

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 №1.2**  
**дисциплины «Введение в специальность»**  
**Вариант \_\_\_\_**

Выполнил:

Иванов Иван Иванович

1 курс, группа ИВЕ-б-о-21-1,

11.03.02 «Информатика и вычислительная  
техника», направленность (профиль)

«Инфокоммуникационные системы и  
сети», очная форма обучения

---

(подпись)

Руководитель практики:

Братченко Н.Ю., канд. физ.-мат. наук,  
доцент, \_\_\_\_\_ доцент \_\_\_\_\_ кафедры  
инфокоммуникаций

---

(подпись)

Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_

Ставрополь, 2021 г.

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


## Порядок выполнения работы:

### 1. Создаем новый репозиторий

#### Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner \*

 Arsen445 ▾

Repository name \*

LB1.2 ✓

Great repository names are short and memorable. Need inspiration? How about [legendary-octo-spoon](#)?

Description (optional)

☒  Public

Anyone on the internet can see this repository. You choose who can commit.

☐  Private

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☒ Add a README file

This is where you can write a long description for your project. [Learn more.](#)

☒ Add .gitignore


Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: Python ▾

☒ Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

License: MIT License ▾

This will set  main as the default branch. Change the default name in your [settings](#).

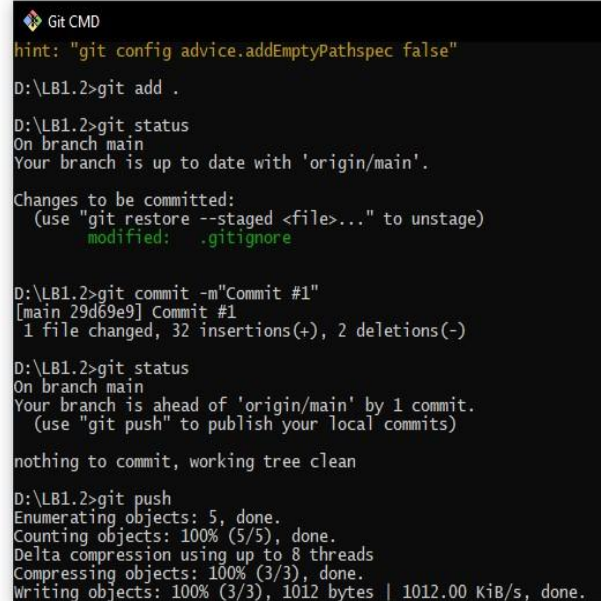
Create repository

### 2. Клонировем репозиторий

```
D:\>git clone https://github.com/Arsen445/LB1.2.git
Cloning into 'LB1.2'...
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.
D:\>
```

### 3. Редактируем gitignore

```
1
2 # Created by https://www.toptal.com/developers/gitignore/api/python
3 # Edit at https://www.toptal.com/developers/gitignore?templates=python
4
5 ### Python ###
6 # Byte-compiled / optimized / DLL files
7 __pycache__/
8 *.py[cod]
9 *$py.class
10
11 # C extensions
12 *.so
13
14 # Distribution / packaging
15 .Python
16 build/
17 develop-eggs/
18 dist/
19 downloads/
20 eggs/
21 .eggs/
22 lib/
23 lib64/
24 parts/
25 sdist/
26 var/
27 wheels/
28 share/python-wheels/
29 *.egg-info/
30 .installed.cfg
31 *.egg
32 MANIFEST
```



```
Git CMD
hint: "git config advice.addEmptyPathsSpec false"
D:\LB1.2>git add .
D:\LB1.2>git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   .gitignore

D:\LB1.2>git commit -m"Commit #1"
[main 29d69e9] Commit #1
1 file changed, 32 insertions(+), 2 deletions(-)

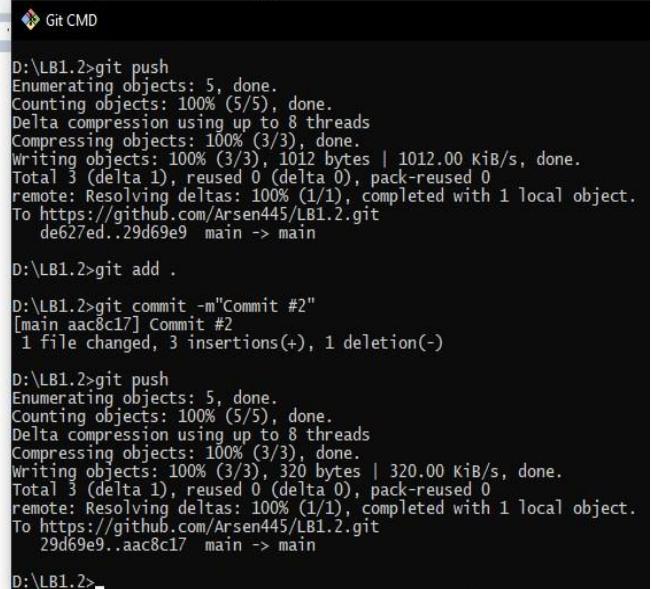
D:\LB1.2>git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
(use 'git push' to publish your local commits)

nothing to commit, working tree clean

D:\LB1.2>git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 1012 bytes | 1012.00 KiB/s, done.
```

### 4. Добавляем readme с информацией о группе и ФИО

Шрифт	Абзац
1 2 3 4 5 6 7 8 9	
# LB1.2	
Эсеналиев Арсен ИВТ-6-о-21-1	
Кроссплатформенное программирование	



```
Git CMD
D:\LB1.2>git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 1012 bytes | 1012.00 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/Arsen445/LB1.2.git
   de627ed..29d69e9  main -> main

D:\LB1.2>git add .
D:\LB1.2>git commit -m"Commit #2"
[main aac8c17] Commit #2
1 file changed, 3 insertions(+), 1 deletion(-)

D:\LB1.2>git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 320 bytes | 320.00 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/Arsen445/LB1.2.git
   29d69e9..aac8c17  main -> main

D:\LB1.2>
```

### 5. Создаем комит с тэгом

```
a=3 D:\LB1.2>git tag -a v0.0 -m "my version 0.0"
D:\LB1.2>git tag --list
v0.0
D:\LB1.2>_
```

```
a=3 D:\LB1.2>git tag -a v0.1 -m "my version 0.1"
b=4 D:\LB1.2>git tag --list
v0.0
v0.1
D:\LB1.2>
```

```
a=3 D:\LB1.2>git tag -a v0.2 -m "my version 0.2"
b=4 D:\LB1.2>git tag --list
a, b = b, a v0.0
v0.1
v0.2
D:\LB1.2>
```

```
a=3 D:\LB1.2>git tag -a v0.3 -m "my version 0.3"
b=4 D:\LB1.2>git tag --list
a, b = b, a v0.0
print(a) v0.1
v0.2
v0.3
D:\LB1.2>
```

```
a=3 D:\LB1.2>git tag -a v0.4 -m "my version 0.4"
b=4 D:\LB1.2>git tag --list
a, b = b, a v0.0
print(a) v0.1
print(b) v0.2
v0.3
v0.4
D:\LB1.2>
```

```
a=3 D:\LB1.2>git tag -a v0.5 -m "my version 0.5"
b=4 D:\LB1.2>git tag --list
a, b = b, a v0.0
print(a) #a=4 v0.1
print(b) v0.2
v0.3
v0.4
v0.5
D:\LB1.2>
```

```
a=3 D:\LB1.2>git tag -a v0.6 -m "my version 0.6"
b=4 D:\LB1.2>git tag --list
a, b = b, a v0.0
print(a) #a=4 v0.1
print(b) #b=3 v0.2
v0.3
v0.4
v0.5
v0.6
D:\LB1.2>
```

```
D:\LB1.2>git push origin --tags
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (9/9), 716 bytes | 716.00 KiB/s, done.
Total 9 (delta 6), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (6/6), done.
To https://github.com/Arsen445/LB1.2.git
* [new tag]          v0.0 -> v0.0
* [new tag]          v0.1 -> v0.1
* [new tag]          v0.2 -> v0.2
* [new tag]          v0.3 -> v0.3
* [new tag]          v0.4 -> v0.4
* [new tag]          v0.5 -> v0.5
* [new tag]          v0.6 -> v0.6
* [new tag]          v0.7 -> v0.7
* [new tag]          v0.8 -> v0.8
```

Tags		
v0.8	...	...
4 minutes ago	6171aab	zip tar.gz
v0.7	...	...
5 minutes ago	8fb83bc	zip tar.gz
v0.6	...	...
12 minutes ago	aac8c17	zip tar.gz
v0.5	...	...
12 minutes ago	aac8c17	zip tar.gz
v0.4	...	...
13 minutes ago	aac8c17	zip tar.gz
v0.3	...	...
14 minutes ago	aac8c17	zip tar.gz
v0.2	...	...
23 minutes ago	aac8c17	zip tar.gz
v0.1	...	...
24 minutes ago	aac8c17	zip tar.gz

6. Смотрим содержимое коммитов с помощью команды `git show <ref>`

```
D:\LB1.2>git show HEAD
commit 6171aabb92e64abfdaa2deb3d2fbf5756fa27009 (HEAD -> main, tag: v0.8, origin/main, origin/HEAD)
Author: Arsen445 <arsen45454545@gmail.com>
Date: Wed Mar 16 02:52:45 2022 +0300

    Commit #6

diff --git a/gg.py b/gg.py
index 4348ccc..c47df53 100644
--- a/gg.py
+++ b/gg.py
@@ -3,3 +3,5 @@ b=4
 a, b = b, a
 print(a) #a=4
 print(b) #b=3
+##f
+
D:\LB1.2>
```

`git show HEAD`



```

D:\LB1.2>git show HEAD~1
commit 8fb83bce6d5196a38e28440251b66db348185f57 (tag: v0.7)
Author: Arsen445 <arsen454545454@gmail.com>
Date:   Wed Mar 16 02:48:41 2022 +0300

    Commit #4

diff --git a/gg.py b/gg.py
new file mode 100644
index 0000000..4348ccc
--- /dev/null
+++ b/gg.py
@@ -0,0 +1,5 @@
+a=3
+b=4
+a, b = b, a
+print(a) #a=4
+print(b) #b=3
D:\LB1.2>

```

git show HEAD~1

```

D:\LB1.2>git show 29d69e9
commit 29d69e929cd45a40b24c9040225000e890a17e12
Author: Arsen445 <arsen454545454@gmail.com>
Date:   Wed Mar 16 01:29:20 2022 +0300

    Commit #1

diff --git a/.gitignore b/.gitignore
index b6e4761..9659373 100644
--- a/.gitignore
+++ b/.gitignore
@@ -1,3 +1,8 @@
+
+# Created by https://www.toptal.com/developers/gitignore/api/python
+# Edit at https://www.toptal.com/developers/gitignore?templates=python
+
+### Python ###
+# Byte-compiled / optimized / DLL files
+__pycache__/
+*.py[cod]
@@ -20,7 +25,6 @@ parts/
sdist/
var/
wheels/
-pip-wheel-metadata/

```

git

show 29d69e9

- Удаляем код из файла и удаляем все несохраненные изменения с помощью команды `git checkout --pp.py`

```
C:\Users\GG_Force>D:
D:\>cd LB1.2
D:\LB1.2>git checkout --gg.py
error: unknown option --gg.py
usage: git checkout [<options>] <branch>
       or: git checkout [<options>] [<branch>] -- <file>...

    -b <branch>          create and checkout a new branch
    -B <branch>          create/reset and checkout a branch
    -l                   create reflog for new branch
    --guess              second guess 'git checkout <no-such-branch>' (default)
    --overlay            use overlay mode (default)
    -q, --quiet          suppress progress reporting
    --recurse-submodules[=<checkout>] control recursive updating of submodules
    --progress           force progress reporting
    -m, --merge           perform a 3-way merge with the new branch
    --conflict <style>   conflict style (merge, diff3, or zdiff3)
    -d, --detach          detach HEAD at named commit
    -t, --track[=(direct|inherit)] set branch tracking configuration
    -f, --force           force checkout (throw away local modifications)
    --orphan <new-branch> new unparented branch
    --overwrite-ignore   update ignored files (default)
    --ignore-other-worktrees do not check if another worktree is holding the given ref
    -2, --ours            checkout our version for unmerged files
    -3, --theirs          checkout their version for unmerged files
    -p, --patch           select hunks interactively
    --ignore-skip-worktree-bits do not limit pathspecs to sparse entries only
    --pathspec-from-file <file> read pathspec from file
```

## 8. Делаем коммит + новый тэг

### last commit

main

Arsen445 committed 4 minutes ago

Showing 1 changed file with 0 additions and 7 deletions.

7 gg.py

...	@@ -1,7 +0,0 @@
1	- a=3
2	- b=4
3	- a, b = b, a
4	- print(a) #a=4
5	- print(b) #b=3
6	- #f
7	-
...	



```

D:\LB1.2>git add .

D:\LB1.2>git tag -a v0.9 -m "my version 0.9"

D:\LB1.2>git commit -m""
error: switch 'm' requires a value

D:\LB1.2>git commit -m"last commit"
[main 7917946] last commit
1 file changed, 7 deletions(-)

D:\LB1.2>git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 250 bytes | 250.00 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/Arsen445/LB1.2.git
6171aab..7917946 main -> main

D:\LB1.2>git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

D:\LB1.2>git tag --list
v0.0
v0.1
v0.2
v0.3
v0.4
v0.5
v0.6
v0.7
v0.8
v0.9

D:\LB1.2>_

```

9. Откатываемся на предыдущую версию с помощью команды `git reset --hard HEAD~1`

```

D:\LB1.2>git reset --hard
HEAD is now at 7917946 last commit

D:\LB1.2>git reset --hard HEAD~1
HEAD is now at 6171aab Commit #6

D:\LB1.2>_

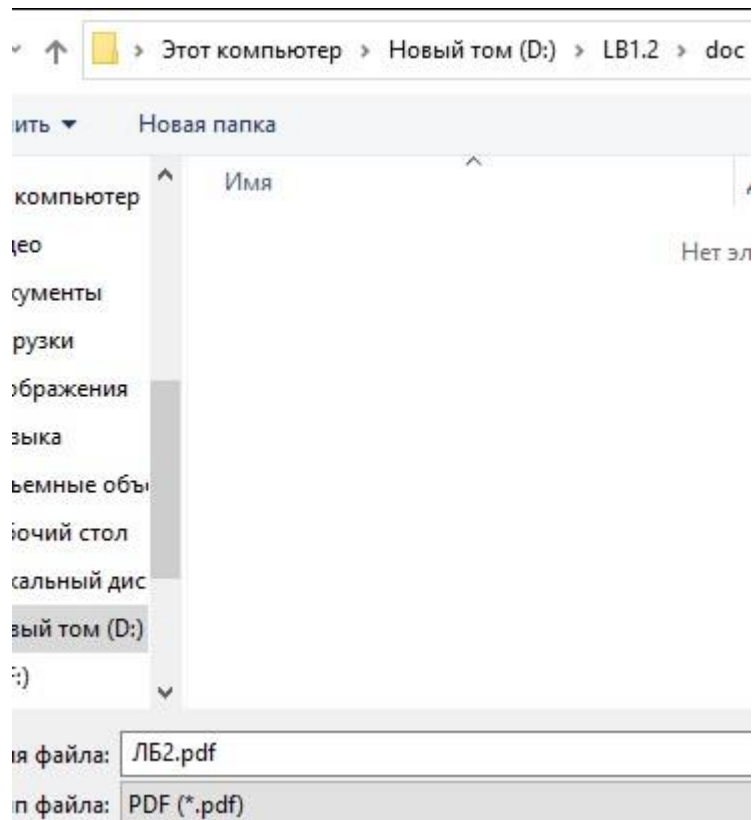
```

```

File Edit Format Run Options Window
a=3
b=4
a, b = b, a
print(a) #a=4
print(b) #b=3
#f

```

10. Проверяем все изменения и сохраняем работу в формате pdf



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

`git log` – для просмотра истории коммитов

`git log (-p или --patch)` позволяет увидеть разницу между коммитами

`git log --stat` краткая статистика (перед выводом коммита кол-во изменений)

`git log -p -2` количество коммитов для вывода

`git log --pretty(смена формата)=oneline short` вывод коммита в одну строку + минимум инфы

`full` среднее кол-во инфы

`fuller` максимум информации

`git log --pretty=format: %h %an .....%t` указание формата

Опция	Описания вывода
%H	Хеш коммита
%h	Сокращенный хеш коммита
%T	Хеш дерева
%t	Сокращенный хеш дерева
%P	Хеш родителей
%p	Сокращенный хеш родителей
%an	Имя автора
%ae	Электронная почта автора
%ad	Дата автора (формат даты можно задать опцией --date=option)
%ar	Относительная дата автора
%cn	Имя коммитера
%ce	Электронная почта коммитера
%cd	Дата коммитера
%cr	Относительная дата коммитера
%s	Содержание

## 2. Как ограничить вывод при просмотре истории коммитов?

git log -p -2 количество комитов для вывода последние 2

git log --since(или --until)=2.weeks последние 2 недели

--author по автору

--grep по ключ слову

-S по изменениям повлекшим добавление или уменьшение количества строк

Опция	Описание
-(n)	Показывает только последние n коммитов.
--since, --after	Показывает только те коммиты, которые были сделаны после указанной даты.
--until, --before	Показывает только те коммиты, которые были сделаны до указанной даты.
--author	Показывает только те коммиты, в которых запись author совпадает с указанной строкой.
--committer	Показывает только те коммиты, в которых запись committer совпадает с указанной строкой.
--grep	Показывает только коммиты, сообщение которых содержит указанную строку.
-S	Показывает только коммиты, в которых изменение в коде повлекло за собой добавление или удаление указанной строки.

## 3. Как внести изменения в уже сделанный коммит?

вносите необходимые изменения, добавьте их в индекс и сделайте коммит ещё раз, указав параметр git commit --amend

## 4. Как отменить индексацию файла в Git?

git reset HEAD <file>... для исключения из индекса. После git status

5. Как отменить изменения в файле?

```
git checkout -- CONTRIBUTING.md
```

```
git status
```

```
git reset HEAD <file>
```

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?

```
git remote
```

8. Как добавить удаленный репозиторий для данного локального репозитория?

```
Git remote add <имя> <ссылка>
```

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

```
Git fetch pb (pull) получение
```

```
git push - отправка
```

10. Как выполнить просмотр удаленного репозитория?

```
git remote show <имя>
```

11. Каково назначение тэгов Git?

Пометка версии продукта

12. Как осуществляется работа с тэгами Git?

```
git tag просмотр списка тегов
```

Легковесный тег — это что-то очень похожее на ветку, которая не изменяется — просто указатель на определённый коммит.

аннотированные теги хранятся в базе данных Git как полноценные объекты. Они имеют контрольную сумму, содержат имя автора, его e-mail и дату создания, имеют комментарий и могут быть подписаны и проверены с помощью GNU Privacy Guard (GPG).

```
git tag -a v1.4 -m "my version 1.4"
```

```
git show v1.4 показать
```

```
git push origin <tagname> или git push origin --tags обмен тегами
```

```
git tag -d v1.4-lw удаление тега
```

```
git push <remote> :refs/tags/<tagname> : удаление тега с внешнего сервера
```

```
git checkout v2.0.0 просмотреть тег
```

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

`Git fetch --prune` команда получения всех изменений с репозитория GitHub.

В команде `git push --prune` удаляет удаленные ветки, у которых нет локального аналога.