

## Лабораторная работа 1.2

### Исследование основных возможностей Git и GitHub

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

**Выполнил:** Баканов Артем ИТС-б-о-22-1

#### Теоретические сведения

После того, как вы создали несколько коммитов или же клонировали репозиторий с уже существующей историей коммитов, вероятно Вам понадобится возможность посмотреть что было сделано — историю коммитов. Одним из основных и наиболее мощных инструментов для этого является команда `git log`.

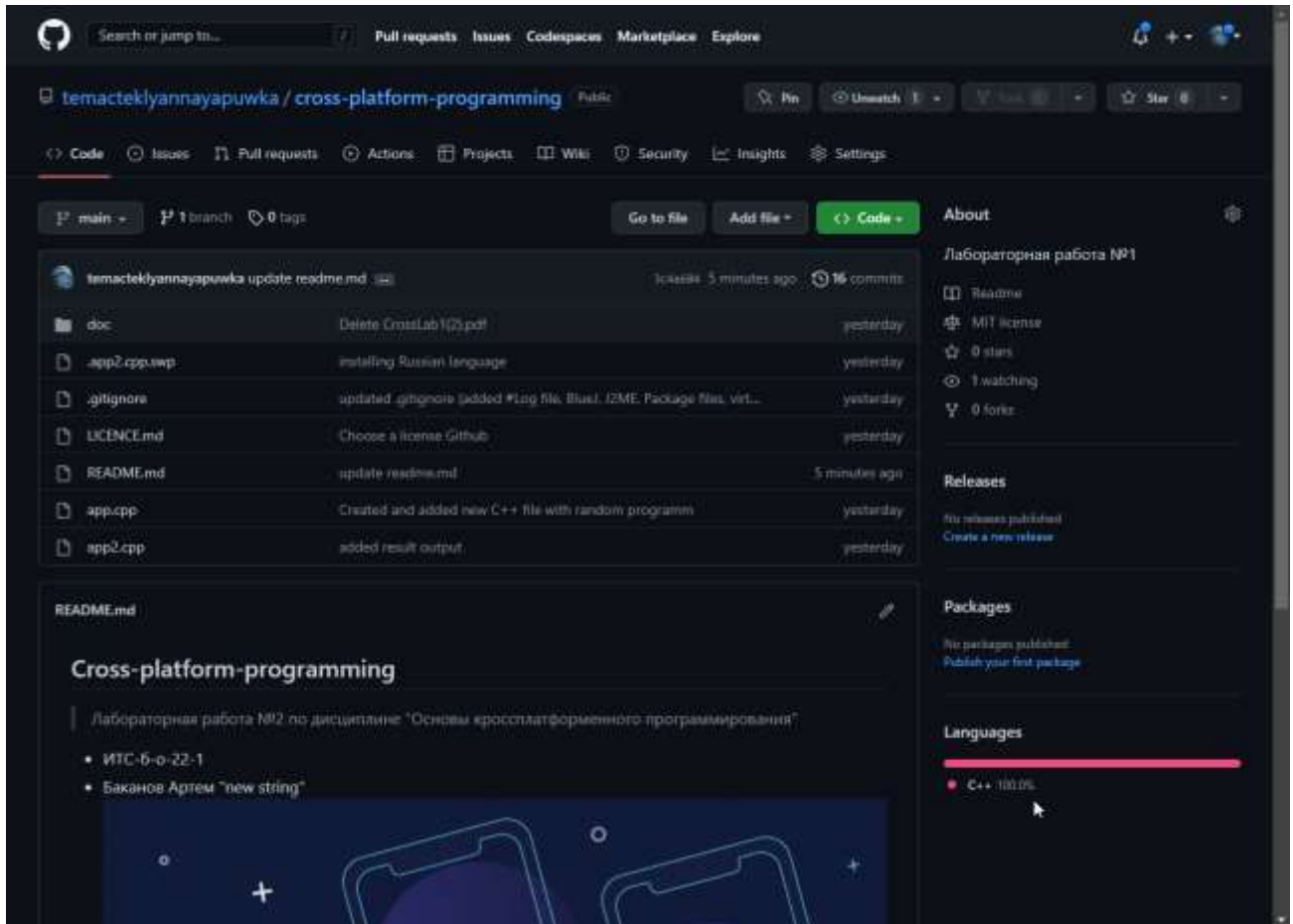
По умолчанию (без аргументов) `git log` перечисляет коммиты, сделанные в репозитории в обратном к хронологическому порядку — последние коммиты находятся вверху. Из примера можно увидеть, что данная команда перечисляет коммиты с их SHA-1 контрольными суммами, именем и электронной почтой автора, датой создания и сообщением коммита. Команда `git log` имеет очень большое количество опций для поиска коммитов по разным критериям. Рассмотрим наиболее популярные из них.

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

Следующей действительно полезной опцией является `--pretty`. Эта опция меняет формат вывода. Существует несколько встроенных вариантов отображения. Опция `oneline` выводит каждый коммит в одну строку, что может быть очень удобным если вы просматриваете большое количество коммитов. К тому же, опции `short`, `full` и `fuller` делают вывод приблизительно в том же формате, но с меньшим или большим количеством информации.

## Методика и порядок выполнения работы

1. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и выбранный Вами язык программирования.



2. Проработайте примеры лабораторной работы. Отрадите вывод на консоли при выполнении команд git в отчете для лабораторной работы.

### Просмотр истории коммитов

```
lena@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2
$ git clone https://github.com/temacteklyannayapuwka/cross-platform-programming-v.2.git
Cloning into 'cross-platform-programming-v.2'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
```

Команда «git clone https://...».



```

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git log --pretty=format:"%h - %an, %ar : %s"
10e419f - Artem Bakanov, 24 minutes ago : Added saving the array in variable A
695f330 - Artem Bakanov, 26 minutes ago : Added input selection (auto/manual)
72feb72 - Artem Bakanov, 32 minutes ago : added menu function for user interaction + 2D array refinement
a047ef6 - Artem Bakanov, 35 minutes ago : added multiplication function
1a103ad - Artem Bakanov, 39 minutes ago : added matrix output function
a5c2bed - Artem Bakanov, 41 minutes ago : creating pseudo-random matrix elements
75a735a - Artem Bakanov, 42 minutes ago : added matrix input
1843655 - Artem Bakanov, 45 minutes ago : created new app/installing C++ libraries
3465b82 - Artem Bakanov, 68 minutes ago : Initial commit

```

Комманда «git log --pretty=format:"%h - %an, %ar : %s"».

```

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git log --pretty=format:"%h %s" --graph
* 10e419f Added saving the array in variable A
* 695f330 Added input selection (auto/manual)
* 72feb72 added menu function for user interaction + 2D array refinement
* a047ef6 added multiplication function
* 1a103ad added matrix output function
* a5c2bed creating pseudo-random matrix elements
* 75a735a added matrix input
* 1843655 created new app/installing C++ libraries
* 3465b82 Initial commit

```

Комманда «git log --pretty=format:"%h %s" --graph».

```

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git log --since=30.minutes
commit 10e419ffb73e6918462c6a3ded6d7e50050c01a0 (HEAD -> main, tag: v1.2, origin/main, origin/HEAD)
Author: Artem Bakanov <rtmbakanov@gmail.com>
Date: Tue Feb 28 22:45:24 2023 +0300

    Added saving the array in variable A

commit 695f330885c829c74da94810bca7174724697b1
Author: Artem Bakanov <rtmbakanov@gmail.com>
Date: Tue Feb 28 22:44:02 2023 +0300

    Added input selection (auto/manual)

```

## Ограничение вывода

Комманда «git log --since=30.minutes».

```

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git log -s #include
commit 10e419ffb73e6918462c6a3ded6d7e50050c01a0 (HEAD -> main, tag: v1.2, origin/main, origin/HEAD)
Author: Artem Bakanov <rtmbakanov@gmail.com>
Date: Tue Feb 28 22:45:24 2023 +0300

    Added saving the array in variable A

commit 695f330885c829c74da94810bca7174724697b1
Author: Artem Bakanov <rtmbakanov@gmail.com>
Date: Tue Feb 28 22:44:02 2023 +0300

    Added input selection (auto/manual)

commit 72feb72a189a70e8b0b6f6f05e378af243042526 (tag: v1.1)
Author: Artem Bakanov <rtmbakanov@gmail.com>
Date: Tue Feb 28 22:37:47 2023 +0300

    added menu function for user interaction + 2D array refinement

commit a047ef609108d71e2d64d8305c501a4332436e08
Author: Artem Bakanov <rtmbakanov@gmail.com>
Date: Tue Feb 28 22:34:59 2023 +0300

    added multiplication function

commit 1a103ad0b3ed40b325fc77f21c355ea007103e5d (tag: v1.0)
Author: Artem Bakanov <rtmbakanov@gmail.com>
Date: Tue Feb 28 22:31:12 2023 +0300

```

Комманда «git log -s #include».

## Операции отмены

```
lena@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git commit -m "added trash string for test --amend"
[main dc69bab] added trash string for test --amend
1 file changed, 3 insertions(+)

lena@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ touch forgotten_file

lena@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git add forgotten_file
fatal: pathspec 'forgotten_file' did not match any files

lena@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ touch forgotten_file.cpp

lena@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git add forgotten_file.cpp

lena@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git commit -m "added trash string for test --amend" --amend
[main ef1862e] added trash string for test --amend
Date: Tue Feb 28 23:15:03 2023 +0300
2 files changed, 3 insertions(+)
create mode 100644 forgotten_file.cpp
```

Комманда «git commit -m "...» –amend».

## Отмена индексации файла

```
lena@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ touch logs.txt

lena@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git add *

lena@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
(use "git push" to publish your local commits)

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   logs.txt

lena@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git reset HEAD logs.txt

lena@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
(use "git push" to publish your local commits)

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

nothing added to commit but untracked files present (use "git add" to track)
```

Комманда «git reset HEAD».



## Отмена изменений в файле

```
tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git commit -m "added logs.txt for test "checkout""
[main cc2d708] added logs.txt for test checkout
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 logs.txt

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ vim logs.txt

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 2 commits.
(use "git push" to publish your local commits)

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:   logs.txt

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

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git checkout -- logs.txt

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 2 commits.
(use "git push" to publish your local commits)

nothing to commit, working tree clean
```

Комманда «git checkout --».

## Просмотр удалённых репозиториев

```
tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git remote -v
crosslabv.2 https://github.com/temaacteklyannayapuwka/cross-platform-programming-v.2.git (fetch)
crosslabv.2 https://github.com/temaacteklyannayapuwka/cross-platform-programming-v.2.git (push)
origin https://ghp_tLEezcDQxxYgoCEN1ETA4IU5YRUZ2w1VmqDB@github.com/temaacteklyannayapuwka/cross-platform-programming-v.2.git (fetch)
origin https://ghp_tLEezcDQxxYgoCEN1ETA4IU5YRUZ2w1VmqDB@github.com/temaacteklyannayapuwka/cross-platform-programming-v.2.git (push)
```

Комманда «git remote -v».

## Добавление удалённых репозиториев

```
tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git remote -v
crosslabv.2 https://github.com/temaacteklyannayapuwka/cross-platform-programming-v.2.git (fetch)
crosslabv.2 https://github.com/temaacteklyannayapuwka/cross-platform-programming-v.2.git (push)
origin https://ghp_tLEezcDQxxYgoCEN1ETA4IU5YRUZ2w1VmqDB@github.com/temaacteklyannayapuwka/cross-platform-programming-v.2.git (fetch)
origin https://ghp_tLEezcDQxxYgoCEN1ETA4IU5YRUZ2w1VmqDB@github.com/temaacteklyannayapuwka/cross-platform-programming-v.2.git (push)

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git fetch crosslabv.2
From https://github.com/temaacteklyannayapuwka/cross-platform-programming-v.2
 * [new branch]      main      -> crosslabv.2/main
```

Комманда «git fetch».

## Просмотр удаленного репозитория

```
tena@DESKTOP-B9HE1QH MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git remote show origin
* remote origin
  Fetch URL: https://ghp_tLEEzcDQXxYgoCENIETA4IU5YRUZ2w1YmqD0@github.com/tenacteklyannayapumka/cross-platform-programming-v.2.git
  Push URL: https://ghp_tLEEzcDQXxYgoCENIETA4IU5YRUZ2w1YmqD0@github.com/tenacteklyannayapumka/cross-platform-programming-v.2.git
  HEAD branch: main
  Remote branch:
    main tracked
  Local branch configured for 'git pull':
    main merges with remote main
  Local ref configured for 'git push':
    main pushes to main (up to date)
```

Команда «git remote show origin».

## Удаление и переименование удалённых репозитория

```
tena@DESKTOP-B9HE1QH MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git remote rename crosslabv.2 crossv.2
Renaming remote references: 100% (1/1), done.
```

Команда «git remote rename».

## Работа с тегами

### 1. Просмотр списка тегов.

```
tena@DESKTOP-B9HE1QH MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git tag
v1.0
v1.1
v1.2
v1.3
```

Команда «git tag».

### 2. Создание тегов.

```
tena@DESKTOP-B9HE1QH MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git tag -a v1.3 -m "v1.3"
```

Команда «git tag -a».

### 3. Обмен тегами.

```
tena@DESKTOP-B9HE1QH MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git push origin --tags
Enumerating objects: 1, done.
Counting objects: 100% (1/1), done.
Writing objects: 100% (1/1), 156 bytes | 75.00 KiB/s, done.
Total 1 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/tenacteklyannayapumka/cross-platform-programming-v.2.git
 * [new tag]          v1.3 -> v1.3
```

Команда «git push origin --tags».

#### 4.Переход на тег.

```
tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git checkout -- logs.txt

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git commit -m "test rollback to a given version part 2"
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:   logs.txt

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

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git add .

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git commit -m "test rollback to a given version part 2"
[main d6fda38] test rollback to a given version part 2
1 file changed, 1 insertion(+), 1 deletion(-)

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 277 bytes | 138.00 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/temacteklyannayapumka/cross-platform-programming-v.2.git
 c8e2fb9..d6fda38  main -> main

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git reset --hard HEAD~1
HEAD is now at c8e2fb9 test rollback to a given version
```

Комманда «git checkout -- ...».

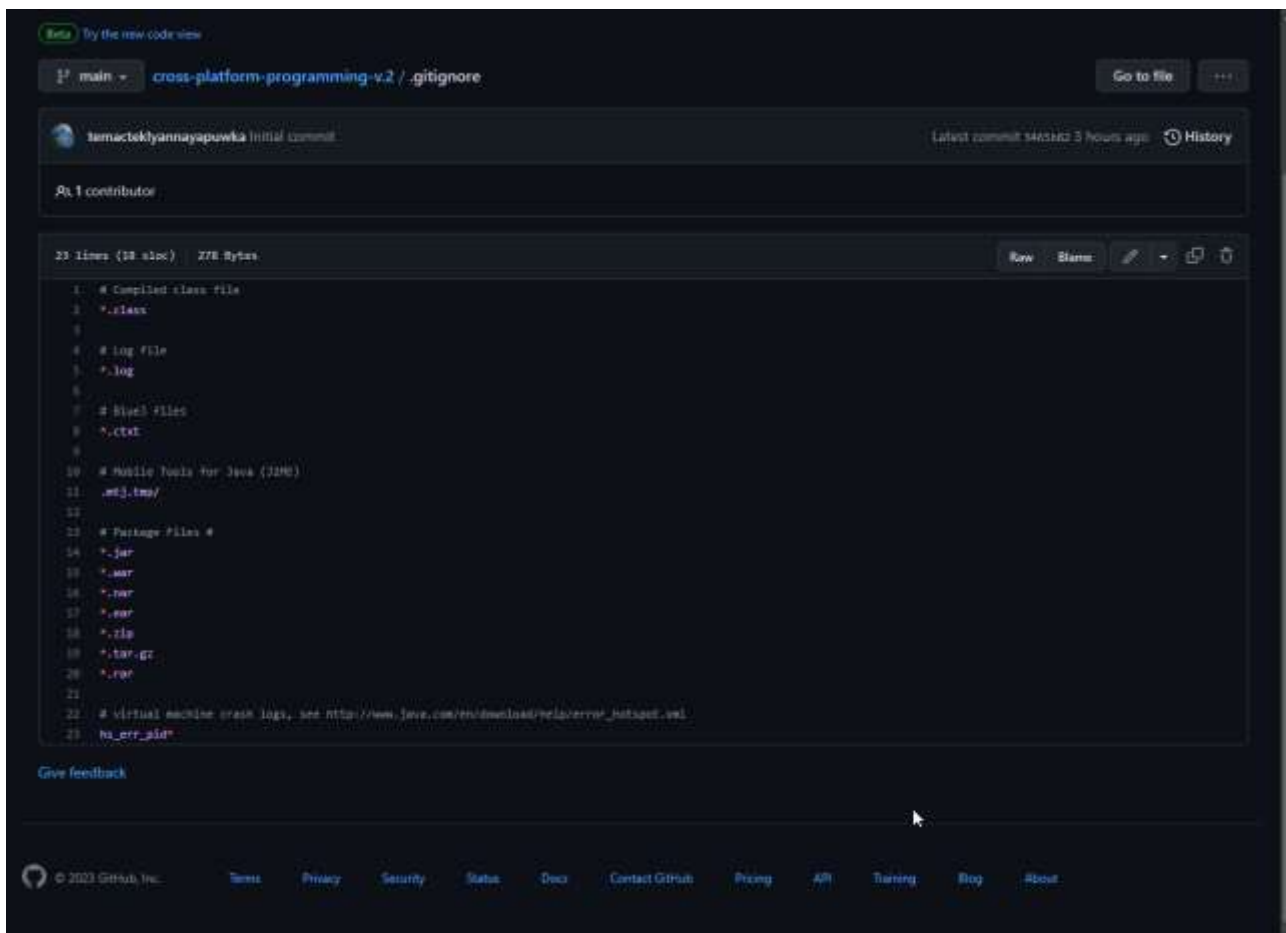
3. Выполните клонирование созданного репозитория на рабочий компьютер.

```
tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs
$ git clone https://github.com/temacteklyannayapumka/cross-platform-programming-v.2.git
Cloning into 'cross-platform-programming-v.2'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
```

Клонирование репозитория.



4. Дополните файл .gitignore необходимыми правилами для выбранного языка программирования и интегрированной среды разработки.



The screenshot shows a code editor with the file `cross-platform-programming-v.2/.gitignore` open. The file contains the following rules:

```
1 # Compiled class file
2 *.class
3
4 # log file
5 *.log
6
7 # BlueJ files
8 *.ctxt
9
10 # native tools for Java (J2SE)
11 *.tmp
12
13 # Package Files #
14 *.jar
15 *.war
16 *.rar
17 *.war
18 *.zip
19 *.tar.gz
20 *.rar
21
22 # virtual machine crash logs, see http://www.java.com/en/download/help/error_hotspot.xml
23 hs_err_pid*
```

Добавлены правила для выбранного языка.

5. Добавьте в файл README.md информацию о дисциплине, группе и ФИО студента, выполняющего лабораторную работу.

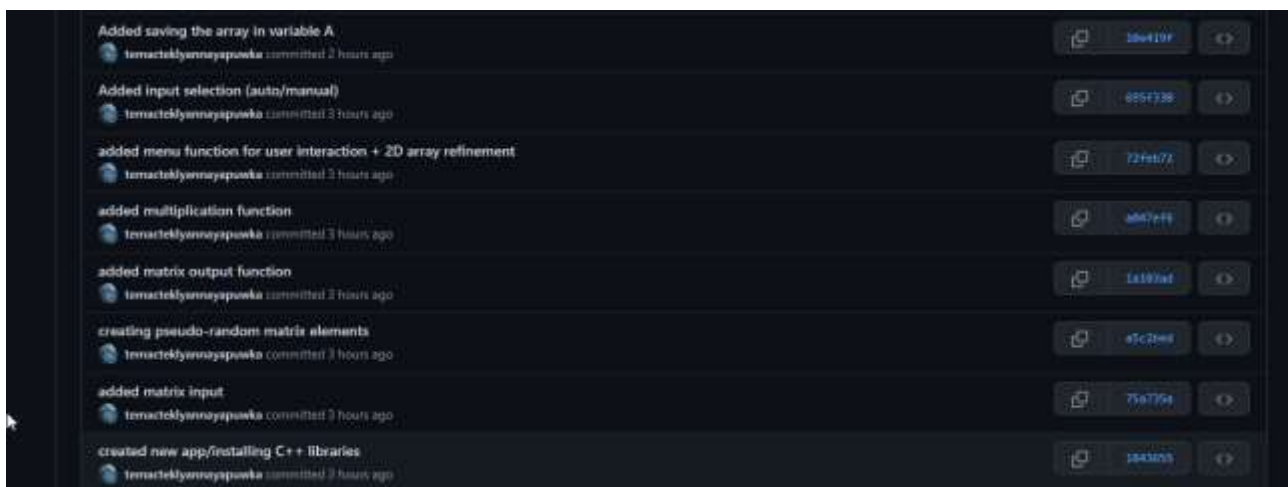


The screenshot shows a code editor with the file `README.md` open. The file contains the following text:

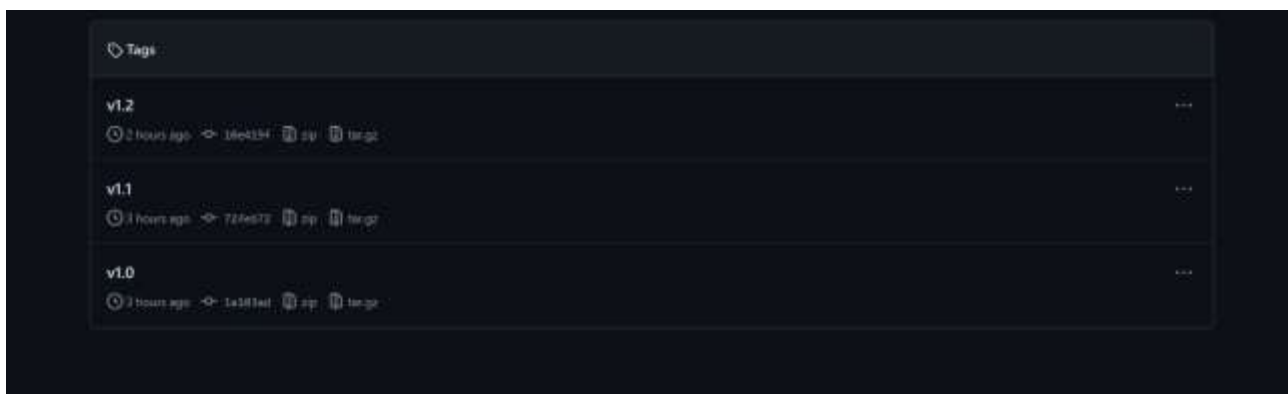
```
1 # Cross-platform-programming-v.2
2 - Лабораторная работа #2 по "Основы кроссплатформенного программирования"
3
4 # Cross-platform-programming
5 - Лабораторная работа #2 по дисциплине "Основы кроссплатформенного программирования".
6 - ИТС-6-4-22-1
7 - Базовый курс
8
9 +
10 + ([https://techtelgraph.co.uk/wp-content/uploads/2022/10/cross-platform.jpg])
```

Добавлены ФИО и группа.

6. Напишите небольшую программу на выбранном Вами языке программирования. Фиксируйте изменения при написании программы в локальном репозитории. Должно быть сделано не менее 7 коммитов, отмеченных не менее 3 тэгами.



Коммиты написания программы.



Теги коммитов.

7. Просмотреть историю (журнал) хранилища командой `git log` . Например, с помощью команды `git log --graph --pretty=oneline --abbrev-commit` . Добавить скриншот консоли с выводом в отчет по лабораторной работе.

```
tena@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git log --graph --pretty=oneline --abbrev-commit
* c4e2fb9 (HEAD -> main) test rollback to a given version
* cc2d708 (tag: v1.3) added logs.txt for test checkout
* ef1862e added trash string for test --amend
* 10e419f (tag: v1.2, crossv.2/main) Added saving the array in variable A
* 695f330 Added input selection (auto/manual)
* 72feb72 (tag: v1.1) added menu function for user interaction + 2D array refinement
* a047ef6 added multiplication function
* 1a103ad (tag: v1.0) added matrix output function
* a5c2bed creating pseudo-random matrix elements
* 75a735a added matrix input
* 1843655 created new app/installing C++ libraries
* 3465b02 Initial commit
```

Скриншот вывода консоли.

8.Просмотреть содержимое коммитов командой `git show <ref>` , где <ref>.

```
tena@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git show HEAD
commit c4e2fb994e0493d6fdc2f2062c87780be47d1ef7 (HEAD -> main)
Author: Artem Bakanov <rtmbakanov@gmail.com>
Date: Tue Feb 28 23:43:48 2023 +0300

    test rollback to a given version

diff --git a/logs.txt b/logs.txt
index a69de29..b5e88d7 100644
--- a/logs.txt
+++ b/logs.txt
@@ -0,0 +1 @@
+bla bla bla bla bla

tena@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ |
```

Скриншот последнего коммита.

```
tena@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming-v.2 (main)
$ git show HEAD~1
commit cc2d708adecea0199e53a3f8bc3f07fc018a8fb6 (tag: v1.3)
Author: Artem Bakanov <rtmbakanov@gmail.com>
Date: Tue Feb 28 23:21:33 2023 +0300

    added logs.txt for test checkout

diff --git a/logs.txt b/logs.txt
new file mode 100644
index 0000000..e69de29
```

Скриншот предпоследнего коммита.



## Вопросы для защиты работы

1. За просмотр истории коммитов отвечает команда `git log`. В сочетании с различными параметрами эта команда выводит историю по-разному.

`git log -p`, расширенный вывод истории,

`git log --oneline`, короткая запись,

`git log --stat --graph`, история в виде дерева

2. Кроме опций для форматирования вывода, `git log` имеет ряд полезных ограничительных параметров, то есть параметров, которые дают возможность отобразить часть коммитов. Параметр `-n`, который отображает только два последних коммита. На самом деле, вы можете задать `-<n>`, где `n` это количество отображаемых коммитов.

А вот параметры, ограничивающие по времени, такие как `--since` и `--until`, весьма полезны. Например, следующая команда выдаёт список коммитов, сделанных за последние две недели:

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

3. Если вы хотите изменить только сообщение вашего последнего коммита, это очень просто:

```
$ git commit --amend
```

Эта команда откроет в вашем текстовом редакторе сообщение вашего последнего коммита, для того, чтобы вы могли его исправить. Когда вы сохраните его и закроете редактор, будет создан новый коммит, содержащий это сообщение, который теперь и будет вашим последним коммитом.

Если вы создали коммит и затем хотите изменить зафиксированный снимок, добавив или изменив файлы (возможно, вы забыли добавить вновь созданный файл, когда совершали изначальный коммит), то процесс выглядит в основном так же. Вы добавляете в индекс необходимые изменения, редактируя файл и выполняя для него `git add` или `git rm` для отслеживаемого файла, а последующая команда `git commit --amend` берет вашу текущую область подготовленных изменений и делает её снимок для нового коммита.

4. Отмена может потребоваться, если вы сделали коммит слишком рано, например, забыв добавить какие-то файлы или комментарий к коммиту. Если



вы хотите переделать коммит — внесите необходимые изменения, добавьте их в индекс и сделайте коммит ещё раз, указав параметр `--amend`:

```
$ git commit --amend
```

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

```
$ git reset HEAD CONTRIBUTING.md
```

Unstaged changes after reset:

```
M    CONTRIBUTING.md
```

```
$ git status
```

On branch master

Changes to be committed:

(use "git reset HEAD <file>..." to unstage)

```
renamed:    README.md -> README
```

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git checkout -- <file>..." to discard changes in working directory)

```
modified:   CONTRIBUTING.md
```

6. Распределённый репозиторий — это, фактически, удалённая копия вашей программы, которую вы делаете с помощью `git` (или другой системы контроля версий) на удалённом сервере, доступном через интернет. После этого все, кому вы дадите доступ (в т.ч. и вы сами), смогут скачивать вашу программу, вносить в неё изменения и загружать их на удалённый сервер с помощью всё того же `git`.

7. Для того, чтобы просмотреть список настроенных удалённых репозиториях, вы можете запустить команду `git remote`. Она выведет названия доступных удалённых репозиториях. Если вы клонировали репозиторий, то увидите как минимум `origin` — имя по умолчанию, которое `Git` даёт серверу, с которого производилось клонирование

8. Чтобы добавить новый удаленный репозиторий, выполните команду `git remote add` в терминале в каталоге, в котором хранится репозиторий.

Команда `git remote add` принимает два аргумента:

имя удаленного репозитория, например, `origin`;

URL-адрес удаленного репозитория,

9. `$ git fetch [remote-name]`

Данная команда связывается с указанным удалённым проектом и забирает все те данные проекта, которых у вас ещё нет. После того как вы выполнили команду, у вас должны появиться ссылки на все ветки из этого удалённого проекта, которые вы можете просмотреть или слить в любой момент.

Когда вы хотите поделиться своими наработками, вам необходимо отправить их в удалённый репозиторий. Команда для этого действия простая: `git push <remote-name> <branch-name>`. Чтобы отправить вашу ветку `master` на сервер `origin` (повторимся, что клонирование обычно настраивает оба этих имени автоматически), вы можете выполнить следующую команду для отправки ваших коммитов:

```
$ git push origin master
```

10. Для того, чтобы просмотреть список настроенных удалённых репозиториях, вы можете запустить команду `git remote`. Она выведет названия доступных удалённых репозиториях. Если вы клонировали репозиторий, то увидите как минимум `origin` — имя по умолчанию, которое Git даёт серверу, с которого производилось клонирование:

11. «Git» теги — это метки, которыми можно пометить коммиты в истории коммитов, хранящихся в Git-репозитории. Часто тегами помечают коммиты, после которых выходит очередной релиз компьютерной программы, если речь идет о проекте написания некой компьютерной программы.

12. Git использует два основных типа тегов: легковесные и аннотированные.

Создание аннотированного тега в Git выполняется легко. Самый простой способ — это указать `-a` при выполнении команды `tag`:

Легковесный тег — это ещё один способ пометить коммит. По сути, это контрольная сумма коммита, сохранённая в файл — больше никакой

информации не хранится. Для создания легковесного тега не передавайте опций -a, -s и -m, укажите только название:

13. Это запускает `git fsck --unreachable`, используя все ссылки, доступные `refs/`, необязательно с дополнительным набором объектов, указанных в командной строке, и удаляет все распакованные объекты, недоступные для любого из этих головных объектов, из базы данных объектов. Кроме того, он удаляет распакованные объекты, которые также находятся в пакетах, запустив `git prune-packed`. Он также удаляет записи из `.git/shallow`, которые недоступны ни по одной ссылке.

Вывод: исследовал базовые возможности системы контроля версий Git для работы с локальными репозиториями.