

**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 №1.2**

Дисциплина: «Основы кроссплатформенного программирования»

Тема: «Исследование возможностей Git для работы с локальными  
репозиториями»

Выполнил: студент 1 курса

группы ИВТ-б-о-21-1

Богдашов Артём Владимирович

Ставрополь 2022

Выполнение работы:

1. Создал общедоступный репозиторий ger\_1.2 на GitHub в котором будет использована лицензия MIT и выбранный мной язык программирования.

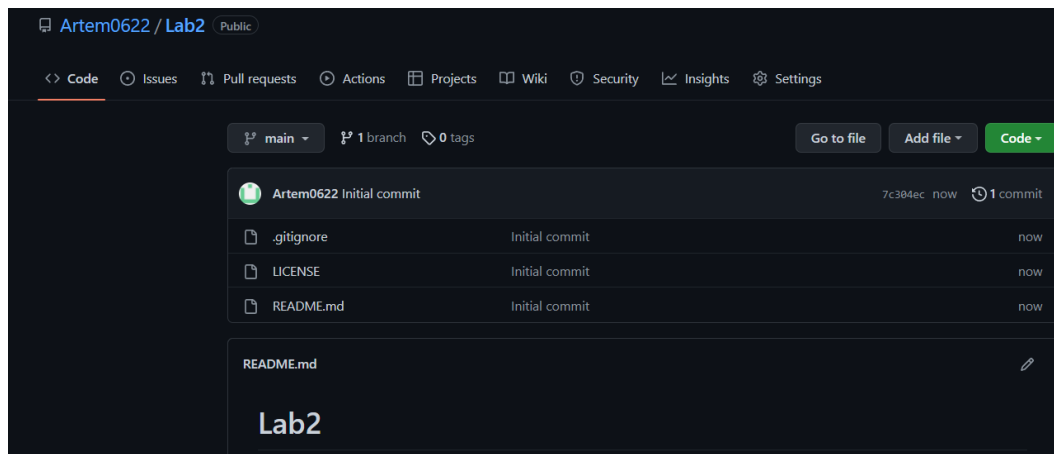


Рисунок 1.1 Созданный репозиторий в GitHub

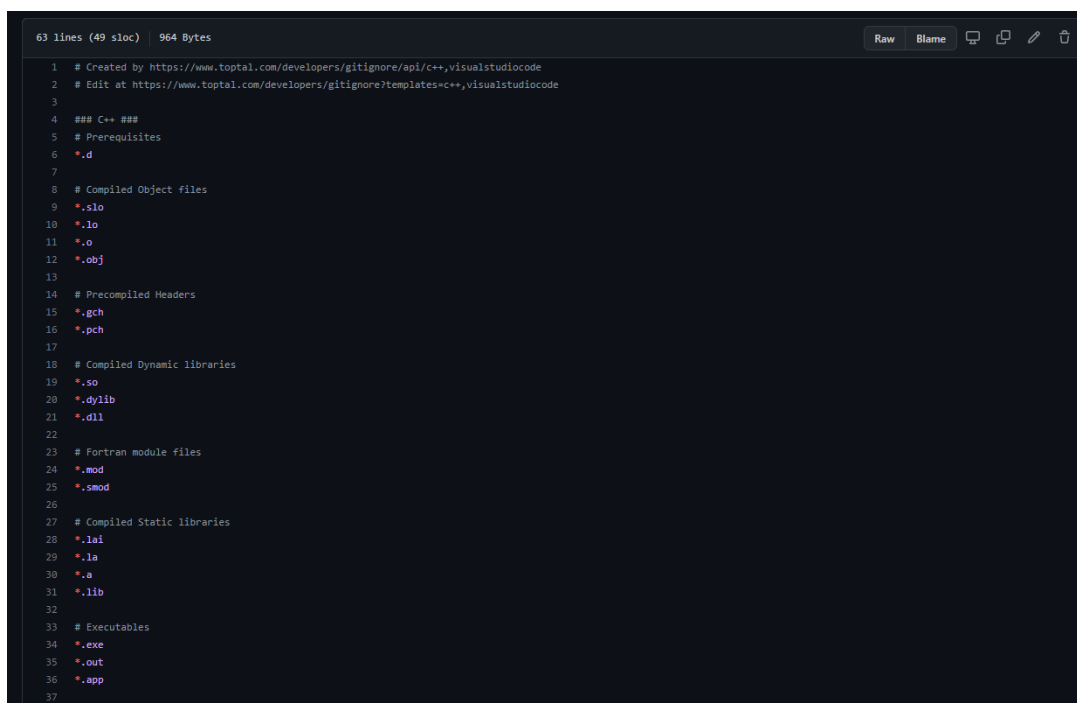


Рисунок 1.2 Изменения в файле .gitignore

2.Клонировал созданный репозиторий на раб. компьютер:

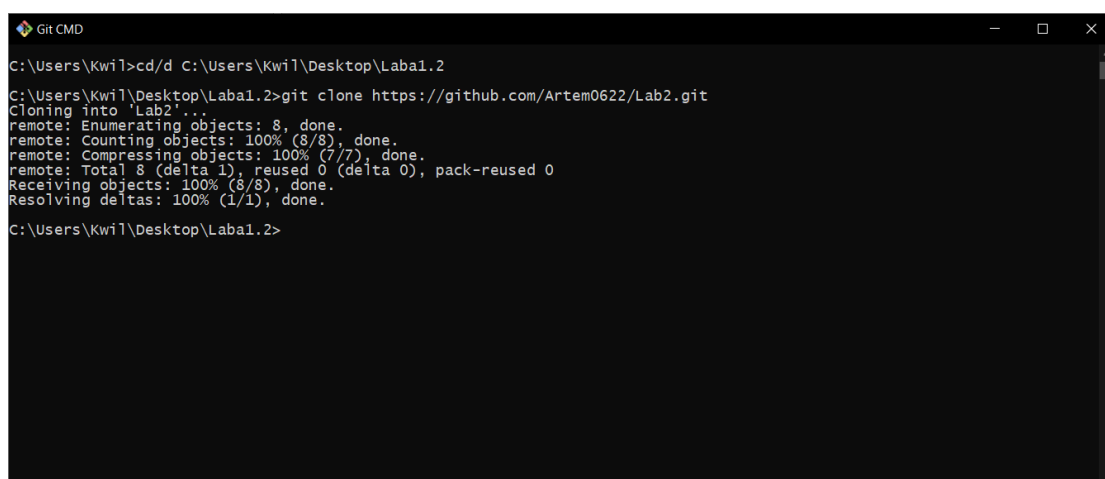


Рисунок 2. Клонирование репозитория

### 3.Добавил информацию в README и закоммитил:

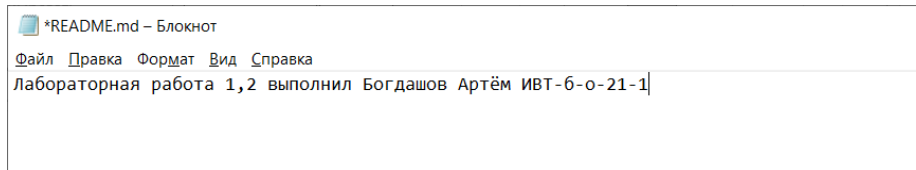


Рисунок 3.1 Добавление информации в README.md

```
Git CMD
nothing to commit, working tree clean

C:\Users\Kw1\Desktop\Lab1.2\Lab2>git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")

C:\Users\Kw1\Desktop\Lab1.2\Lab2>git add .

C:\Users\Kw1\Desktop\Lab1.2\Lab2>git commit -a "R"
fatal: paths 'R ...' with -a does not make sense

C:\Users\Kw1\Desktop\Lab1.2\Lab2>git add .

C:\Users\Kw1\Desktop\Lab1.2\Lab2>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:   README.md

C:\Users\Kw1\Desktop\Lab1.2\Lab2>git git commit -m "mod README"
git: 'git' is not a git command. See 'git --help'.

The most similar command is
    init

C:\Users\Kw1\Desktop\Lab1.2\Lab2>
```

Рисунок 3.2 Коммит файла README.md

```
C:\Users\Kw1\Desktop\Lab1.2\Lab2>git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 340 bytes | 340.00 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/Artem0622/Lab2.git
   40eb0c8..acbf5c0  main -> main

C:\Users\Kw1\Desktop\Lab1.2\Lab2>
```

Рисунок 3.3 Пуш коммита

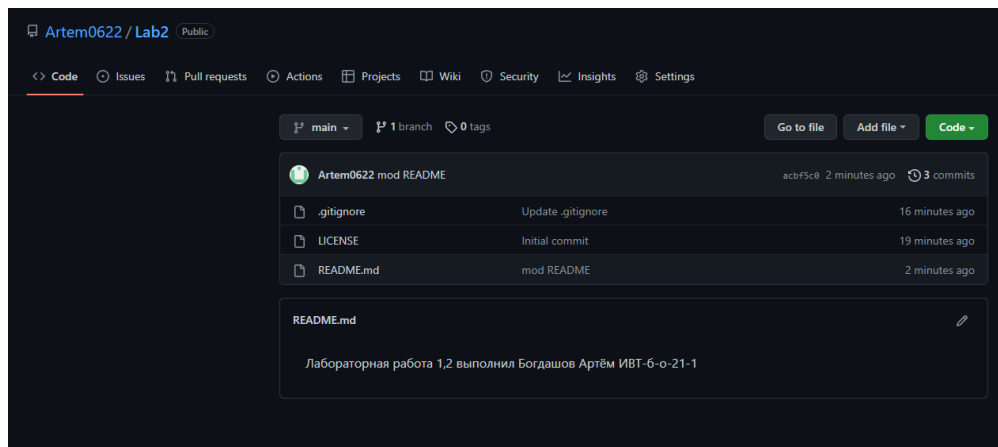


Рисунок 3.4 Изменения на удалённом сервере

4. Написал небольшую программу, сделал коммит и push.

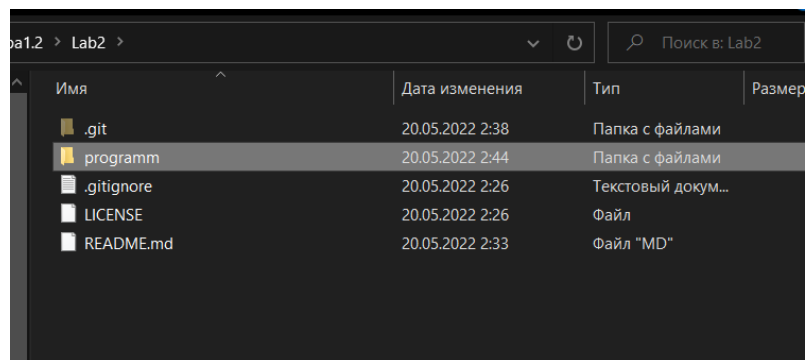


Рисунок 4.1 Добавлена папка с проектом на C++

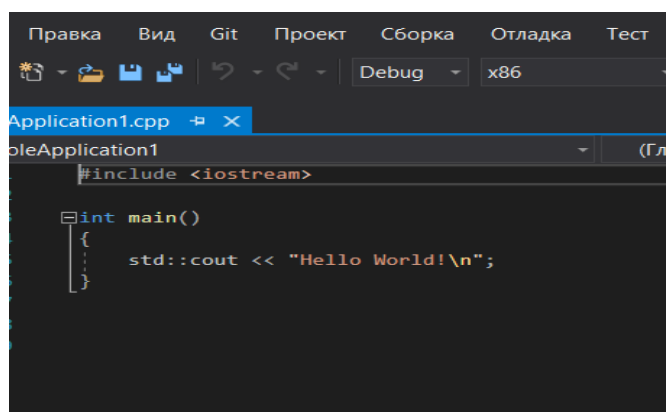


Рисунок 4.2 Изменения в программе

```
Git CMD - "C:\Program Files\Git\cmd\git.exe" push
C:\Users\Kw1\Desktop\Lab1.2\Lab2
C:\Users\Kw1\Desktop\Lab1.2\Lab2>git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
  program/

nothing added to commit but untracked files present (use "git add" to track)
C:\Users\Kw1\Desktop\Lab1.2\Lab2>git add .
C:\Users\Kw1\Desktop\Lab1.2\Lab2>git commit -m "proga"
[main 11cac20] proga
 8 files changed, 212 insertions(+)
 create mode 100644 program/ConsoleApplication1/.vs/ConsoleApplication1/v16/.suo
 create mode 100644 program/ConsoleApplication1/.vs/ConsoleApplication1/v16/Browse.VC.db
 create mode 100644 program/ConsoleApplication1/.vs/ConsoleApplication1/v16/ipch/AutoPCH/8b3740b0fef6e0c0/CONSOLEAPPLICATION1.ipch
 create mode 100644 program/ConsoleApplication1/ConsoleApplication1.sln
 create mode 100644 program/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp
 create mode 100644 program/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.vcxproj
 create mode 100644 program/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.vcxproj.filters
 create mode 100644 program/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.vcxproj.user
C:\Users\Kw1\Desktop\Lab1.2\Lab2>git push
Enumerating objects: 20, done.
Counting objects: 100% (20/20), done.
Delta compression using up to 12 threads
Compressing objects: 100% (14/14), done.
Writing objects: 63% (12/19), 7.93 MiB | 1.11 MiB/s
```

Рисунок 4.3 Коммит и пуш программы на уд. сервер

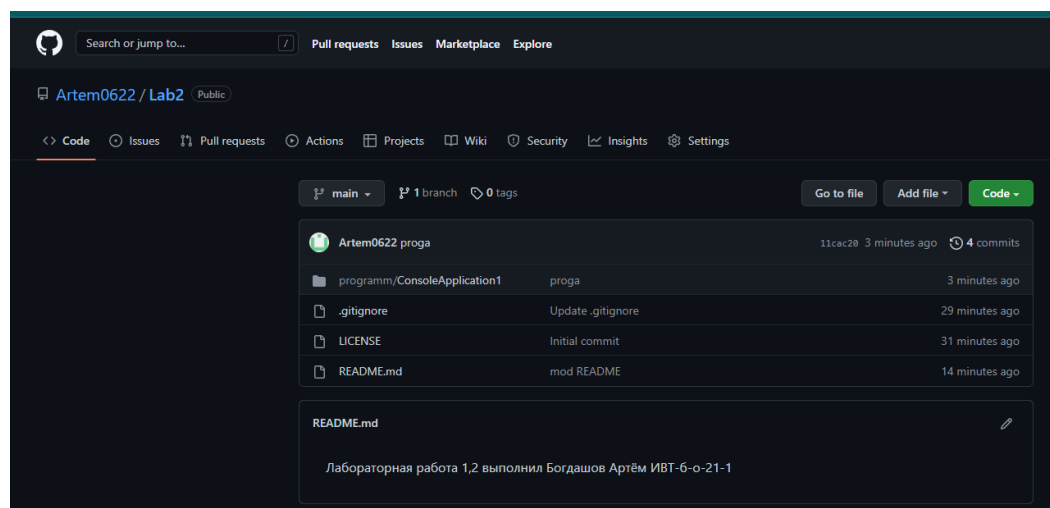


Рисунок 4.4 Изменения на уд. сервере

5.Сделал коммиты в процессе изменения программы, отметил ихтегами и запустил на удалённый сервер коммиты затем теги:

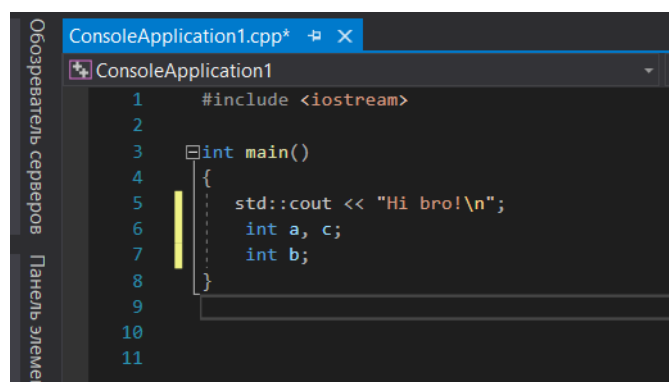
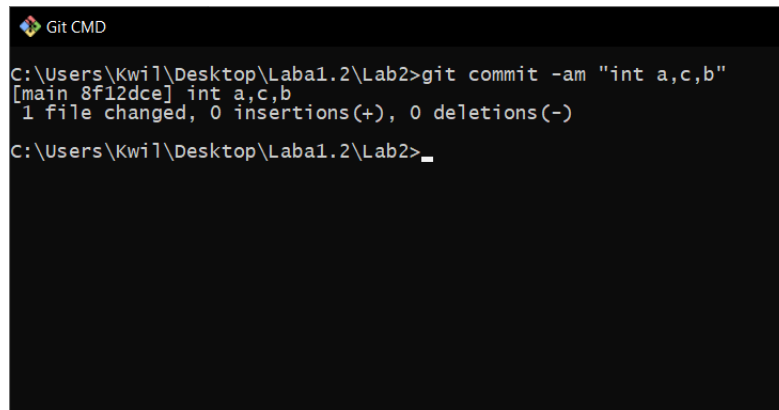
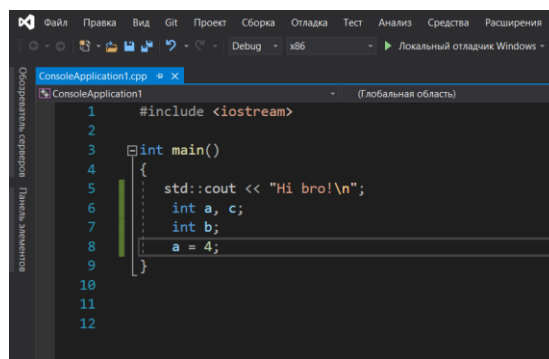


Рисунок 5.1 Изменения в программе



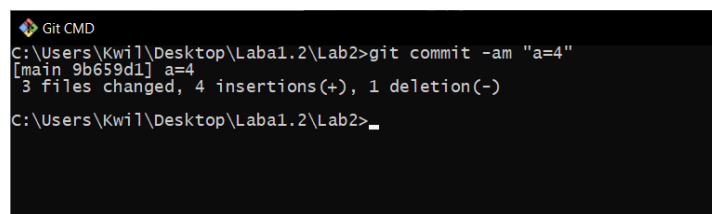
```
Git CMD
C:\Users\Kwi1\Desktop\Laba1.2\Lab2>git commit -am "int a,c,b"
[main 8f12dce] int a,c,b
1 file changed, 0 insertions(+), 0 deletions(-)
C:\Users\Kwi1\Desktop\Laba1.2\Lab2>_
```

Рисунок 5.2 Коммит изменений



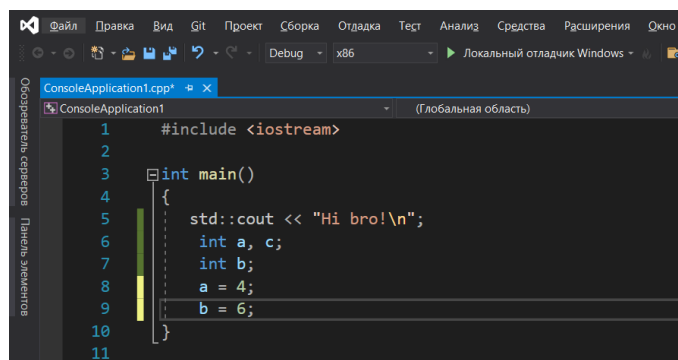
```
ConsoleApplication1.cpp
1 #include <iostream>
2
3 int main()
4 {
5     std::cout << "Hi bro!\n";
6     int a, c;
7     int b;
8     a = 4;
9 }
10
11
12
```

Рисунок 5.3 Изменения в программе



```
Git CMD
C:\Users\Kwi1\Desktop\Laba1.2\Lab2>git commit -am "a=4"
[main 9b659d1] a=4
3 files changed, 4 insertions(+), 1 deletion(-)
C:\Users\Kwi1\Desktop\Laba1.2\Lab2>_
```

Рисунок 5.4 Коммит изменений

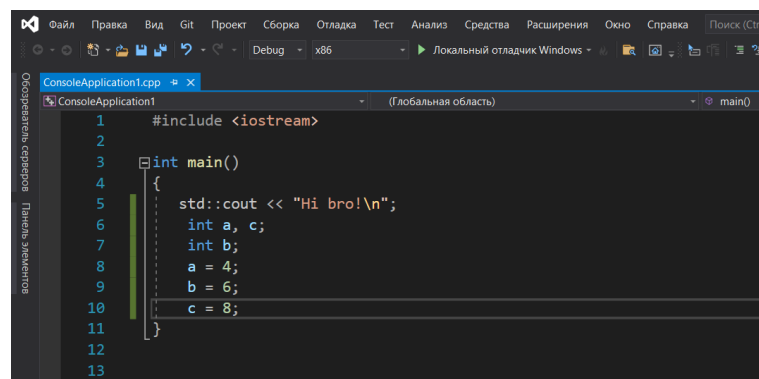


```
ConsoleApplication1.cpp
1 #include <iostream>
2
3 int main()
4 {
5     std::cout << "Hi bro!\n";
6     int a, c;
7     int b;
8     a = 4;
9     b = 6;
10 }
11
```

Рисунок 5.5 Изменения в программе

```
Git CMD
C:\Users\Kwi1\Desktop\Laba1.2\Lab2>git commit -am "b=4"
[main 4c6f913] b=4
3 files changed, 1 insertion(+)
C:\Users\Kwi1\Desktop\Laba1.2\Lab2>
```

Рисунок 5.6 Коммит изменений

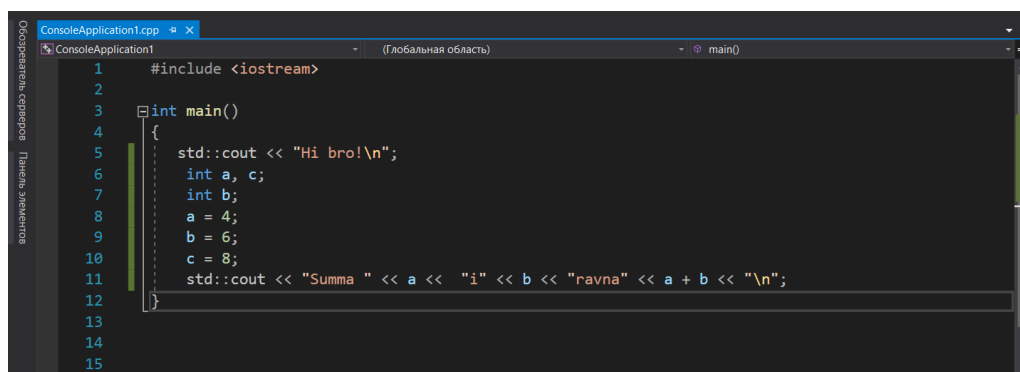


```
File Edit View Git Project Build Debug Test Analysis Tools Extensions Window Help (Ctrl+...)
Debug x86 Локальный отладчик Windows
ConsoleApplication1.cpp
ConsoleApplication1 (Глобальная область) main()
1 #include <iostream>
2
3 int main()
4 {
5     std::cout << "Hi bro!\n";
6     int a, c;
7     int b;
8     a = 4;
9     b = 6;
10    c = 8;
11 }
12
13
```

Рисунок 5.7 Изменения в программе

```
Git CMD
3 files changed, 1 insertion(+)
C:\Users\Kwi1\Desktop\Laba1.2\Lab2>git commit -am "c=8"
[main 9ffa1cc] c=8
3 files changed, 1 insertion(+)
C:\Users\Kwi1\Desktop\Laba1.2\Lab2>
```

Рисунок 5.8 Коммит изменений



```
ConsoleApplication1.cpp
ConsoleApplication1 (Глобальная область) main()
1 #include <iostream>
2
3 int main()
4 {
5     std::cout << "Hi bro!\n";
6     int a, c;
7     int b;
8     a = 4;
9     b = 6;
10    c = 8;
11    std::cout << "Summa " << a << " + " << b << " = " << a + b << "\n";
12 }
13
14
15
```

Рисунок 5.9 Изменения в программе

```
Git CMD
C:\Users\Kwi1\Desktop\Lab1.2\Lab2>git commit -am "summa"
[main b7cf953] summa
3 files changed, 1 insertion(+)
rewrite program/ConsoleApplication1/.vs/ConsoleApplication1/v16/.suo (68%)
C:\Users\Kwi1\Desktop\Lab1.2\Lab2>_
```

Рисунок 5.10 Коммит изменений

```
Git CMD
rewrite program/ConsoleApplication1/.vs/ConsoleApplication1/v16/.suo (68%)
C:\Users\Kwi1\Desktop\Lab1.2\Lab2>git push
Enumerating objects: 60, done.
Counting objects: 100% (60/60), done.
Delta compression using up to 12 threads
Compressing objects: 100% (37/37), done.
Writing objects: 100% (52/52), 9.90 KiB | 1.10 MiB/s, done.
Total 52 (delta 20), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (20/20), completed with 3 local objects.
To https://github.com/Artem0622/Lab2.git
11cac20..b7cf953 main -> main
C:\Users\Kwi1\Desktop\Lab1.2\Lab2>_
```

Рисунок 5.11 Пуш изменений

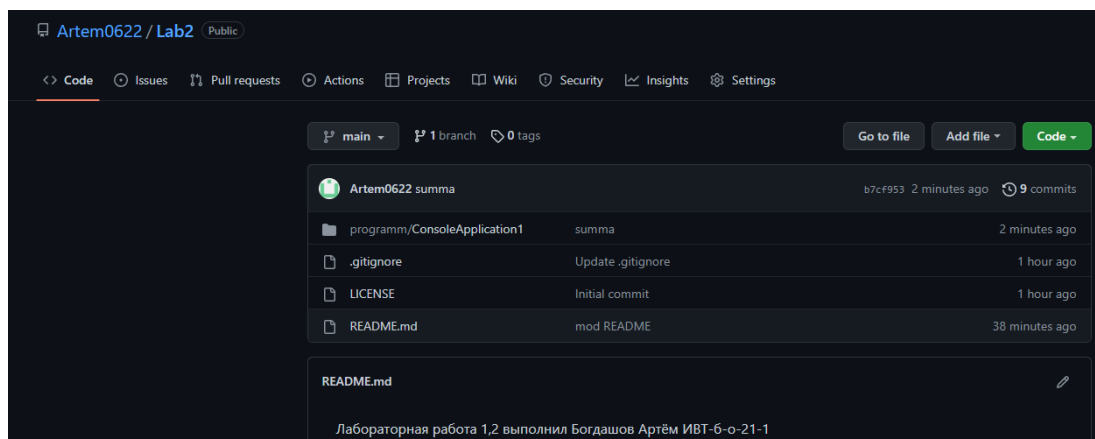


Рисунок 5.12 Изменения на удалённом сервере

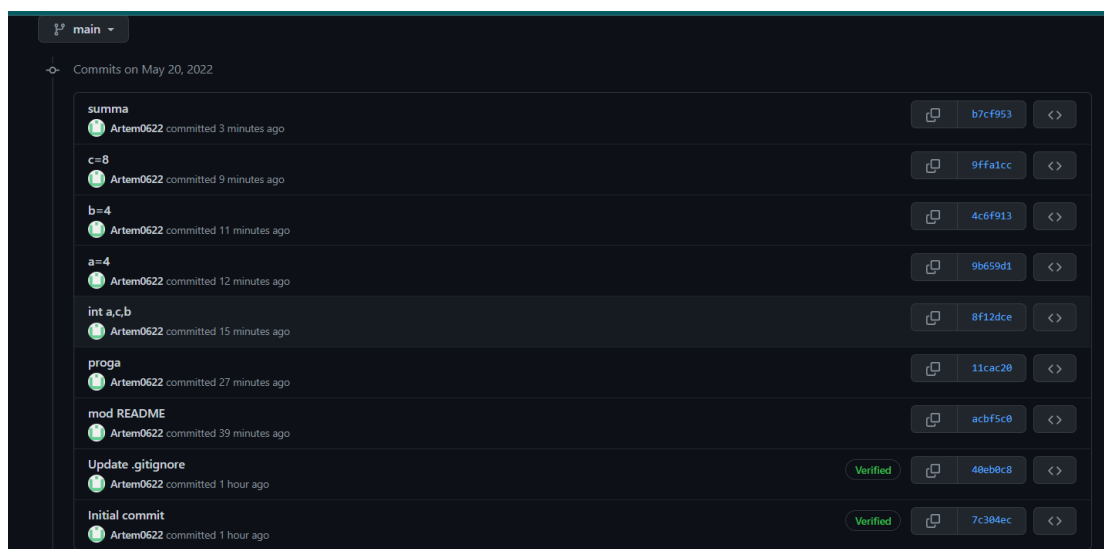




Рисунок 5.13 История коммитов на удалённом сервере

```
Git CMD
C:\Users\Kwi1\Desktop\Laba1.2\Lab2>git tag -a v1.0 "summa value for b"
fatal: Failed to resolve 'summa value for b' as a valid ref.
C:\Users\Kwi1\Desktop\Laba1.2\Lab2>git tag -a v1.0 -m "summa"
C:\Users\Kwi1\Desktop\Laba1.2\Lab2>git tag
v1.0
C:\Users\Kwi1\Desktop\Laba1.2\Lab2>_
```

Рисунок 5.14 Присваивание тега коммиту

```
Git CMD
v1.0
C:\Users\Kwi1\Desktop\Laba1.2\Lab2>git tag -a v1.1 -m "summa"
C:\Users\Kwi1\Desktop\Laba1.2\Lab2>git tag
v1.0
v1.1
C:\Users\Kwi1\Desktop\Laba1.2\Lab2>
```

Рисунок 5.15 Присваивание тега коммиту

```
Git CMD
C:\Users\Kwi1\Desktop\Laba1.2\Lab2>git push origin --tags
Enumerating objects: 2, done.
Counting objects: 100% (2/2), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 198 bytes | 198.00 KiB/s, done.
Total 2 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/Artem0622/Lab2.git
 * [new tag]         v1.0 -> v1.0
 * [new tag]         v1.1 -> v1.1
C:\Users\Kwi1\Desktop\Laba1.2\Lab2>
```

Рисунок 5.16 Push тегов

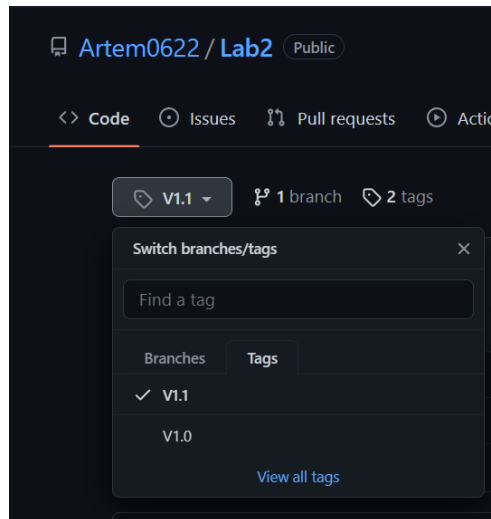


Рисунок 5.17 История тегов на удалённом сервере

6.Просмотрел историю хранилища командой git log:

```

Git CMD
fatal: invalid --pretty format: online

C:\Users\Kwil\Desktop\Lab1.2\Lab2>git log --graph --pretty=oneline --abbrev-commit
* b7cf953 (HEAD -> main, tag: V1.1, tag: V1.0, origin/main, origin/HEAD) summa
* 9ffa1cc c=8
* 4c6f913 b=4
* 9b659d1 a=4
* 8f12dce int a,c,b
* 11cac20 proga
* acbf5c0 mod README
* 40eb0c8 Update .gitignore
* 7c304ec Initial commit

C:\Users\Kwil\Desktop\Lab1.2\Lab2>

```

Рисунок 6. История коммитов

7.Просмотрел содержимое коммитов командой git show HEAD, gitshow HEAD~, git show cb59ad1:

```

C:\Users\Kwil\Desktop\Lab1.2\Lab2>git show HEAD
commit b7cf953bf81754fd1558b3213b33716ebc13cbc5 (HEAD -> main, tag: V1.1, tag: V1.0, origin/main, origin/HEAD)
Author: Artem0622 <kwilmajor707@gmail.com>
Date: Fri May 20 03:14:57 2022 +0300

    summa

diff --git a/programm/ConsoleApplication1/.vs/ConsoleApplication1/v16/.suo b/programm/ConsoleApplication1/.vs/ConsoleApplication1/v16/.suo
index d6c4411..536f9c2 100644
Binary files a/programm/ConsoleApplication1/.vs/ConsoleApplication1/v16/.suo and b/programm/ConsoleApplication1/.vs/ConsoleApplication1/v16/.suo differ
diff --git a/programm/ConsoleApplication1/.vs/ConsoleApplication1/v16/Browse.VC.db b/programm/ConsoleApplication1/.vs/ConsoleApplication1/v16/Browse.VC.db
index 08cbb28..3444792 100644
Binary files a/programm/ConsoleApplication1/.vs/ConsoleApplication1/v16/Browse.VC.db and b/programm/ConsoleApplication1/.vs/ConsoleApplication1/v16/Browse.VC.db differ
diff --git a/programm/ConsoleApplication1/ConsoleApplication1.cpp b/programm/ConsoleApplication1/ConsoleApplication1.cpp
index 12daa3e..b077634 100644
--- a/programm/ConsoleApplication1/ConsoleApplication1.cpp
+++ b/programm/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp
@@ -8,6 +8,7 @@ int main()
 {
     a = 4;
     b = 6;
     c = 8;
     std::cout << "Summa " << a << " + " << b << " = " << a + b << "\n";
 }

```

Рисунок 7.1 Содержимое коммитов командами

```
Git CMD - "C:\Program Files\Git\cmd\git.exe" show HEAD~
C:\Users\Kwil\Desktop\Lab1.2\Lab2>git show HEAD~
commit 9ffa1cc2c0c62dcbf86ef0863111d6f9d16af130
Author: Artem0622 <kwilmajor707@gmail.com>
Date:   Fri May 20 03:08:57 2022 +0300

    c=8

diff --git a/programm/ConsoleApplication1/.vs/ConsoleApplication1/v16/.suo b/programm/ConsoleApplication1/.vs/ConsoleApplication1/v16/.suo
index 4e86b14..d6c4411 100644
Binary files a/programm/ConsoleApplication1/.vs/ConsoleApplication1/v16/.suo and b/programm/ConsoleApplication1/.vs/ConsoleApplication1/v16/.suo differ
diff --git a/programm/ConsoleApplication1/.vs/ConsoleApplication1/v16/Browse.VC.db b/programm/ConsoleApplication1/.vs/ConsoleApplication1/v16/Browse.VC.db
index 6220dc2..08cbb28 100644
Binary files a/programm/ConsoleApplication1/.vs/ConsoleApplication1/v16/Browse.VC.db and b/programm/ConsoleApplication1/.vs/ConsoleApplication1/v16/Browse.VC.db differ
diff --git a/programm/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp b/programm/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp
index bade85a..12daa3e 100644
--- a/programm/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp
+++ b/programm/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp
@@ -7,6 +7,7 @@ int main()
     int b;
     a = 4;
     b = 6;
+    c = 8;
 }
```

Рисунок 7.2 Содержание коммитов командами

8. Удалил весь код в файле ConsoleApplication1.cpp и сохранил его, затем удалил все несохраненные изменения командой, после этого еще раз удалил весь код в файле и сделал коммит, после чего откатил состояние файла к предыдущей версии.

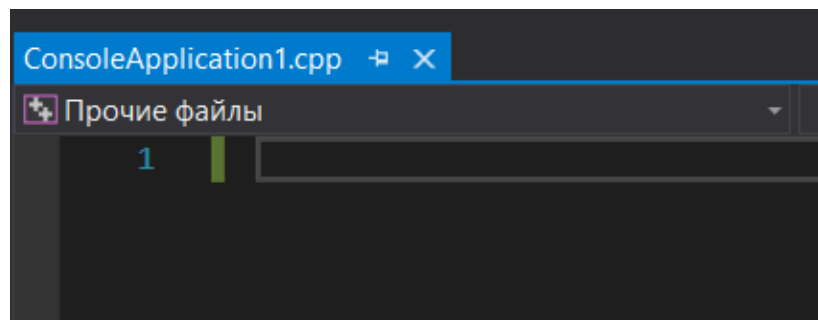
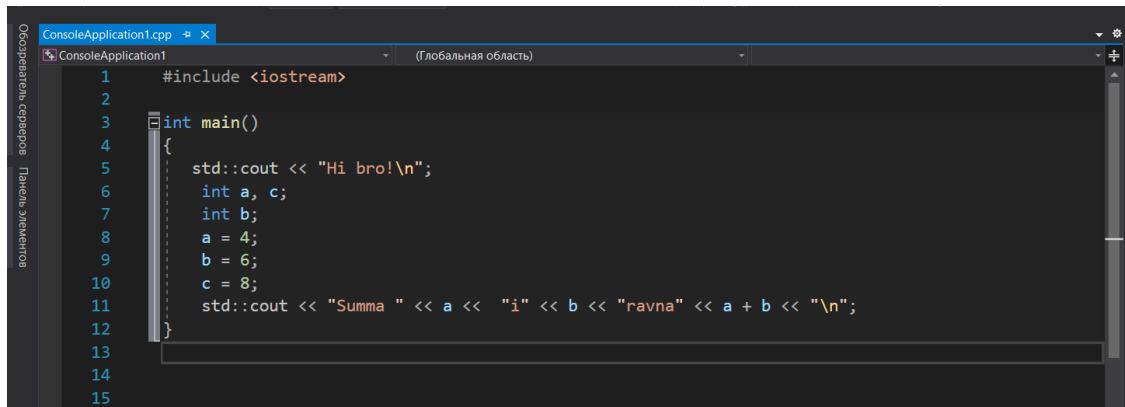


Рисунок 8.1 Удаление кода в файле ConsoleApplication.cpp

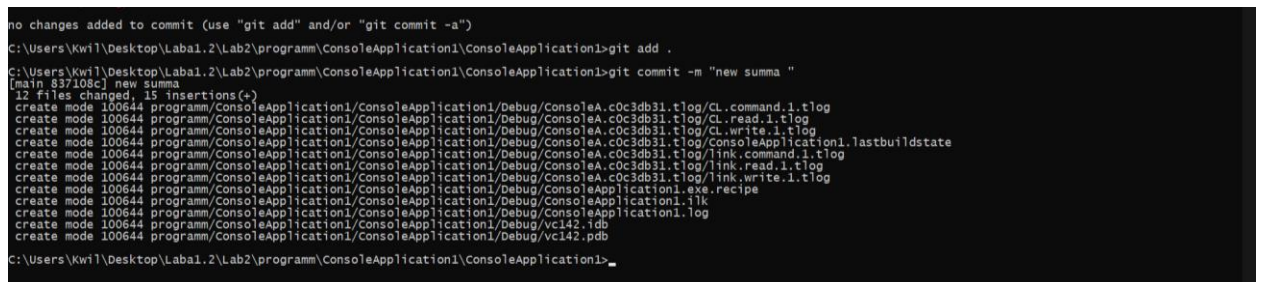
```
C:\Users\Kwil\Desktop\Lab1.2\Lab2>cd/d C:\Users\Kwil\Desktop\Lab1.2\Lab2\programm\ConsoleApplication1
C:\Users\Kwil\Desktop\Lab1.2\Lab2\programm\ConsoleApplication1>git checkout -- ConsoleApplication1.cpp
error: pathspec 'ConsoleApplication1.cpp' did not match any file(s) known to git
C:\Users\Kwil\Desktop\Lab1.2\Lab2\programm\ConsoleApplication1>git checkout -- ConsoleApplication1.cpp
error: pathspec 'ConsoleApplication1.cpp' did not match any file(s) known to git
C:\Users\Kwil\Desktop\Lab1.2\Lab2\programm\ConsoleApplication1>cd/d C:\Users\Kwil\Desktop\Lab1.2\Lab2\programm\ConsoleApplication1\ConsoleApplication1
C:\Users\Kwil\Desktop\Lab1.2\Lab2\programm\ConsoleApplication1\ConsoleApplication1>git checkout -- ConsoleApplication1.cpp
```

Рисунок 8.2 checkout изменений файла ConsoleApplication.cpp



```
1 #include <iostream>
2
3 int main()
4 {
5     std::cout << "Hi bro!\n";
6     int a, c;
7     int b;
8     a = 4;
9     b = 6;
10    c = 8;
11    std::cout << "Summa " << a << "i" << b << "ravna" << a + b << "\n";
12 }
13
14
15
```

Рисунок 8.3 Изменения в файле с программой после команды



```
no changes added to commit (use "git add" and/or "git commit -a")
C:\Users\Kw1\Desktop\Lab1.2\Lab2\programm\ConsoleApplication1\ConsoleApplication1>git add .
C:\Users\Kw1\Desktop\Lab1.2\Lab2\programm\ConsoleApplication1\ConsoleApplication1>git commit -m "new summa "
[main 837108c] new summa
12 files changed, 15 insertions(+)
create mode 100644 programm\ConsoleApplication1\ConsoleApplication1/Debug/ConsoleA.c0c3db31.tlog/CL.command.1.tlog
create mode 100644 programm\ConsoleApplication1\ConsoleApplication1/Debug/ConsoleA.c0c3db31.tlog/CL.read.1.tlog
create mode 100644 programm\ConsoleApplication1\ConsoleApplication1/Debug/ConsoleA.c0c3db31.tlog/CL.write.1.tlog
create mode 100644 programm\ConsoleApplication1\ConsoleApplication1/Debug/ConsoleA.c0c3db31.tlog/ConsoleApplication1.lastbuildstate
create mode 100644 programm\ConsoleApplication1\ConsoleApplication1/Debug/ConsoleA.c0c3db31.tlog/link.command.1.tlog
create mode 100644 programm\ConsoleApplication1\ConsoleApplication1/Debug/ConsoleA.c0c3db31.tlog/link.read.1.tlog
create mode 100644 programm\ConsoleApplication1\ConsoleApplication1/Debug/ConsoleA.c0c3db31.tlog/link.write.1.tlog
create mode 100644 programm\ConsoleApplication1\ConsoleApplication1/Debug/ConsoleApplication1.exe.recipe
create mode 100644 programm\ConsoleApplication1\ConsoleApplication1/Debug/ConsoleApplication1.ilk
create mode 100644 programm\ConsoleApplication1\ConsoleApplication1/Debug/ConsoleApplication1.log
create mode 100644 programm\ConsoleApplication1\ConsoleApplication1/Debug/vc142.idb
create mode 100644 programm\ConsoleApplication1\ConsoleApplication1/Debug/vc142.pdb
C:\Users\Kw1\Desktop\Lab1.2\Lab2\programm\ConsoleApplication1\ConsoleApplication1>
```

Рисунок 8.4 Коммит изменений

**Вывод:** команда `git -checkout <FileName>` удаляет изменения произошед

**Контрольные вопросы и ответы на них:**

Вопросы для защиты работы.

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

Наиболее простой и в то же время мощный инструмент для этого — команда `git log`. По умолчанию, без аргументов, `git log` выводит список коммитов созданных в данном репозитории в обратном хронологическом порядке. То есть самые последние коммиты показываются первыми.

Одна из опций, когда вы хотите увидеть сокращенную статистику для каждого коммита, вы можете использовать опцию `—stat`.

Вторая опция (одна из самых полезных аргументов) является `-p` или `--patch`, который показывает разницу (выводит патч), внесенную в каждый коммит. Так же вы можете ограничить количество записей в выводе команды; используйте параметр `-2` для вывода только двух записей (пример команды `git log -p -2`).

Третья действительно полезная опция это `--pretty`. Она 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. Особенно это может быть полезным, когда вы хотите сгенерировать вывод для автоматического анализа — так как вы указываете формат явно, он не будет изменен даже после обновления Git.

Для опции `git log --pretty=format` существуют различного рода опции для изменения формата отображения.

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

Для ограничения может использоваться функция `git log <n>`, где `n` число записей.

Также, существуют опции для ограничения вывода по времени, такие как `--since` и `--until`, они являются очень удобными. Например, следующая команда покажет список коммитов, сделанных за последние две недели:

```
git log --since=2.weeks
```

Это команда работает с большим количеством форматов — вы можете указать определенную дату вида 2008-01-15 или же относительную дату, например 2 years 1 day 3 minutes ago.

Также вы можете фильтровать список коммитов по заданным параметрам. Опция --author дает возможность фильтровать по автору коммита, а опция --grep (показывает только коммиты, сообщение которых содержит указанную строку) искать по ключевым словам в сообщении коммита. Функция -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?

Внести изменения можно с помощью команды `git commit --amend`

Эта команда берёт индекс и применяет его к последнему коммиту. Если после последнего коммита не было никаких проиндексированных изменений (например, вы запустили приведённую команду сразу после предыдущего коммита), то состояние проекта будет абсолютно таким же и всё, что мы изменим, это комментарий к коммиту.

Для того, чтобы внести необходимые изменения - нам нужно проиндексировать их и выполнить команду `git commit --amend`.

```
git commit -m 'initial commit'
```

```
git add forgotten_file
```

```
git commit --amend
```

Эффект от выполнения этой команды такой, как будто мы не выполнили предыдущий коммит, а еще раз выполнили команду `git add` и выполнили коммит.

### 4. Как отменить индексацию файла в Git?

Например, вы изменили два файла и хотите добавить их в разные коммиты, но случайно выполнили команду `git add *` и добавили в индекс оба. Как исключить из индекса один из них? Команда `git status` напомним вам:

Прямо под текстом «Changes to be committed» говорится: используйте `git reset HEAD <file>` для исключения из индекса.

5. Как отменить изменения в файле?

С помощью команды `git checkout -- <file>`.

6. Что такое удаленный репозиторий Git?

Удалённый репозиторий это своего рода наше облако, в которое мы сохраняем те или иные изменения в нашей программе/коде/файлах.

7. Как выполнить просмотр удаленных репозитория данного локального репозитория?

Для того, чтобы просмотреть список настроенных удалённых репозитория, необходимо запустить команду `git remote`.

Также можно указать ключ `-v`, чтобы просмотреть адреса для чтения и записи, привязанные к репозиторию. Пример: `git remote -v`

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?

Если необходимо получить изменения, которые есть у Пола, но нету у вас, вы можете выполнить команду `git fetch <Название репозитория>`. Важно отметить, что команда `git fetch` забирает данные в ваш локальный репозиторий, но не сливает их с какими-либо вашими наработками и не модифицирует то, над чем вы работаете в данный момент. Вам необходимо вручную слить эти данные с вашими, когда вы будете готовы.

Если ветка настроена на отслеживание удалённой ветки, то вы можете использовать команду `git pull` чтобы автоматически получить изменения из удалённой ветки и слить их со своей текущей. Выполнение `git pull`, как

правило, извлекает (fetch) данные с сервера, с которого вы изначально клонировали, и автоматически пытается слить (merge) их с кодом, над которым вы в данный момент работаете.

Чтобы отправить изменения на удалённый репозиторий необходимо отправить их в удалённый репозиторий. Команда для этого действия про-стая: `git push <remote-name> <branch-name>`.

#### 10. Как выполнить просмотр удаленного репозитория?

Для просмотра удалённого репозитория, можно использовать ко-манду `git remote show <remote>`.

#### 11. Каково назначение тэгов Git?

Теги - это ссылки указывающие на определённые версии ко-да/написанной программы. Они удобно чтобы в случае чего вернуться к нужному моменту. Также при помощи тегов можно пометать важные мо-менты.

#### 12. Как осуществляется работа с тэгами Git?

Просмотреть наличие тегов можно с помощью команды: `git tag`.

А назначить (указать, добавить тег) можно с помощью команды `git tag -a v1.4(версия изначальная) -m "Название"`.

С помощью команды `git show` вы можете посмотреть данные тега вместе с коммитом: `git show v1.4`.

Отправка тегов, по умолчанию, команда `git push` не отправляет теги на удалённые сервера. После создания теги нужно отправлять явно на уда-лённый сервер. Процесс аналогичен отправке веток — достато-чно выпол-нить команду `git push origin <tagname>`. Для отправки всех тегов можно использовать команду `git push origin tags`.

Для удаления тега в локальном репозитории достаточно выполнить команду `git tag -d <tagname>`. Например, удалить созданный ранее легко-весный тег можно следующим образом: `git tag -d v1.4-lw`



Для удаления тега из внешнего репозитория используется команда  
`git`

`push origin --delete <tagname>`.

Если вы хотите получить версии файлов, на которые указывает тег, то  
вы можете сделать `git checkout` для тега пример: `git checkout -b version2`  
`v2.0.0`.

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.

шие с файлом в репозитории до коммита.

