplication table and a tag for the commit.

```
aleksejefifanov@MacBook-Pro Lab_2_py % git commit -am "commit5"  
[main 6b9411b] commit5  
1 file changed, 1 insertion(+)  
aleksejefifanov@MacBook-Pro Lab_2_py % git tag -a v1.0 -m "5commit"  
fatal: метка «v1.0» уже существует  
aleksejefifanov@MacBook-Pro Lab_2_py % git tag -a v1.1 -m "5commit"  
aleksejefifanov@MacBook-Pro Lab_2_py % git commit -am "commit6"  
[main 10bcf36] commit6  
1 file changed, 1 insertion(+)  
aleksejefifanov@MacBook-Pro Lab_2_py % git tag -a v1.2 -m "6commit"  
aleksejefifanov@MacBook-Pro Lab_2_py % git commit -am "commit7"  
[main 75ac5bf] commit7  
1 file changed, 1 insertion(+), 1 deletion(-)  
aleksejefifanov@MacBook-Pro Lab_2_py % git tag -a v1.3 -m "7commit"  
aleksejefifanov@MacBook-Pro Lab_2_py %
```

```
2 print("hello ivt")  
3 print("it is my program")  
4 print("Таблица умножения чисел от 1 до 10")  
5 a = []  
6 for i in range(1,11):  
7     b = []  
8     for j in range(1,11):  
9         b.append(i*j)  
10    a.append(b)  
11 print(nmp.matrix(a))  
12 print("Работу выполнил Елифанов А.А.")
```

Рисунок 23. Последние 3 коммита с тегами

#### 6. Просмотрел историю хранилища



```
Lab_2_py — zsh — 97x19
aleksejepifanov@MacBook-Pro Lab_2_py % git log --graph --pretty=oneline --abbrev-commit
* 75ac5bf (HEAD -> main, tag: v1.3, origin/main, origin/HEAD) commit7
* 10bcf36 (tag: v1.2) commit6
* 6b9411b (tag: v1.1) commit5
* 53b81a9 commit4
* 62066d4 commit3
* 9db04a8 commit2
* f5d78bb commit1
* a3a5afa commit2
* 4c0155a commit2
* 2403c7b commit1
* d3d4860 commit1
* d872b84 commit
* 558f5ec commit1
* 40d0ebc commit1
* 72f6ccb readme
* 903a340 Update .gitignore
* e47df18 Initial commit
aleksejepifanov@MacBook-Pro Lab_2_py %
```

Рисунок 24. Результат работы команды `git log --graph --pretty=oneline --abbrev-commit`

## 7. Просмотрел содержимое коммитов

```
Lab_2_py — less ◀ git show d872b84 — 103x57
aleksejepifanov@MacBook-Pro Lab_2_py % git show HEAD
commit 75ac5bf67e217dccb70654f4c4abc0be910dcc0 (HEAD -> main, tag: v1.3, origin/main, origin/HEAD)
Author: alexeiepipif <aled2093746@gmail.com>
Date: Tue Sep 19 23:46:54 2023 +0300

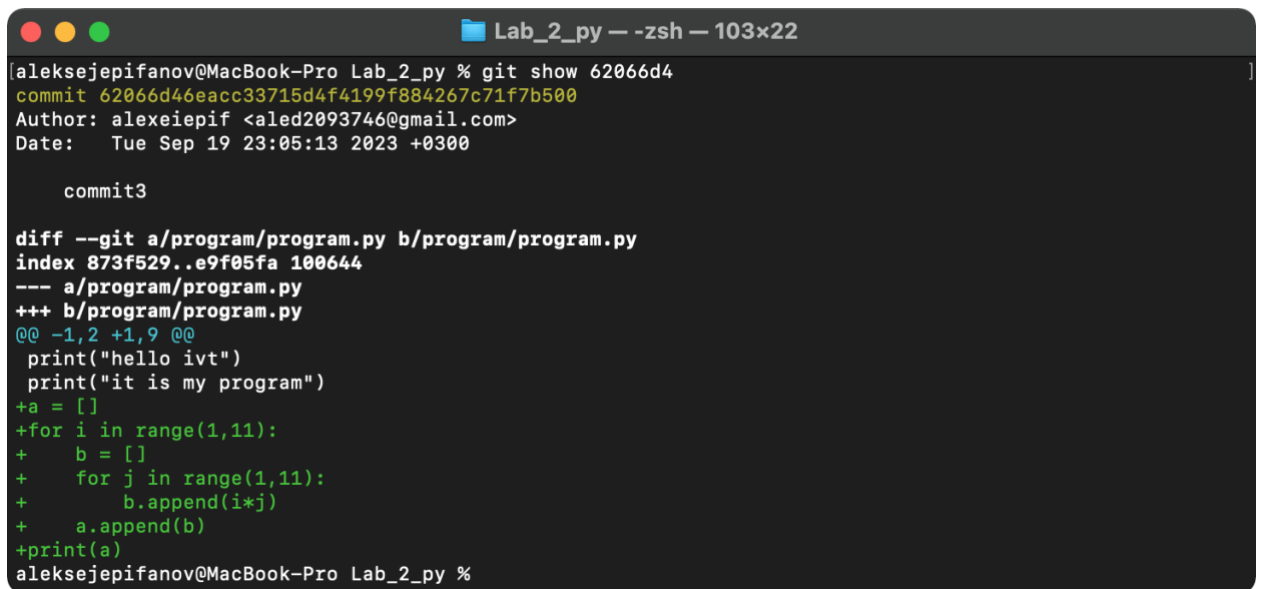
    commit7

diff --git a/program/program.py b/program/program.py
index d3cd8b9..52c2e2c 100644
--- a/program/program.py
+++ b/program/program.py
@@ -1,7 +1,7 @@
import numpy as nmp
print("hello ivt")
print("it is my program")
-print("Таблица умножения")
+print("Таблица умножения чисел от 1 до 10")
a = []
for i in range(1,11):
    b = []
aleksejepifanov@MacBook-Pro Lab_2_py % git show HEAD~1
commit 10bcf36bbaad4a64f41082aa94f6d8055b80291e (tag: v1.2)
Author: alexeiepipif <aled2093746@gmail.com>
Date: Tue Sep 19 23:43:40 2023 +0300

    commit6

diff --git a/program/program.py b/program/program.py
index 9692806..d3cd8b9 100644
--- a/program/program.py
+++ b/program/program.py
@@ -9,3 +9,4 @@ for i in range(1,11):
     b.append(i*j)
    a.append(b)
print(nmp.matrix(a))
+print("Работу выполнил Епифанов А.А.")
\ No newline at end of file
```

Рисунок 25. Результат работы команд `git show HEAD` и `git show HEAD~1`



```
Lab_2_py — -zsh — 103x22
aleksejepifanov@MacBook-Pro Lab_2_py % git show 62066d4
commit 62066d46eacc33715d4f4199f884267c71f7b500
Author: alexeiepip <aled2093746@gmail.com>
Date: Tue Sep 19 23:05:13 2023 +0300

    commit3

diff --git a/program/program.py b/program/program.py
index 873f529..e9f05fa 100644
--- a/program/program.py
+++ b/program/program.py
@@ -1,2 +1,9 @@
 print("hello ivt")
 print("it is my program")
+a = []
+for i in range(1,11):
+    b = []
+    for j in range(1,11):
+        b.append(i*j)
+    a.append(b)
+print(a)
aleksejepifanov@MacBook-Pro Lab_2_py %
```

Рисунок 26. Результат работы команды git show 62066d4

8. Удалил код из файла program.py, а затем удалил все несохраненные изменения

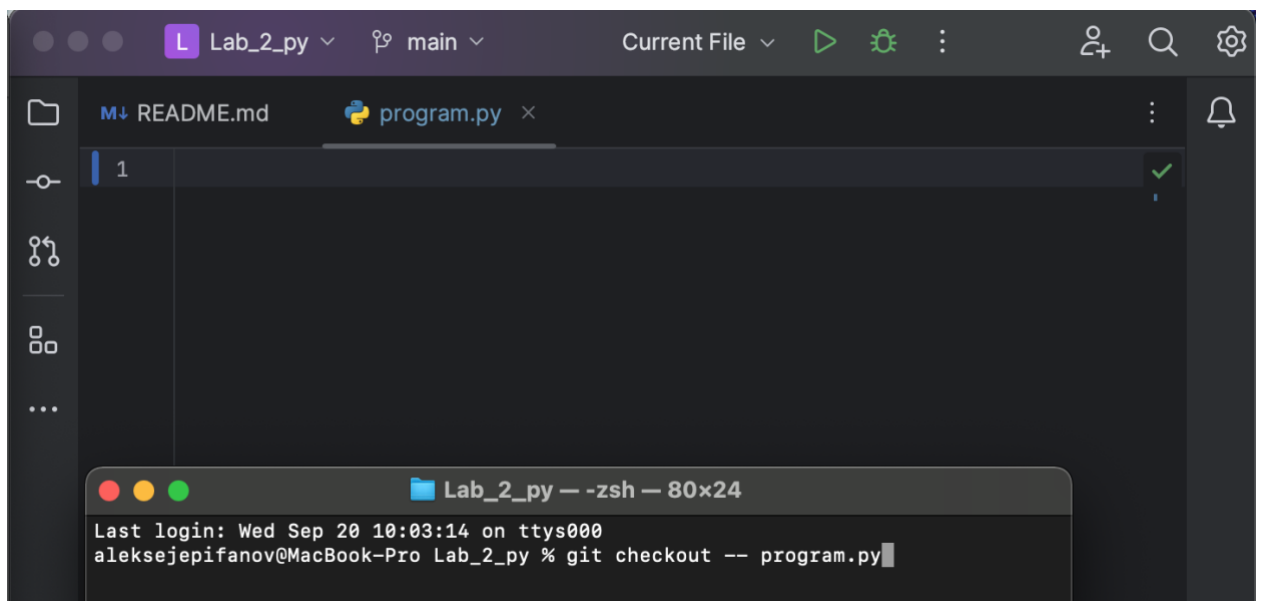


Рисунок 27. Пустой program.py

```
1 import numpy as nmp
2 print("hello ivt")
3 print("it is my program")
4 print("Таблица умножения чисел от 1 до 10")
5 a = []
6 for i in range(1,11):
7     b = []
8     for j in range(1,11):
9         b.append(i*j)
10    a.append(b)
11 print(nmp.matrix(a))
12 print("Работу выполнил Епифанов А.А.")
```

```
Lab_2_py — zsh — 80x24
Last login: Wed Sep 20 10:03:14 on ttys000
[aleksejepifanov@MacBook-Pro Lab_2_py % git checkout -- program.py
error: pathspec 'program.py' did not match any file(s) known to git
[aleksejepifanov@MacBook-Pro Lab_2_py % git checkout -- program/program.py
aleksejepifanov@MacBook-Pro Lab_2_py %
```

Рисунок 28. Код вернулся после команды `git checkout -- program/program.py`

- Удалил весь код из файла `program.py`, а затем сделал коммит

```
1
```

```
Lab_2_py — zsh — 80x24
[aleksejepifanov@MacBook-Pro Lab_2_py % git commit -am "badcommit"
[main 1a1019d] badcommit
1 file changed, 12 deletions(-)
aleksejepifanov@MacBook-Pro Lab_2_py %
```

Рисунок 29. Коммит после удаления кода

- Откатил состояние хранилища к предыдущей версии коммита

```
1 import numpy as nmp
2 print("hello ivt")
3 print("it is my program")
4 print("Таблица умножения чисел от 1 до 10")
5 a = []
6 for i in range(1,11):
7     b = []
8     for j in range(1,11):
9         b.append(i*j)
10    a.append(b)
11 print(nmp.matrix(a))
12 print("Работу выполнил Епифанов А.А.")
```

```
Lab_2_py — zsh — 80x24
aleksejepifanov@MacBook-Pro Lab_2_py % git commit -am "badcommit"
[main 1a1019d] badcommit
1 file changed, 12 deletions(-)
aleksejepifanov@MacBook-Pro Lab_2_py % git reset --hard HEAD~1
Указатель HEAD сейчас на коммите 75ac5bf commit7
aleksejepifanov@MacBook-Pro Lab_2_py %
```

Рисунок 30. 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 за раз.

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, с которого 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 `-a` 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 локальной 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 удаленной репозитории, а из описания, предоставленного `git help push: --prune` используется для удаления ветвей на удаленной репозитории, для которых нет аналога в локальной репозитории.

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