

**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

Уланбекова Айканыш Уланбековна

Ставрополь 2022

Выполнение работы:

1. Создал общедоступный репозиторий `rep_1.2` на GitHub в котором будет использована лицензия MIT и выбранный мной язык программирования.

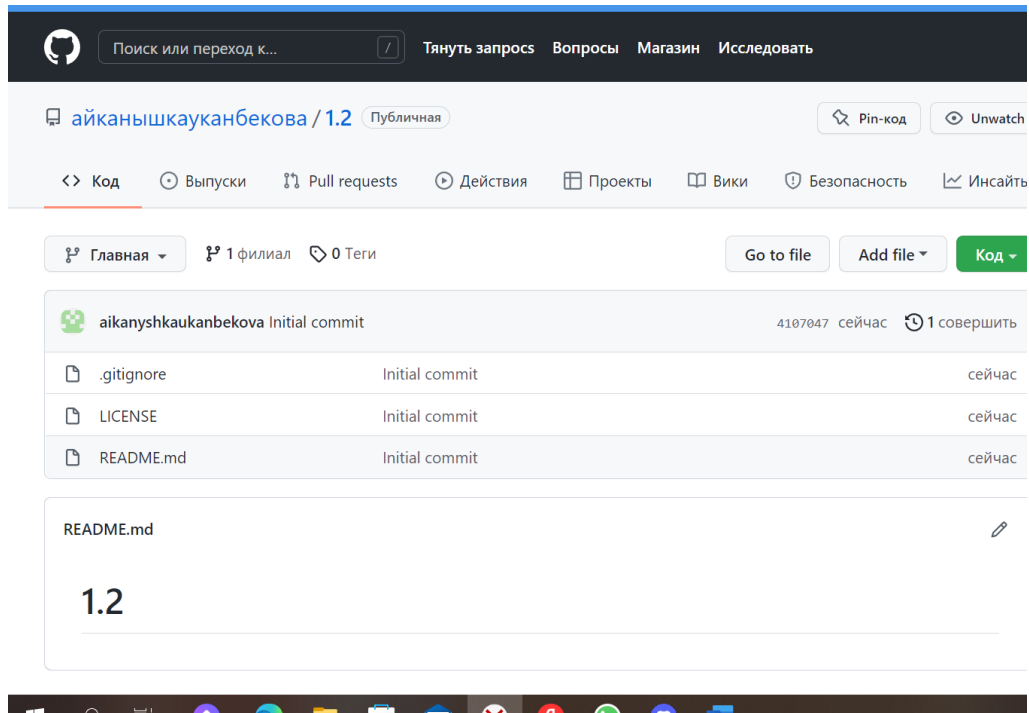


Рисунок 1.1 Созданный репозиторий в GitHub

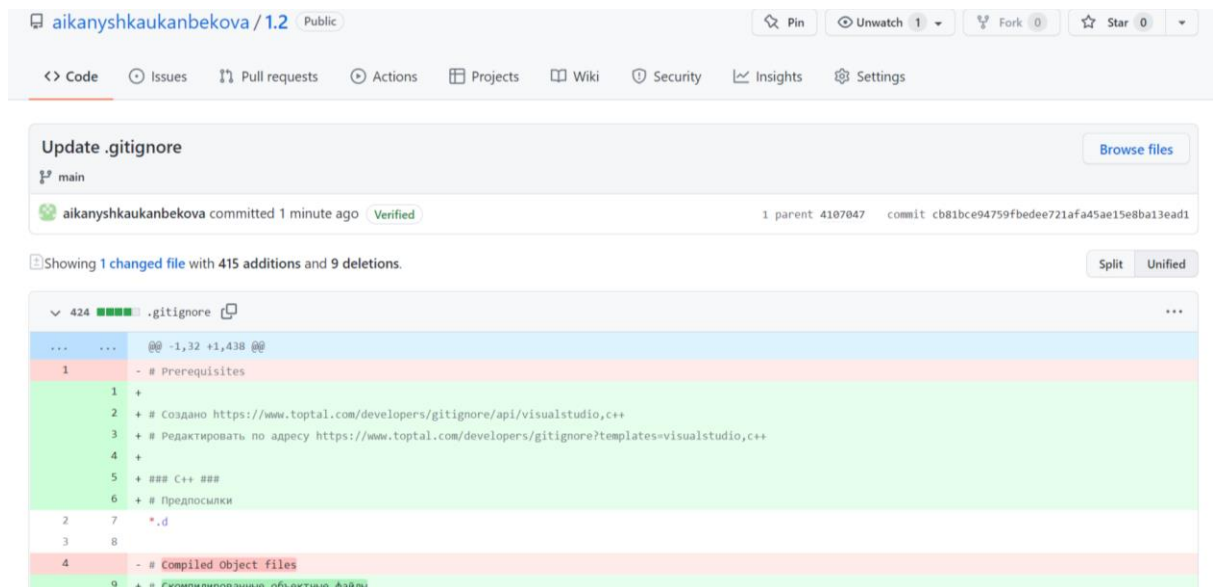
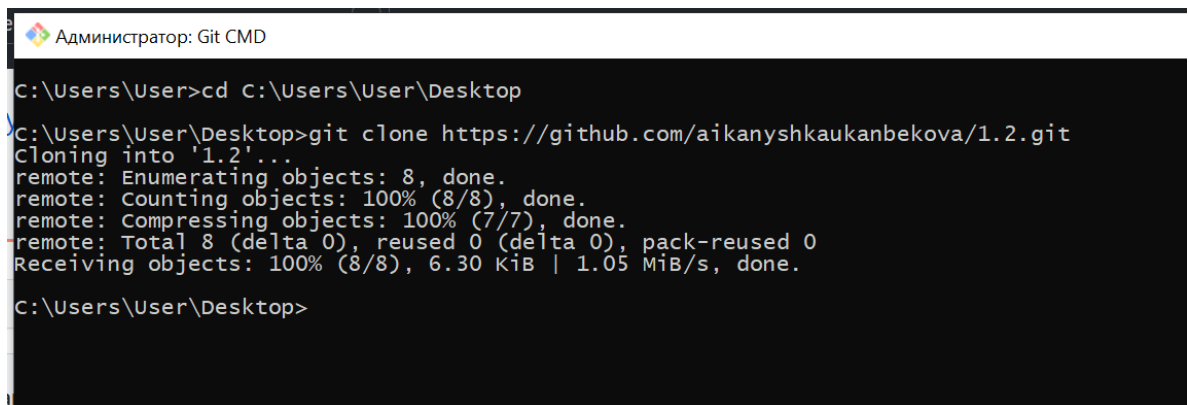


Рисунок 1.2 Изменения в файле .gitignore

2. Клонировал созданный репозиторий на раб. компьютер:



```
Администратор: Git CMD

C:\Users\User>cd C:\Users\User\Desktop

C:\Users\User\Desktop>git clone https://github.com/aikanyshkaukanbekova/1.2.git
Cloning into '1.2'...
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 8 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (8/8), 6.30 KiB | 1.05 MiB/s, done.

C:\Users\User\Desktop>
```

Рисунок 2. Клонирование репозитория

3. Добавила информацию в README и закоммитила:

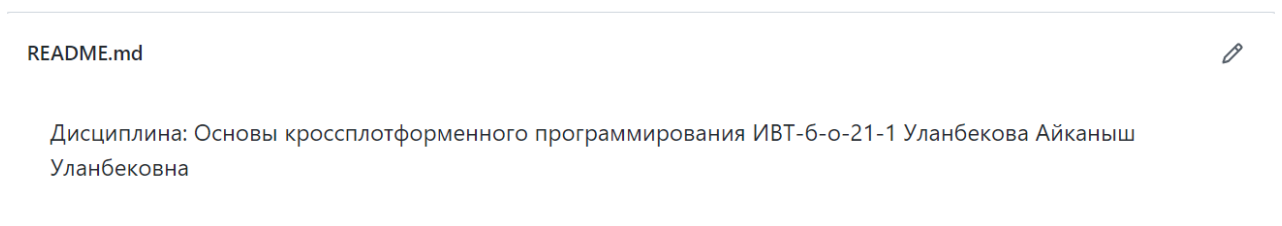
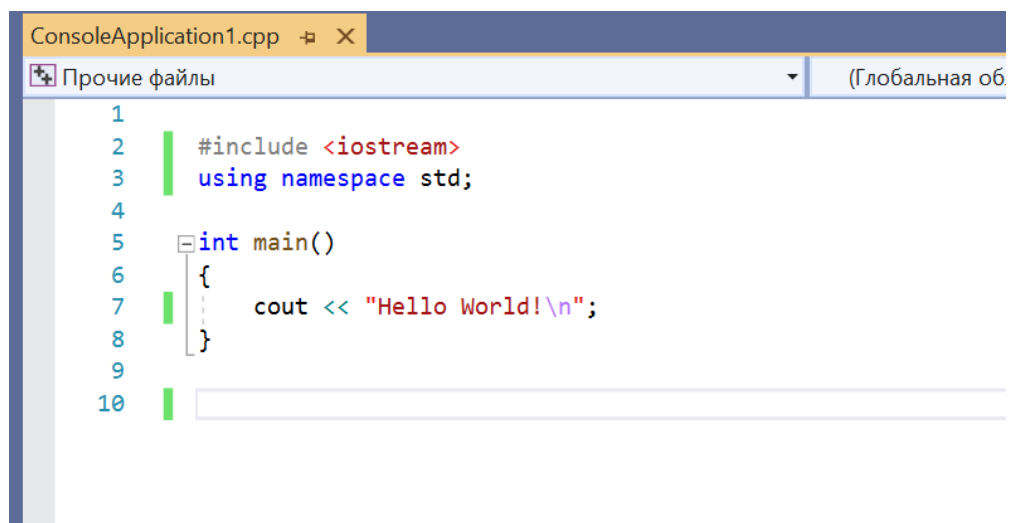


Рисунок 3.1 Добавление информации в README.md

4. Написал в репозитории небольшую программу, сделал коммит и пуш:



```
ConsoleApplication1.cpp

1
2 #include <iostream>
3 using namespace std;
4
5 int main()
6 {
7     cout << "Hello World!\n";
8 }
9
10
```

Рисунок 4.1 Изменения в программе

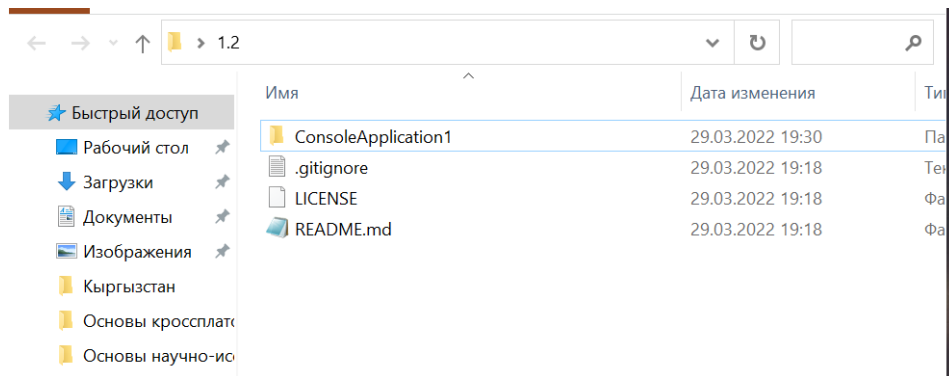


Рисунок 4.2 Добавлена папка с проектом на C++

```
C:\Users\User\Desktop\1.2>git add .
C:\Users\User\Desktop\1.2>git commit -m "red"
[main c324dc8] red
4 files changed, 209 insertions(+)
create mode 100644 ConsoleApplication1/ConsoleApplication1.sln
create mode 100644 ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp
create mode 100644 ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.vcxproj
create mode 100644 ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.vcxproj.filters

C:\Users\User\Desktop\1.2>git push
Enumerating objects: 12, done.
Counting objects: 100% (12/12), done.
Delta compression using up to 8 threads
Compressing objects: 100% (10/10), done.
Writing objects: 100% (10/10), 2.95 KiB | 504.00 KiB/s, done.
Total 10 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/aikanyshkauanbekova/1.2.git
 3914066..ef019cf main -> main
C:\Users\User\Desktop\1.2>
```

Рисунок 4.3 Коммит и пуш программы на уд. сервер

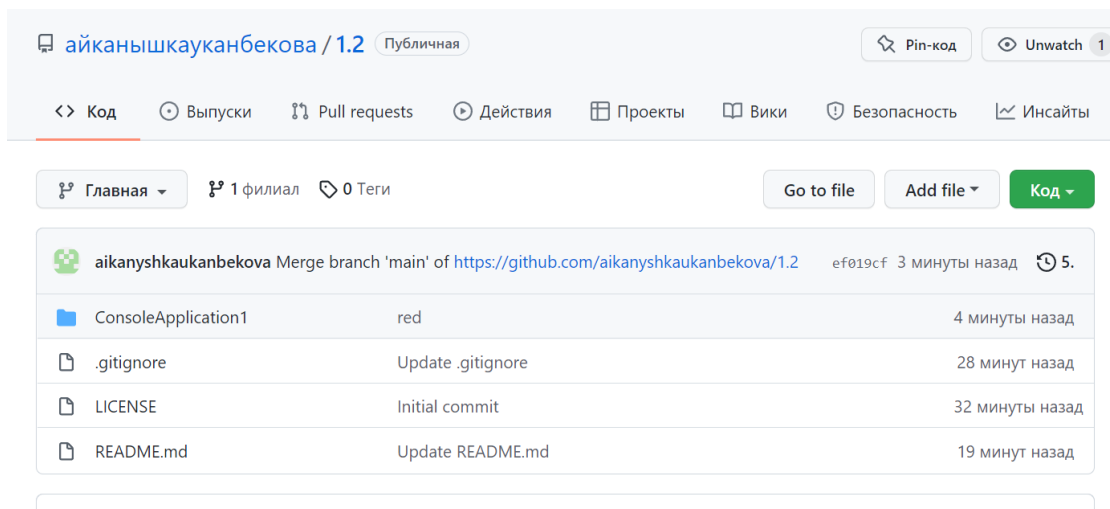


Рисунок 4.4 Изменения на уд. сервере

5. Делал коммиты в процессе изменения программы, отметил их тегами и запустил на уд. сервер коммиты затем теги:

```
1
2  #include <iostream>
3  using namespace std;
4
5  int main()
6  {
7      cout << "Hello World!\n";
8      int a, b;
9  }
```

Рисунок 5.1 Изменения в программе

```
C:\Users\User\Desktop\1.2>git add .
C:\Users\User\Desktop\1.2>git commit -m "a"
[main 317f768] a
1 file changed, 1 insertion(+)
C:\Users\User\Desktop\1.2>
```

Рисунок 5.2 Коммит изменений

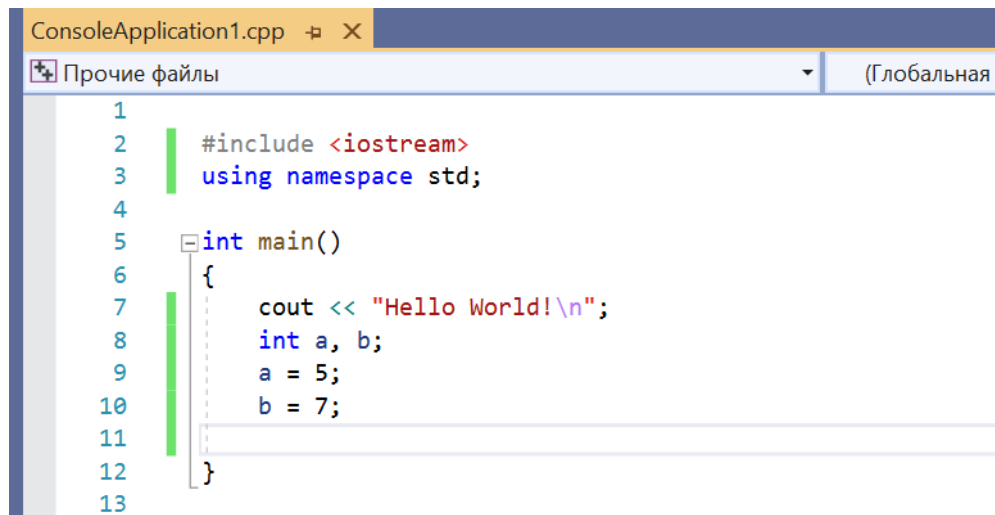
```
1
2  #include <iostream>
3  using namespace std;
4
5  int main()
6  {
7      cout << "Hello World!\n";
8      int a, b;
9      a = 5;
10
11 }
```

Рисунок 5.3 Изменения в программе

```
C:\Users\User\Desktop\1.2>git add .
C:\Users\User\Desktop\1.2>git commit -m "b"
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
C:\Users\User\Desktop\1.2>
```

Рисунок 5.4 Коммит изменений



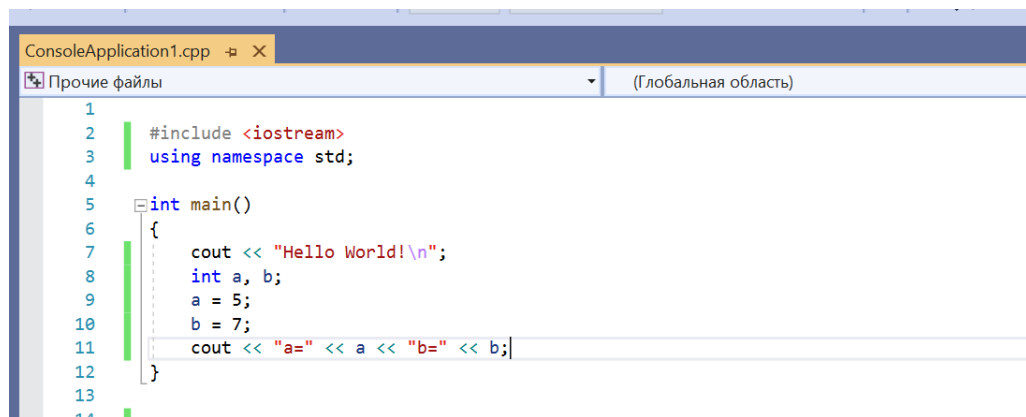
```
ConsoleApplication1.cpp  X
Прочие файлы (Глобальная)
1
2 #include <iostream>
3 using namespace std;
4
5 int main()
6 {
7     cout << "Hello World!\n";
8     int a, b;
9     a = 5;
10    b = 7;
11
12 }
13
```

Рисунок 5.5 Изменения в программе



```
C:\Users\User\Desktop\1.2>git add .
C:\Users\User\Desktop\1.2>git commit -m "c"
[main 2f37fb1] c
1 file changed, 3 insertions(+)
C:\Users\User\Desktop\1.2>
```

Рисунок 5.6 Коммит изменений



```
ConsoleApplication1.cpp  X
Прочие файлы (Глобальная область)
1
2 #include <iostream>
3 using namespace std;
4
5 int main()
6 {
7     cout << "Hello World!\n";
8     int a, b;
9     a = 5;
10    b = 7;
11    cout << "a=" << a << "b=" << b;
12 }
13
14
```

Рисунок 5.7 Изменения в программе



```
C:\Users\User\Desktop\1.2>git add .
C:\Users\User\Desktop\1.2>git commit -m "d"
[main 12c3d18] d
1 file changed, 1 insertion(+), 1 deletion(-)
C:\Users\User\Desktop\1.2>
```

Рисунок 5.8 Коммит изменений

```

1
2 #include <iostream>
3 using namespace std;
4
5 int main()
6 {
7     cout << "Hello World!\n";
8     int a, b;
9     a = 5;
10    b = 7;
11    cout << "a=" << a << "b=" << b;
12    if (a > b)
13    {
14        cout << "\n a>b";
15    }
16 }
17

```

Рисунок 5.9 Изменения в программе

```

C:\Users\User\Desktop\1.2>git add .
C:\Users\User\Desktop\1.2>git commit -m "i"
[main 700508d] i
1 file changed, 4 insertions(+)
C:\Users\User\Desktop\1.2>_

```

Рисунок 5.10 Коммит изменений

Прочие файлы (Глобальная)

```

1
2 #include <iostream>
3 using namespace std;
4
5 int main()
6 {
7     cout << "Hello World!\n";
8     int a, b;
9     a = 5;
10    b = 7;
11    cout << "a=" << a << "b=" << b;
12    if (a > b)
13    {
14        cout << "\n a<b";
15    }
16    else
17    {
18        cout << "\n a>b";
19    }
20 }
21

```

Рисунок 5.9 Изменения в программе

```

C:\Users\User\Desktop\1.2>git add .
C:\Users\User\Desktop\1.2>git commit -m "f"
[main ba72cfa] f
1 file changed, 5 insertions(+), 1 deletion(-)
C:\Users\User\Desktop\1.2>_

```

Рисунок 5.11 Коммит изменений

```

C:\Users\User\Desktop\1.2>git push
Enumerating objects: 28, done.
Counting objects: 100% (28/28), done.
Delta compression using up to 8 threads
Compressing objects: 100% (25/25), done.
Writing objects: 100% (25/25), 1.92 KiB | 491.00 KiB/s, done.
Total 25 (delta 18), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (18/18), completed with 2 local objects.
To https://github.com/aikanyshkauanbekova/1.2.git
   ef019cf..ba72cfa  main -> main
C:\Users\User\Desktop\1.2>

```

Рисунок 5.12 Пуш изменений

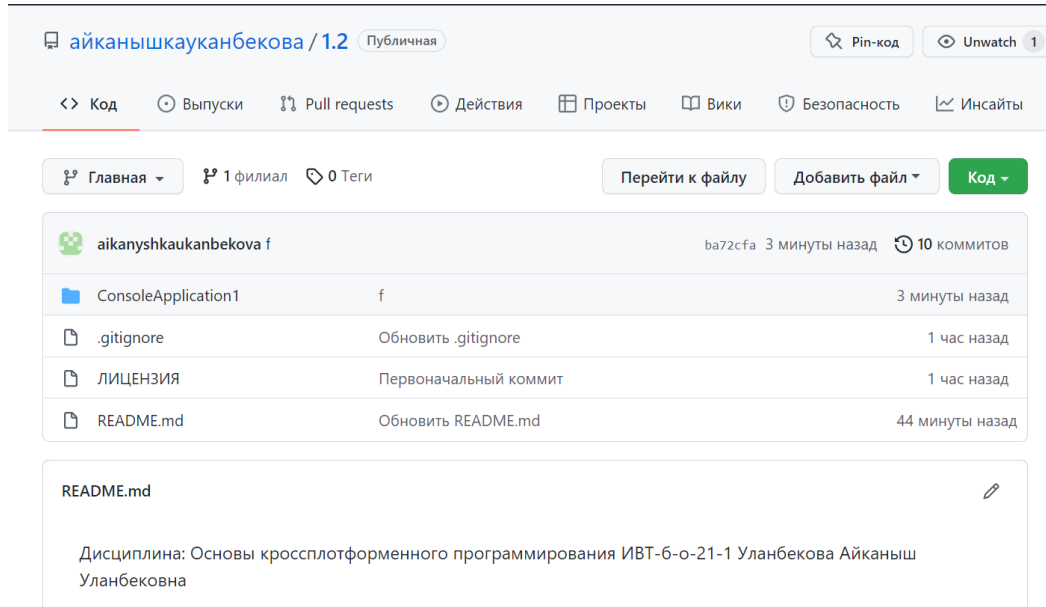


Рисунок 5.13 Изменения на уд. сервере

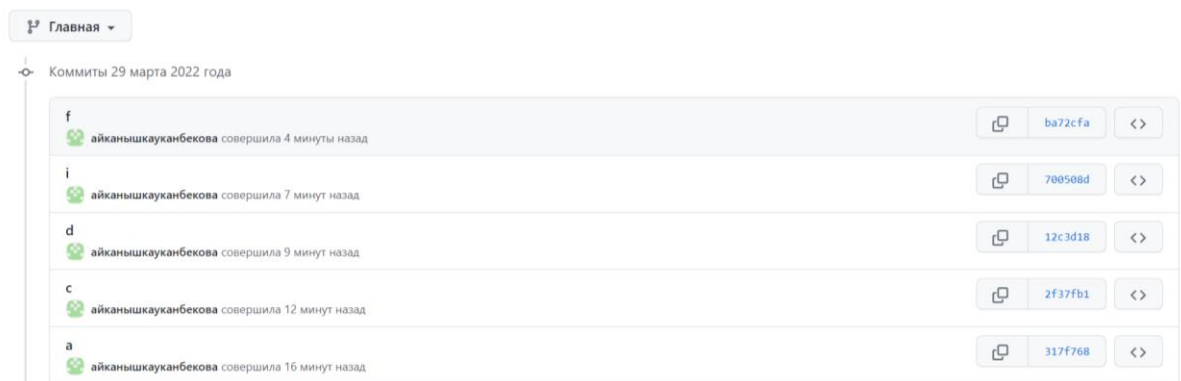


Рисунок 5.14 История коммитов на уд. сервере

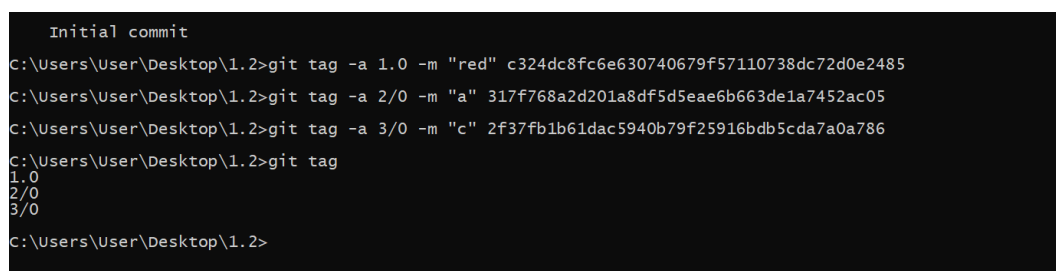


Рисунок 5.15 Присваивание тега коммиту

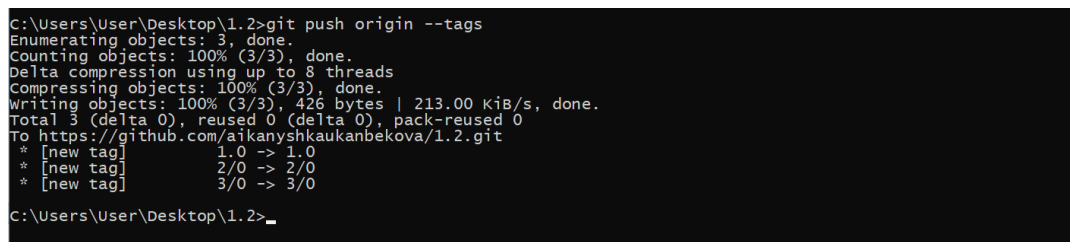


Рисунок 5.16 Пуш тегов

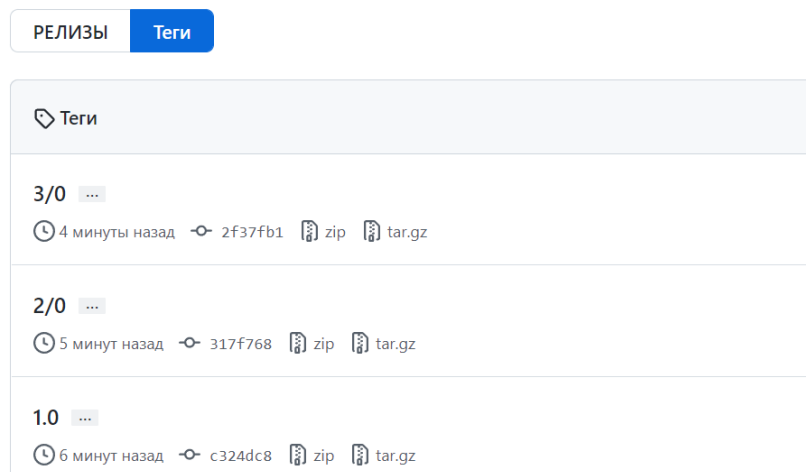


Рисунок 5.17 История тегов на уд. сервере

6. Просмотрел историю хранилища командой git log:

```
C:\Users\User\Desktop\1.2>git log --graph --pretty=oneline --abbrev-commit
* ba72cfa (HEAD -> main, origin/main, origin/HEAD) f
* 700508d i
* 12c3d18 d
* 2f37fb1 (tag: 3/0) c
* 317f768 (tag: 2/0) a
* ef019cf Merge branch 'main' of https://github.com/aikanyshkaukanbekova/1.2
|
| * 3914066 Update README.md
| * c324dc8 (tag: 1.0) red
|/
* cb81bce Update .gitignore
* 4107047 Initial commit
C:\Users\User\Desktop\1.2>
```

Рисунок 6. История коммитов

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

```
Администратор: Git CMD
C:\Users\User\Desktop\1.2>git show HEAD
commit ba72cfa1b15d2aad9f5092e018599f3563ad162 (HEAD -> main, origin/main, origin/HEAD)
Author: aikanyshkaukanbekova <ulanbekovaaajkanys5@gmail.com>
Date: Tue Mar 29 20:04:21 2022 +0300

f

diff --git a/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp b/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp
index f986934..8dcdda9 100644
--- a/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp
+++ b/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp
@@ -11,7 +11,11 @@ int main()
{
    cout << "a=" << a << "b=" << b;
    if (a > b)
    {
        cout << "\n a>b";
        cout << "\n a<b";
    }
    else
    {
        cout << "\n a>b";
    }
}
...skipping...
Author: aikanyshkaukanbekova <ulanbekovaaajkanys5@gmail.com>
Date: Tue Mar 29 20:04:21 2022 +0300

f

diff --git a/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp b/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp
index f986934..8dcdda9 100644
--- a/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp
+++ b/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp
@@ -11,7 +11,11 @@ int main()
{
    cout << "a=" << a << "b=" << b;
    if (a > b)
    {
        cout << "\n a>b";
        cout << "\n a<b";
    }
    else
    {
        cout << "\n a>b";
    }
}
}
```

Рисунок 7.1 Содержимое коммитов командами

```
Администратор: Git CMD

C:\Users\User\Desktop\1.2>git show HEAD~
commit 700508dc468358b6ba283c507c751edf17ecfce8
Author: aikanyshkaukanbekova <ulanbekovaaajkanys5@gmail.com>
Date: Tue Mar 29 20:01:52 2022 +0300

i
diff --git a/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp b/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp
index d50d1b4..f986934 100644
--- a/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp
+++ b/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp
@@ -9,5 +9,9 @@ int main()
     a = 5;
     b = 7;
     cout << "a=" << a << "b=" << b;
     if (a > b)
     {
         cout << "\n a>b";
     }
 }

C:\Users\User\Desktop\1.2>git show 317f768
commit 317f768a2d01a8df5d5eae6b663de1a7452ac05 (tag: 2/0)
Author: aikanyshkaukanbekova <ulanbekovaaajkanys5@gmail.com>
Date: Tue Mar 29 19:52:52 2022 +0300

a
diff --git a/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp b/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp
index 2addc97..b04498d 100644
--- a/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp
+++ b/ConsoleApplication1/ConsoleApplication1/ConsoleApplication1.cpp
@@ -5,5 +5,6 @@ using namespace std;
 int main()
 {
     cout << "Hello world!\n";
     int a, b;
 }
```

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.

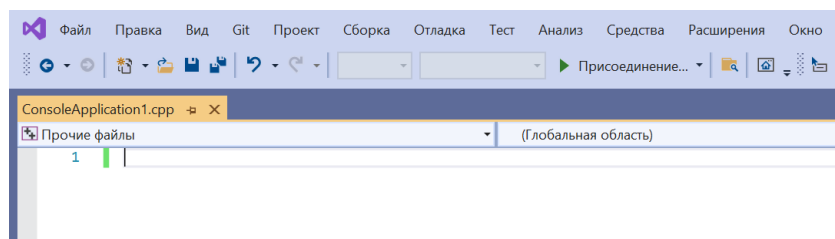
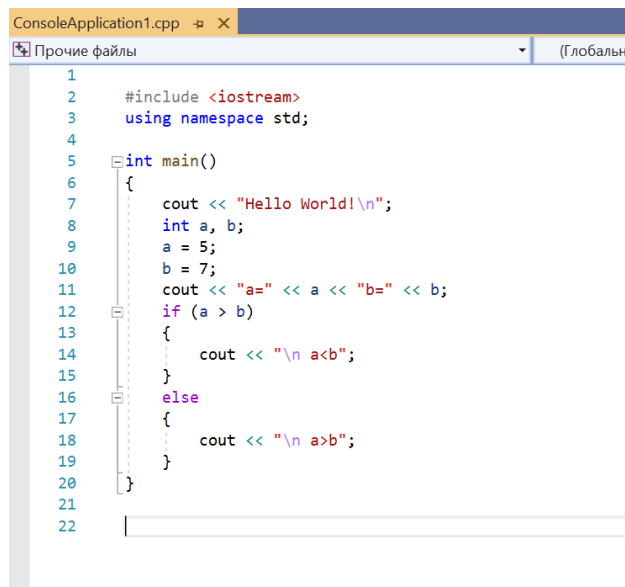


Рисунок 8.1 Удаление кода в файле ConsoleApplication.cpp

```
C:\Users\User\Desktop\1.2\ConsoleApplication1>cd ConsoleApplication1
C:\Users\User\Desktop\1.2\ConsoleApplication1\ConsoleApplication1>git checkout -- ConsoleApplication1.cpp
C:\Users\User\Desktop\1.2\ConsoleApplication1\ConsoleApplication1>_
```

Рисунок 8.2 checkout изменений файла ConsoleApplication.cpp



```
1
2 #include <iostream>
3 using namespace std;
4
5 int main()
6 {
7     cout << "Hello World!\n";
8     int a, b;
9     a = 5;
10    b = 7;
11    cout << "a=" << a << "b=" << b;
12    if (a > b)
13    {
14        cout << "\n a<b";
15    }
16    else
17    {
18        cout << "\n a>b";
19    }
20 }
21
22
```

Рисунок 8.3 Изменения в файле с программой после команды

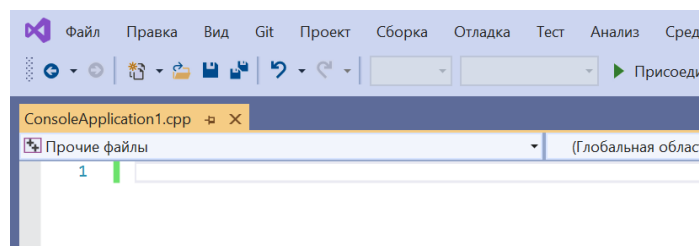
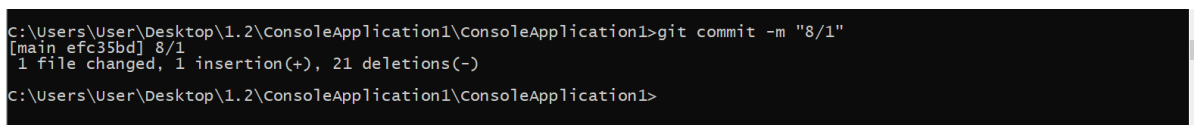


Рисунок 8.4 Удаление кода в файле с программой



```
C:\Users\User\Desktop\1.2\ConsoleApplication1\ConsoleApplication1>git commit -m "8/1"
[main efc35bd] 8/1
1 file changed, 1 insertion(+), 21 deletions(-)
C:\Users\User\Desktop\1.2\ConsoleApplication1\ConsoleApplication1>
```

Рисунок 8.5 Коммит изменений

Вывод: команда `git -checkout <FileName>` удаляет изменения произошедшие с файлом в репозитории до коммита.

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
```

Эта 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 помечать важные моменты.

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 удалённый сервер. Процесс аналогичен отправке веток — достаточно 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.