

**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

Хашиев Адам Мухарбекович

Ставрополь 2022

Выполнение работы:

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

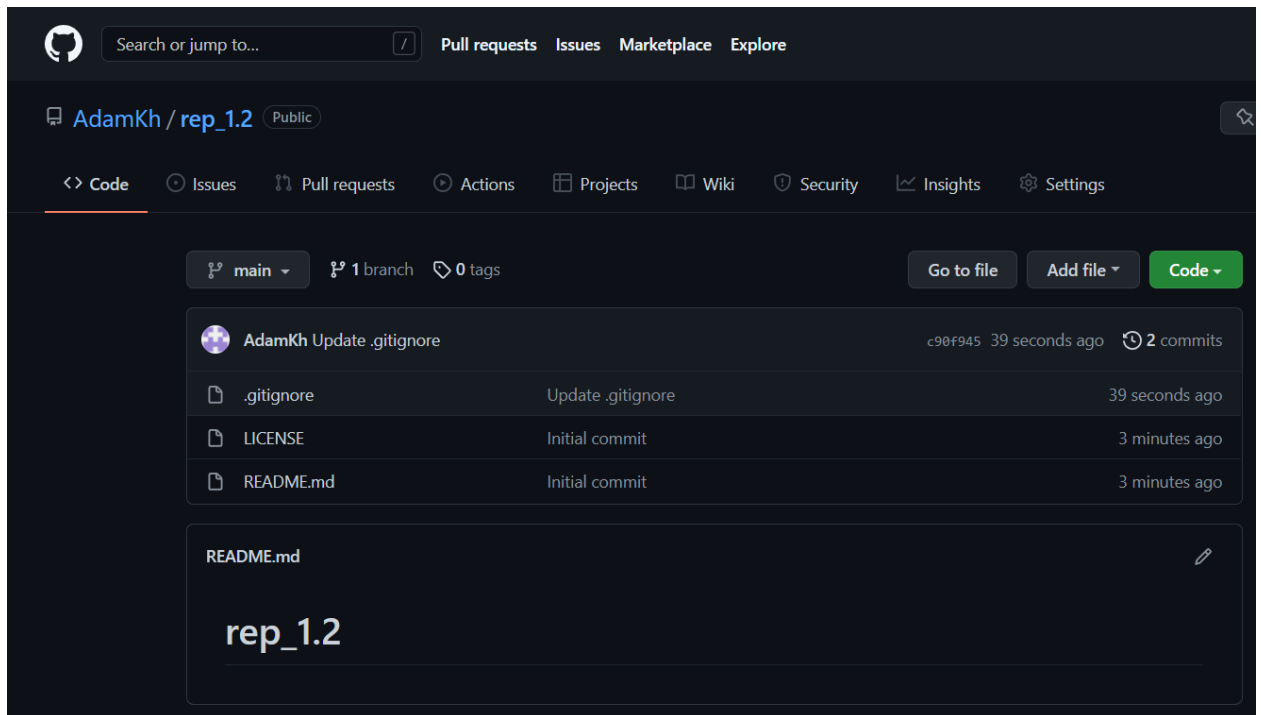


Рисунок 1.1 Созданный репозиторий в GitHub

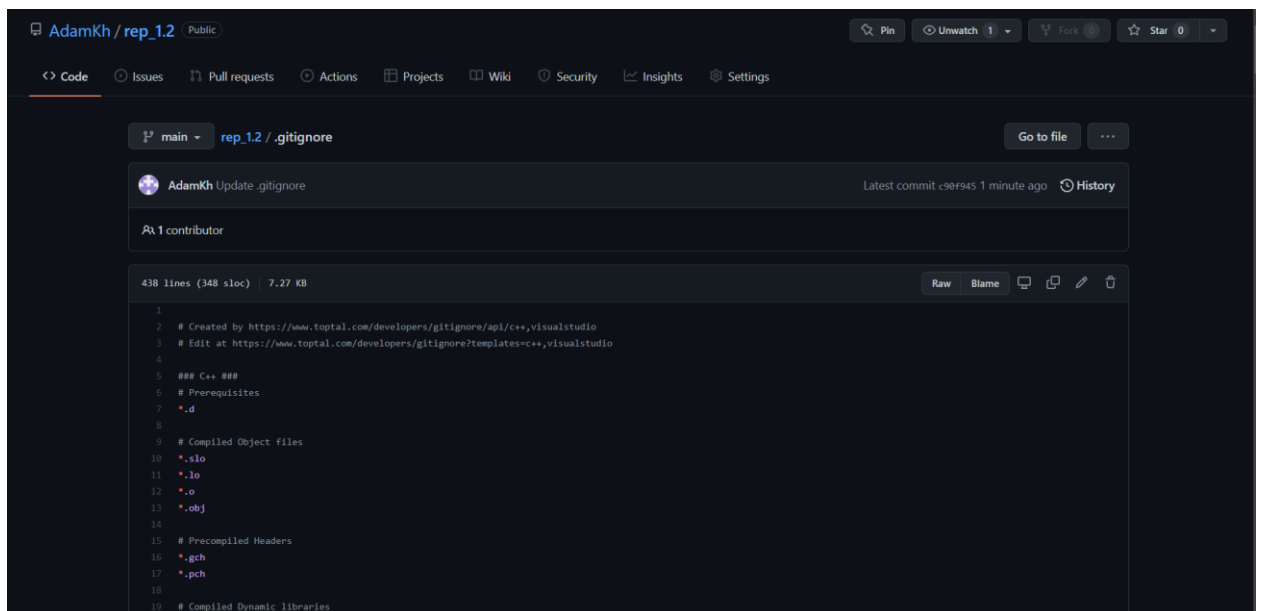


Рисунок 1.2 Изменения в файле .gitignore

2. Клонировал созданный репозиторий на раб. компьютер:

```
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2>git clone https://github.com/AdamKh/rep_1.2.git
Cloning into 'rep_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 1), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (8/8), 5.04 KiB | 573.00 KiB/s, done.
Resolving deltas: 100% (1/1), done.
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2>
```

Рисунок 2. Клонирование репозитория

3. Добавил информацию в README, закоммитил и запустил изменения на уд. сервер:

```
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>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\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>git add .
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>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:   README.md

C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>git commit -m "modified README"
[main 4041eea] modified README
1 file changed, 2 insertions(+), 1 deletion(-)
```

Рисунок 3.1 Коммит файла README.md

```
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>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
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>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), 412 bytes | 206.00 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/AdamKh/rep_1.2.git
   c90f945..4041eea  main -> main
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>
```

Рисунок 3.2 Пуш коммита

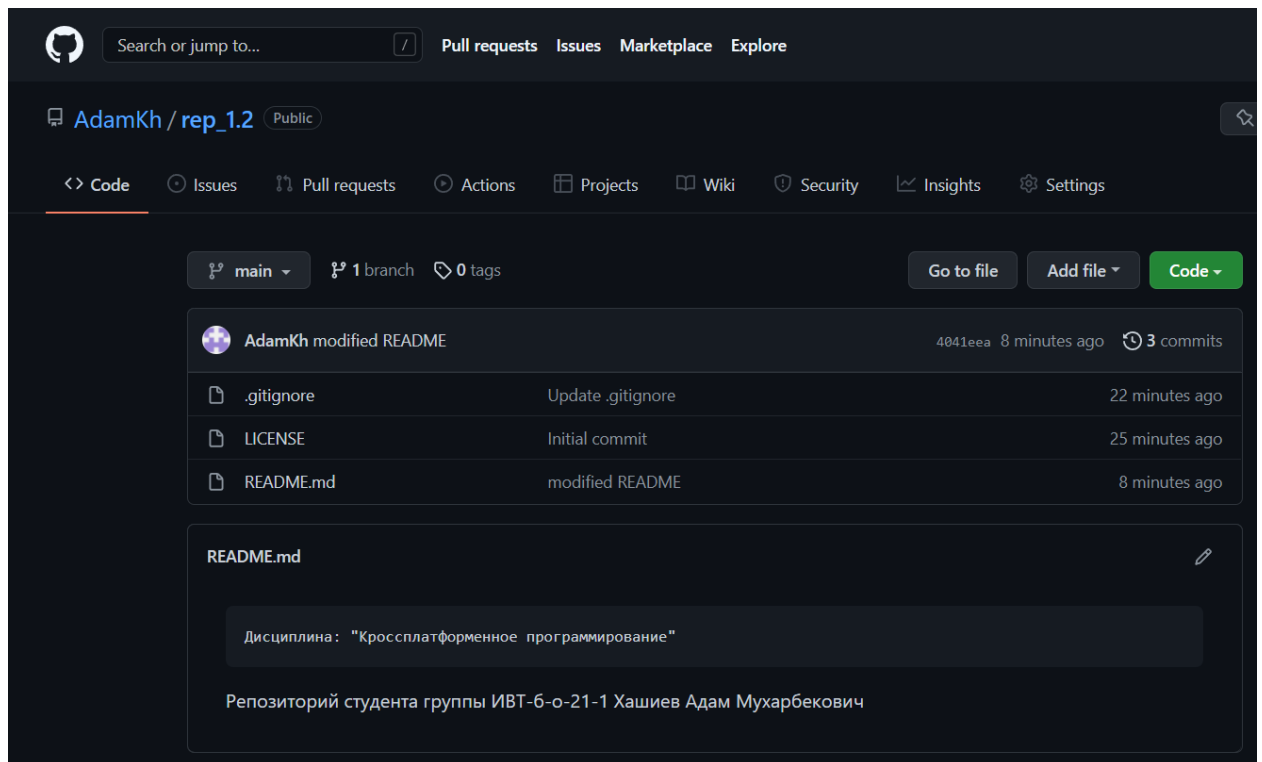


Рисунок 3.3 Изменения на уд. сервере

- Написал в репозитории небольшую программу, сделал коммит и пуш:

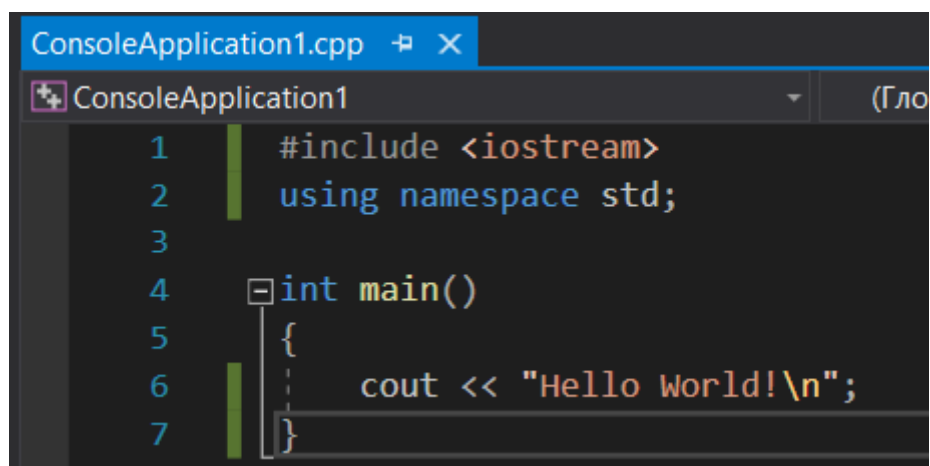


Рисунок 4.1 Изменения в программе

.git	11.03.2022 11:47	Папка с файлами	
.vs	11.03.2022 11:56	Папка с файлами	
prog	11.03.2022 11:57	Папка с файлами	
.gitignore	11.03.2022 11:37	Текстовый докум...	8 КБ
LICENSE	11.03.2022 11:37	Файл	2 КБ
README.md	11.03.2022 11:44	Файл "MD"	1 КБ

Рисунок 4.2 Добавлена папка с проектом на C++

```

C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>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)
  prog/

nothing added to commit but untracked files present (use "git add" to track)
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>git add .
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>git commit -m "added prog"
[main 11d7f5b] added prog
4 files changed, 207 insertions(+)
create mode 100644 prog/ConsoleApplication1/ConsoleApplication1.cpp
create mode 100644 prog/ConsoleApplication1/ConsoleApplication1.vcxproj
create mode 100644 prog/ConsoleApplication1/ConsoleApplication1.vcxproj.filters
create mode 100644 prog/prog.sln
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>git push
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (8/8), 2.60 KiB | 888.00 KiB/s, done.
Total 8 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/AdamKh/rep_1.2.git
  4041eea..11d7f5b  main -> main
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>_

```

Рисунок 4.3 Коммит и пуш программы на уд. сервер

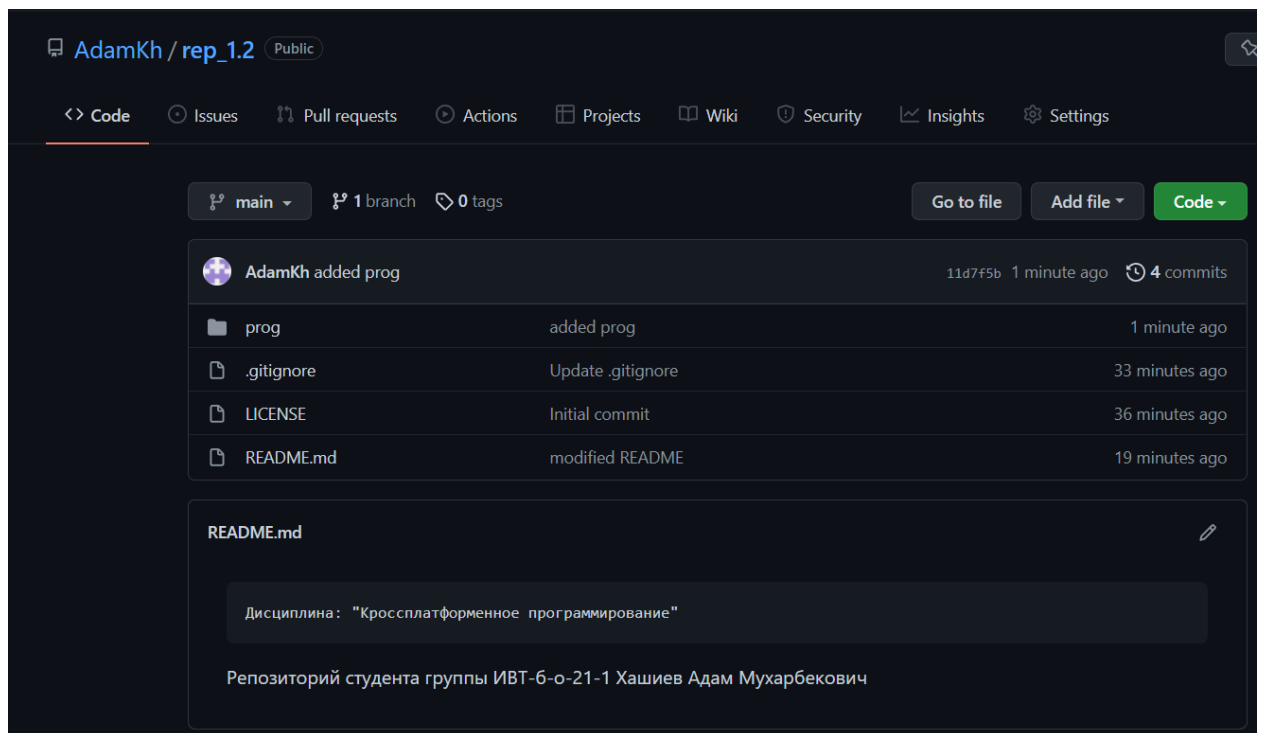
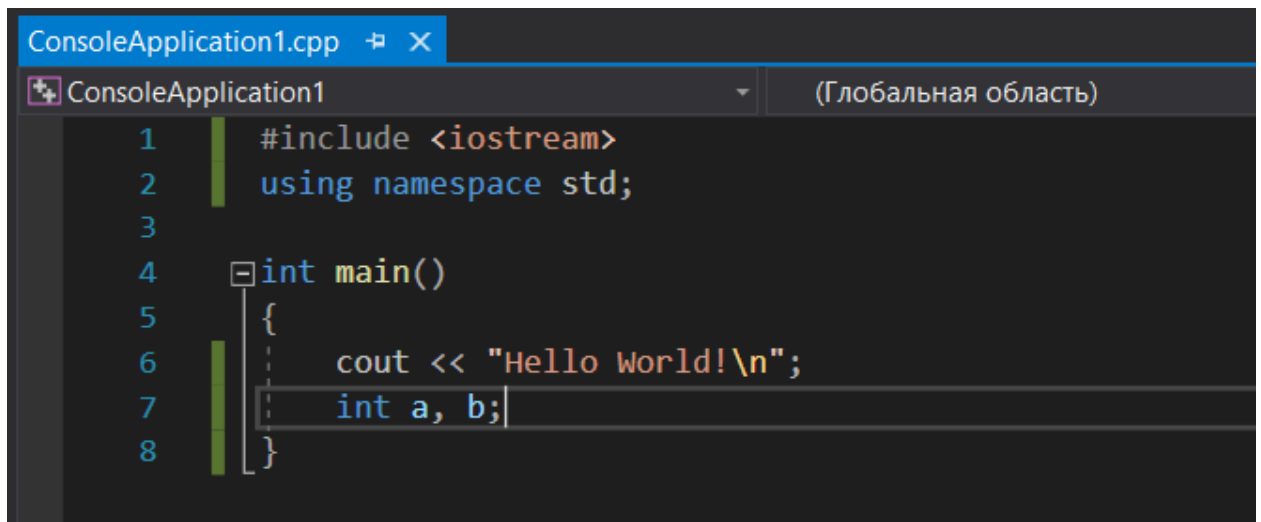


Рисунок 4.4 Изменения на уд. сервере

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



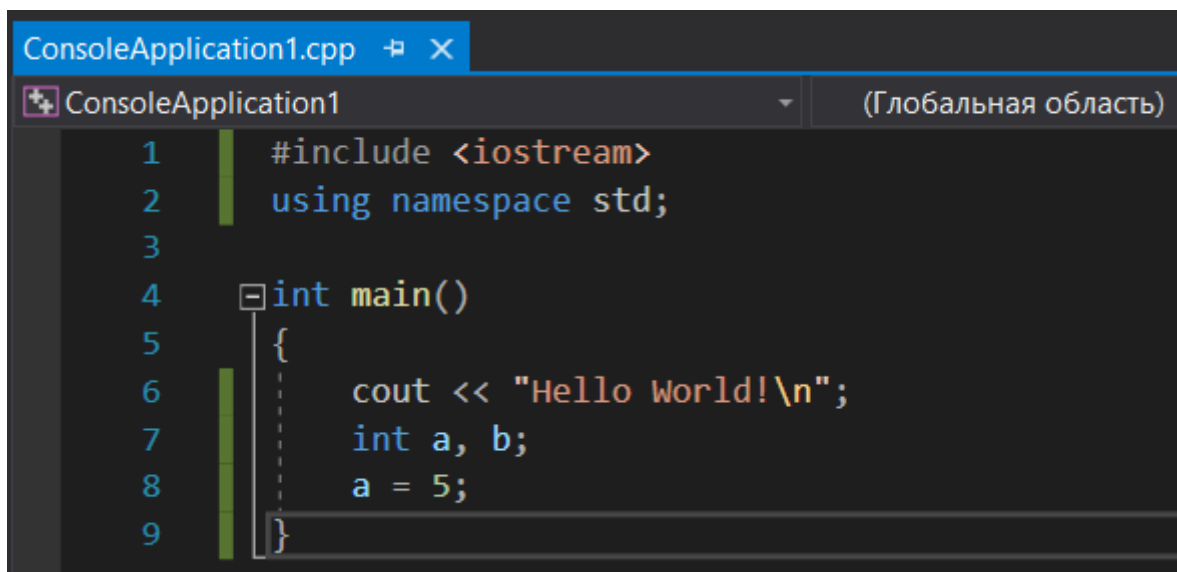
```
ConsoleApplication1.cpp  + X
ConsoleApplication1      (Глобальная область)

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

Рисунок 5.1 Изменения в программе

```
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\гер_1.2>git commit -am "added a, b"
[main b898517] added a, b
1 file changed, 1 insertion(+)
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\гер_1.2>
```

Рисунок 5.2 Коммит изменений



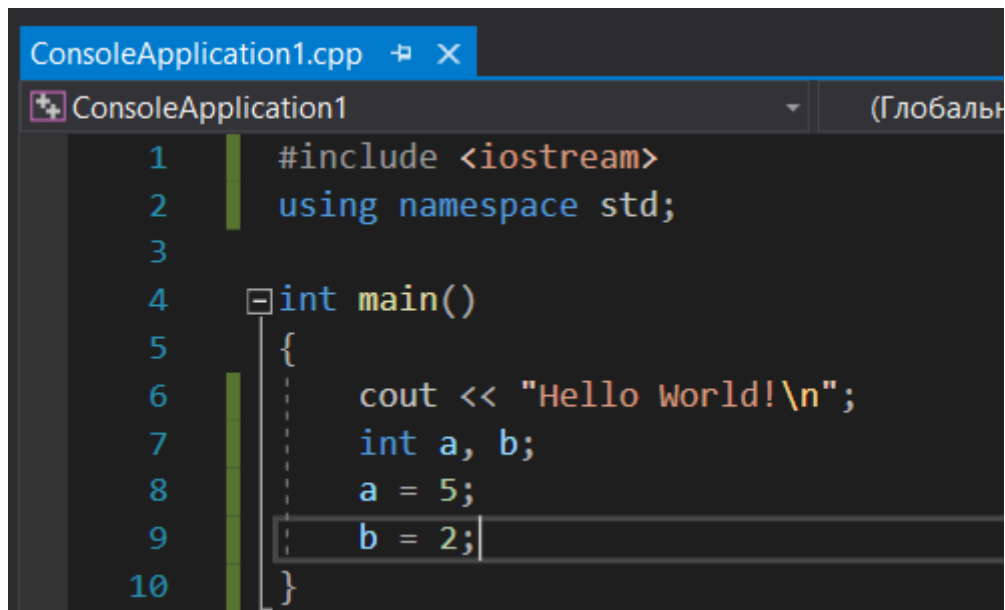
```
ConsoleApplication1.cpp  + X
ConsoleApplication1      (Глобальная область)

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

Рисунок 5.3 Изменения в программе

```
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\гер_1.2>git commit -am "added value for a"
[main 6d1851f] added value for a
1 file changed, 1 insertion(+)
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\гер_1.2>
```

Рисунок 5.4 Коммит изменений

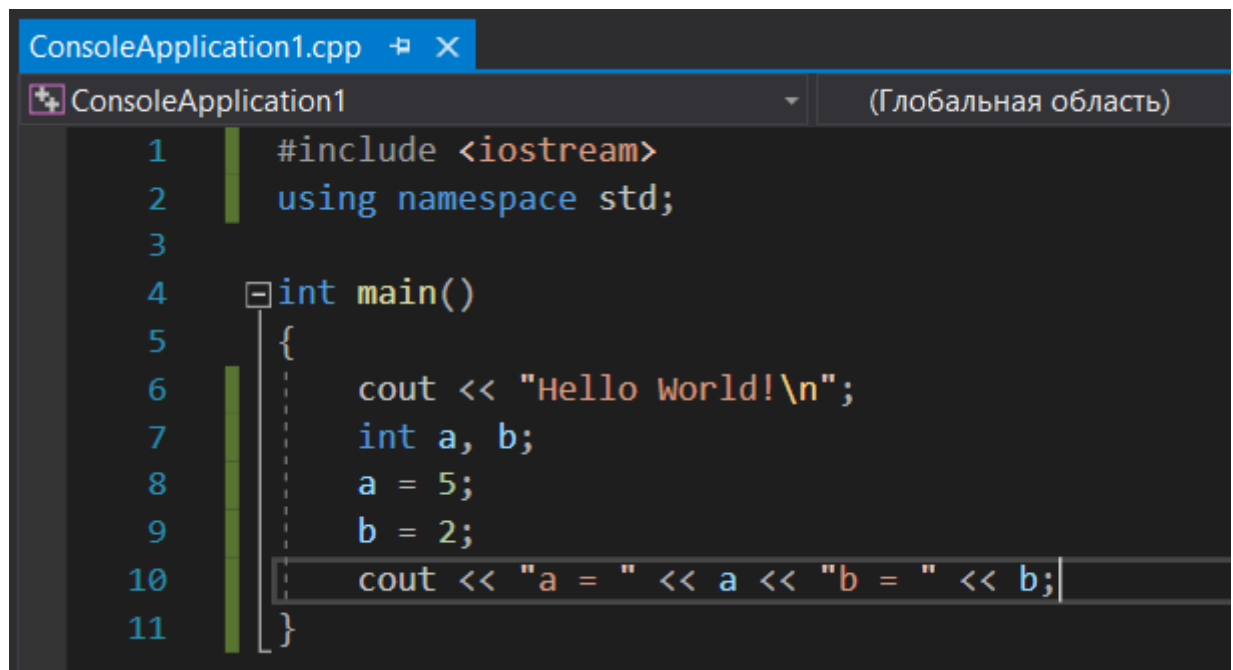


```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      cout << "Hello World!\n";
7      int a, b;
8      a = 5;
9      b = 2;
10 }
```

Рисунок 5.5 Изменения в программе

```
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>git commit -am "added value for b"
[main cb59ad1] added value for b
1 file changed, 1 insertion(+)
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>
```

Рисунок 5.6 Коммит изменений

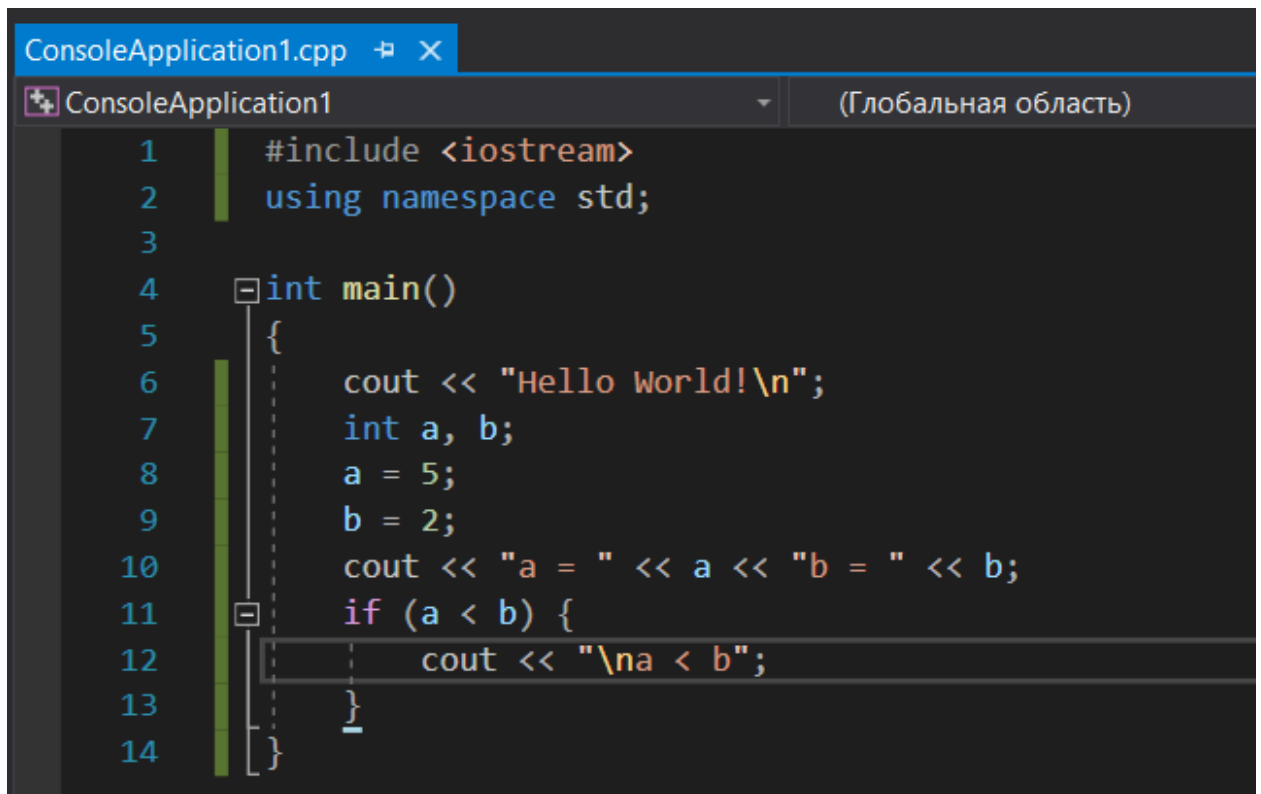


```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      cout << "Hello world!\n";
7      int a, b;
8      a = 5;
9      b = 2;
10     cout << "a = " << a << "b = " << b;
11 }
```

Рисунок 5.7 Изменения в программе

```
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>git commit -am "output of values a and b"
[main 79ff928] output of values a and b
1 file changed, 1 insertion(+)
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>
```

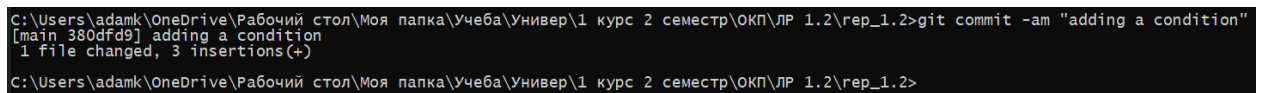
Рисунок 5.8 Коммит изменений



```
ConsoleApplication1.cpp  [icon] X
ConsoleApplication1  (Глобальная область)

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

Рисунок 5.9 Изменения в программе



```
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>git commit -am "adding a condition"
[main 380dfd9] adding a condition
1 file changed, 3 insertions(+)

C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>
```

Рисунок 5.10 Коммит изменений


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

Рисунок 5.9 Изменения в программе

```
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>git commit -am "added else"
[main 2a54daf] added else
1 file changed, 4 insertions(+)
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>
```

Рисунок 5.11 Коммит изменений

```
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>git push
Enumerating objects: 34, done.
Counting objects: 100% (34/34), done.
Delta compression using up to 8 threads
Compressing objects: 100% (30/30), done.
Writing objects: 100% (30/30), 2.45 KiB | 358.00 KiB/s, done.
Total 30 (delta 17), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (17/17), completed with 2 local objects.
To https://github.com/AdamKh/rep_1.2.git
 11d7f5b..2a54daf main -> main
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>
```

Рисунок 5.12 Пуш изменений

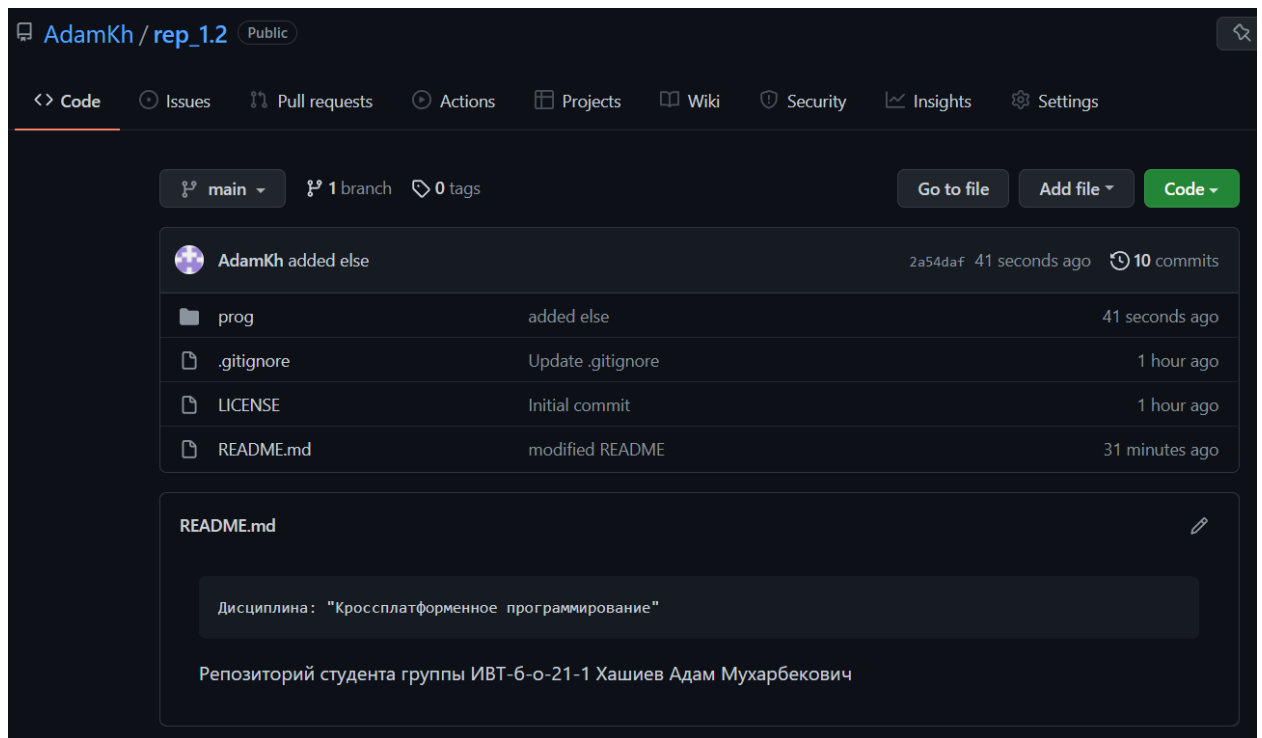


Рисунок 5.13 Изменения на уд. сервере

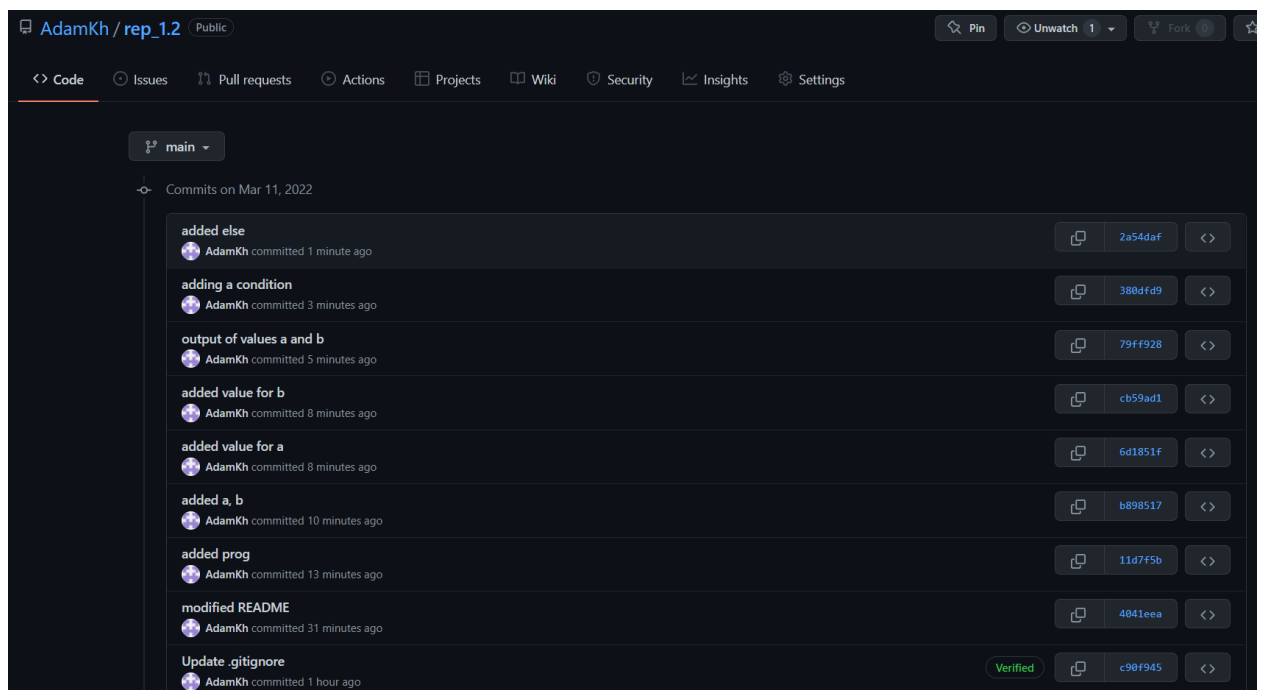


Рисунок 5.14 История коммитов на уд. сервере

```
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>git tag -a V1.0 -m "added prog" 11d7f5b3889aa85c1322ab8d759ecb663c9ce8a7
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>git tag V1.0
```

Рисунок 5.15 Присваивание тега коммиту

```
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>git tag -a V1.1 -m "added value for b" cb59ad1d04c80f1f5c5b52d072be6a5e207fb763
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>git tag V1.0
V1.0
V1.1
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>
```

Рисунок 5.15 Присваивание тега коммиту

```
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>git tag -a V2.0 -m "added else" 2a54d4f752a6ee1d5bc533f411bb33bbce4a096b
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>git tag
V1.0
V1.1
V2.0
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>
```

Рисунок 5.15 Присваивание тега коммиту

```
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>git push origin --tags
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 426 bytes | 213.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/AdamKh/rep_1.2.git
 * [new tag]          V1.0 -> V1.0
 * [new tag]          V1.1 -> V1.1
 * [new tag]          V2.0 -> V2.0
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>
```

Рисунок 5.18 Пуш тегов

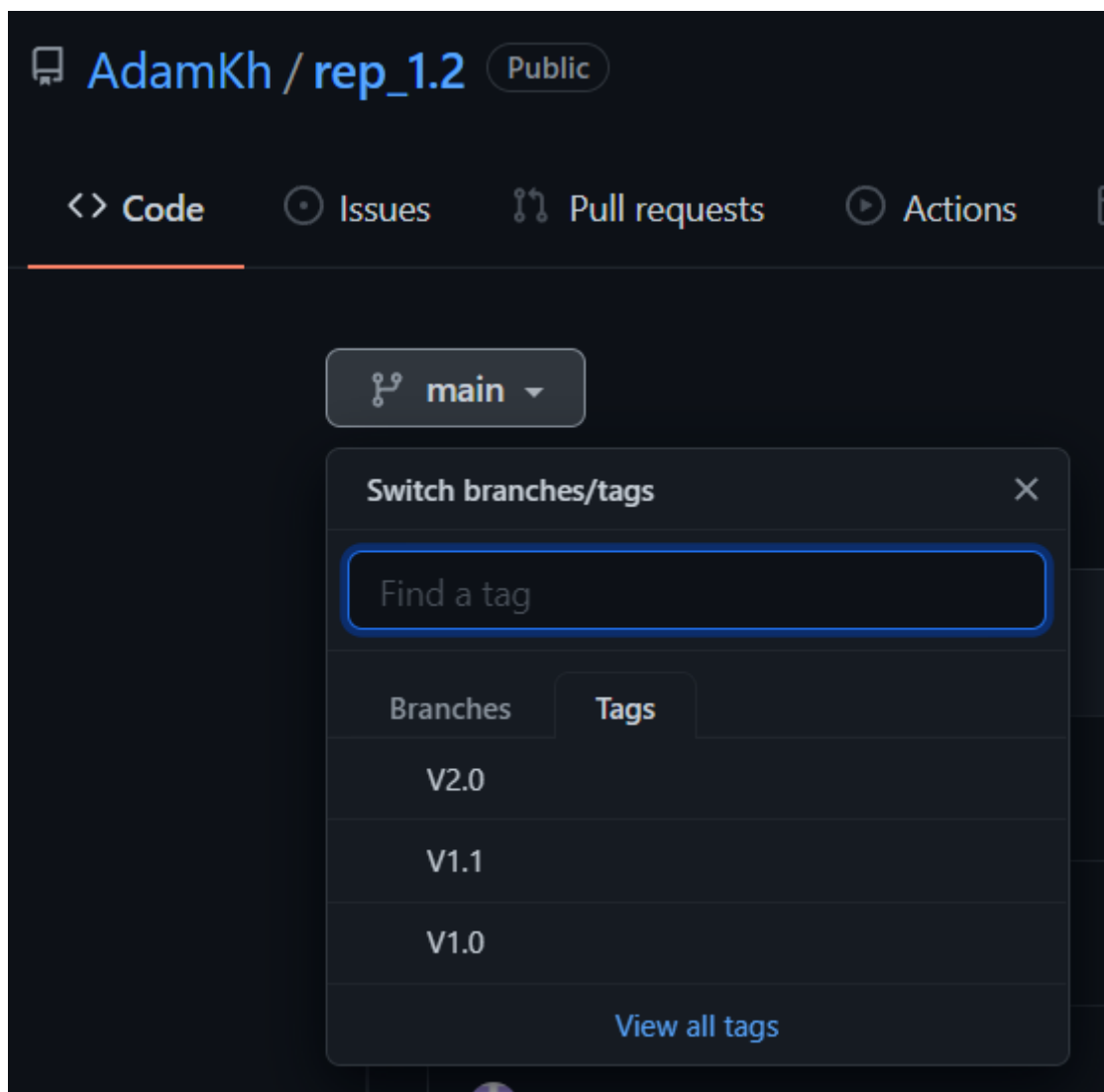


Рисунок 5.19 История тегов на уд. сервере

6. Просмотрел историю хранилища командой git log:

```
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>git log --graph --pretty=oneline --abbrev-commit
* 2a54daf (HEAD -> main, tag: V2.0, origin/main, origin/HEAD) added else
* 380df99 adding a condition
* 79ff928 output of values a and b
* cb59ad1 (tag: V1.1) added value for b
* 6d1851f added value for a
* b898517 added a, b
* 11d7f5b (tag: V1.0) added prog
* 4041eea modified README
* c90f945 Update .gitignore
* d7bca34 Initial commit

C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>
```

Рисунок 6. История коммитов

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

```
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>git show HEAD
commit 2a54daf752a6ee1d5bc533f411bb33bbce4a096b (HEAD -> main, tag: V2.0, origin/main, origin/HEAD)
Author: AdamKh <adamkhashiev2004@mail.ru>
Date: Fri Mar 11 12:17:39 2022 +0300

    added else

diff --git a/prog/ConsoleApplication1/ConsoleApplication1.cpp b/prog/ConsoleApplication1/ConsoleApplication1.cpp
index ab2f590..f7445cf 100644
--- a/prog/ConsoleApplication1/ConsoleApplication1.cpp
+++ b/prog/ConsoleApplication1/ConsoleApplication1.cpp
@@ -11,4 +11,8 @@ int main()
     if (a < b) {
         cout << "\na < b";
     }
+    else
+    {
+        cout << "\nb < a";
+    }
 }
\ No newline at end of file

C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>git show HEAD~
commit 380df9d0813cff4abfefbaae291a67dcb7cc0e
Author: AdamKh <adamkhashiev2004@mail.ru>
Date: Fri Mar 11 12:15:34 2022 +0300

    adding a condition

diff --git a/prog/ConsoleApplication1/ConsoleApplication1.cpp b/prog/ConsoleApplication1/ConsoleApplication1.cpp
index f6ca0f5..ab2f590 100644
--- a/prog/ConsoleApplication1/ConsoleApplication1.cpp
+++ b/prog/ConsoleApplication1/ConsoleApplication1.cpp
@@ -8,4 +8,7 @@ int main()
     a = 5;
     b = 2;
     cout << "a = " << a << "b = " << b;
+    if (a < b) {
+        cout << "\na < b";
+    }
 }
\ No newline at end of file
```

Рисунок 7.1 Содержимое коммитов командами

```
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>git show cb59ad1
commit cb59ad1d04c80f1f5cfb52d072be6a5e207fb763 (tag: V1.1)
Author: AdamKh <adamkhashiev2004@mail.ru>
Date: Fri Mar 11 12:11:03 2022 +0300

    added value for b

diff --git a/prog/ConsoleApplication1/ConsoleApplication1.cpp b/prog/ConsoleApplication1/ConsoleApplication1.cpp
index 9b8b803..be2e089 100644
--- a/prog/ConsoleApplication1/ConsoleApplication1.cpp
+++ b/prog/ConsoleApplication1/ConsoleApplication1.cpp
@@ -6,4 +6,5 @@ int main()
     cout << "Hello World!\n";
     int a, b;
     a = 5;
+    b = 2;
 }
\ No newline at end of file

C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2>
```

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.

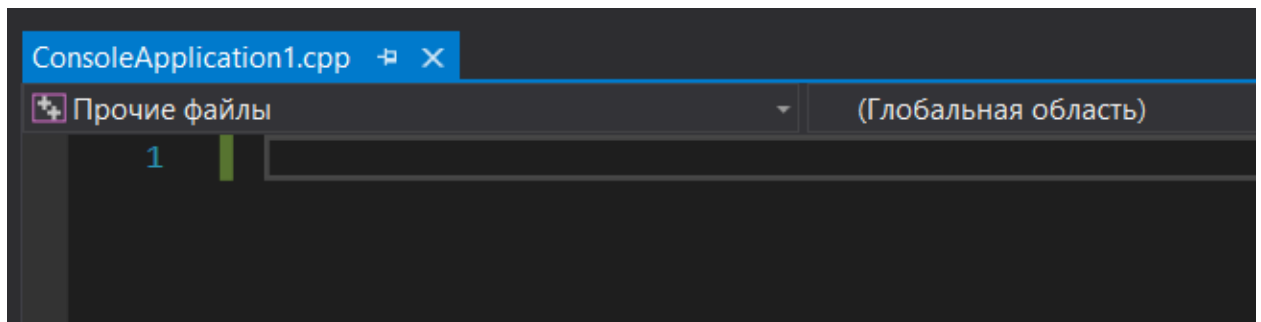


Рисунок 8.1 Удаление кода в файле ConsoleApplication.cpp

```
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2\prog\ConsoleApplication1
>git checkout -- ConsoleApplication1.cpp
C:\Users\adamk\OneDrive\Рабочий стол\Моя папка\Учеба\Универ\1 курс 2 семестр\ОКП\ЛР 1.2\rep_1.2\prog\ConsoleApplication1
>
```

Рисунок 8.2 checkout изменений файла ConsoleApplication.cpp

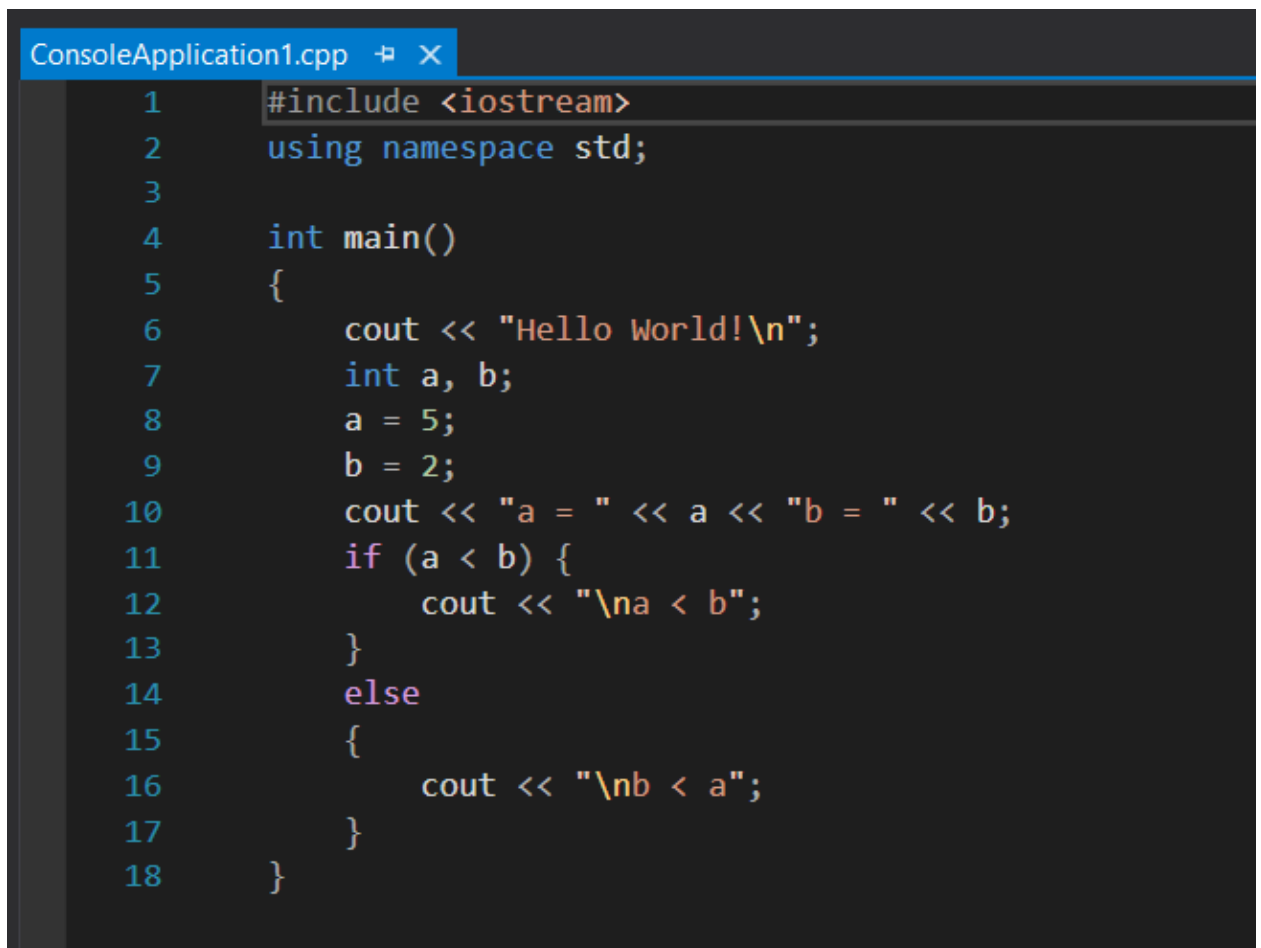


Рисунок 8.3 Изменения в файле с программой после команды

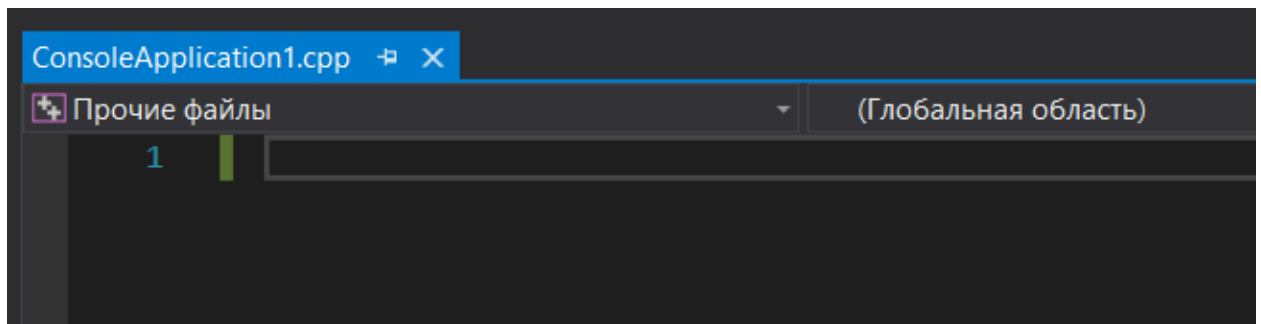


Рисунок 8.4 Удаление кода в файле с программой

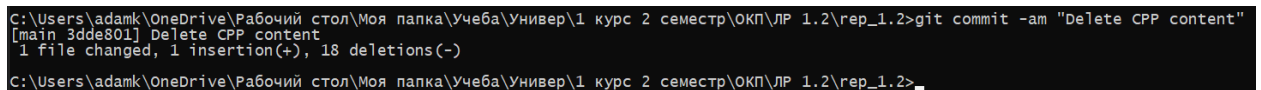


Рисунок 8.5 Коммит изменений

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 удоб-

ным 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 простая: `git push <remote-name> <branch-name>`.

10. Как выполнить просмотр удаленного репозитория?

Для просмотра удалённого репозитория, можно использовать команду `git remote show <remote>`.

11. Каково назначение тэгов Git?

Теги - это ссылки указывающие на определённые версии кода/написанной программы. Они удобны чтобы в случае чего вернуться к нужному моменту. Также при помощи тегов можно пометить 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 выполнить 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 лёгкий 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 удалённые ветки, у которых нет локального аналога.