

## **Лабораторная работа 1.1**

### **Исследование основных возможностей Git и GitHub**

**Цель работы:** исследовать базовые возможности системы контроля версий Git и веб-сервиса для хостинга IT-проектов GitHub.

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

#### **Теоретические сведения**

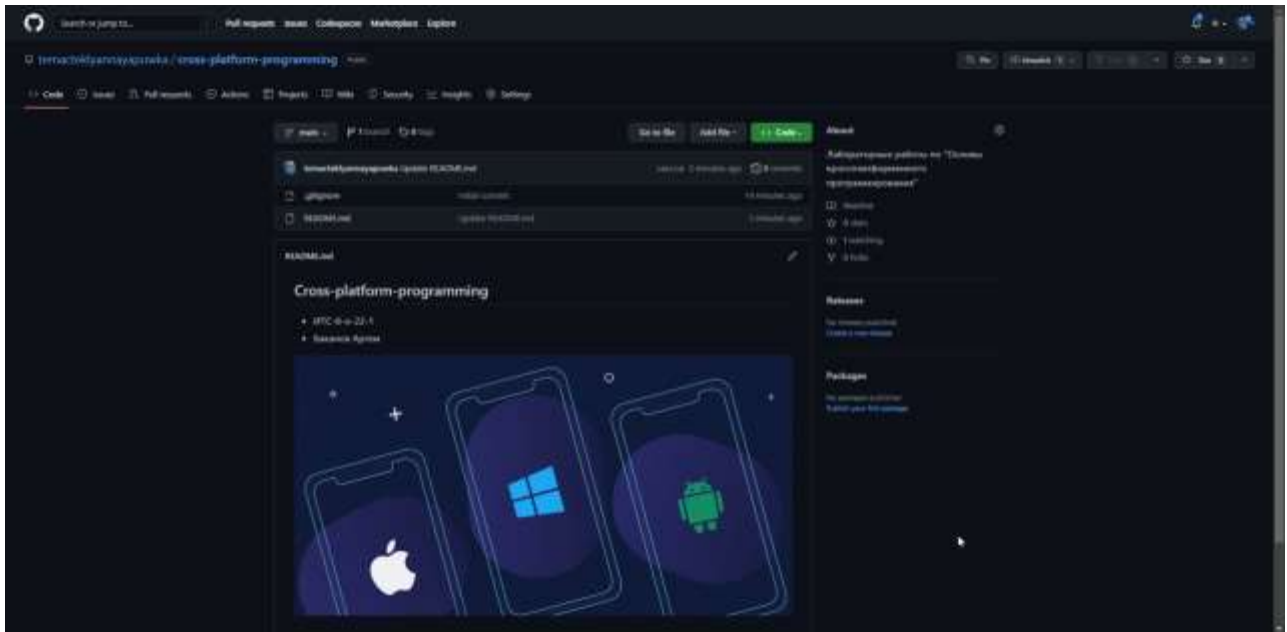
Система контроля версий (СКВ) – это система, регистрирующая изменения в одном или нескольких файлах с тем, чтобы в дальнейшем была возможность вернуться к определённым старым версиям этих файлов. Программисты обычно помещают в систему контроля версий исходные коды программ, но на самом деле под версионный контроль можно поместить файлы практически любого типа.

Локальные системы контроля версий – Многие люди в качестве метода контроля версий применяют копирование файлов в отдельную директорию (возможно даже, директорию с отметкой по времени, если они достаточно сообразительны). Данный подход очень распространён из-за его простоты, однако он невероятно сильно подвержен появлению ошибок. Можно легко забыть, в какой директории вы находитесь, и случайно изменить не тот файл или скопировать не те файлы, которые вы хотели.

Для того, чтобы решить эту проблему, программисты давным-давно разработали локальные СКВ с простой базой данных, которая хранит записи о всех изменениях в файлах, осуществляя тем самым контроль ревизий (рис. 1.1).

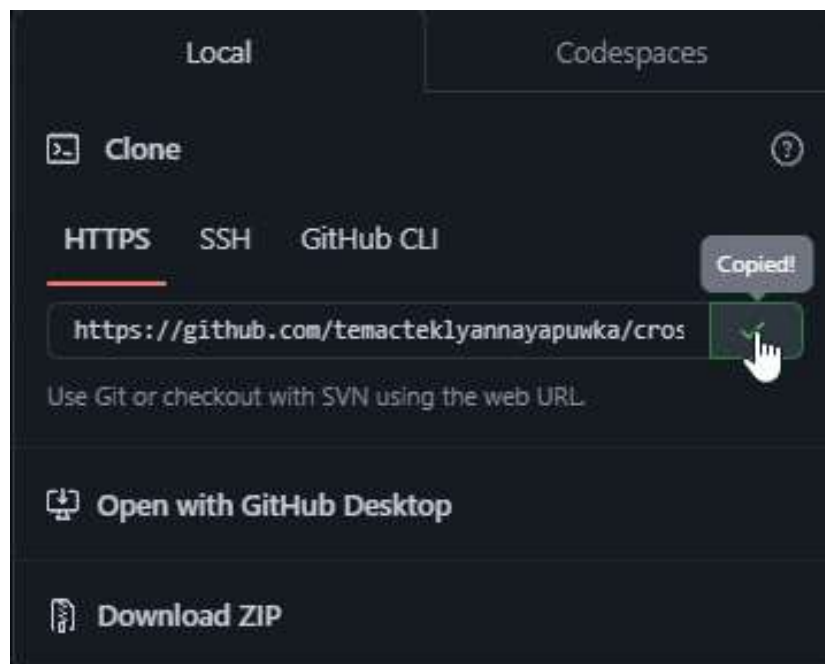
## Методика и порядок выполнения работы

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



Создан репозиторий на GitHub.

2. Выполните клонирование созданного репозитория на рабочий компьютер.



Клонирование репозитория с сайта GitHub.

```
tema@DESKTOP-B9HE1QR MINGW64 /c/WINDOWS/System32 (master)
$ cd /d/Crosslabs

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs
$ git clone https://github.com/temaeteklyannayapuwka/cross-platform-programming.git
Cloning into 'cross-platform-programming'...
remote: Enumerating objects: 10, done.
remote: Counting objects: 100% (10/10), done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 10 (delta 2), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (10/10), done.
Resolving deltas: 100% (2/2), done.

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs
$ ls -la
total 12
drwxr-xr-x 1 tema 197121 0 Feb 27 15:26 ./
drwxr-xr-x 1 tema 197121 0 Feb 27 15:24 ../
drwxr-xr-x 1 tema 197121 0 Feb 27 15:26 cross-platform-programming/

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs
$ cd /d/Crosslabs/cross-platform-programming

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming (main)
$ |
```

Клонирование репозитория на локальный компьютер.

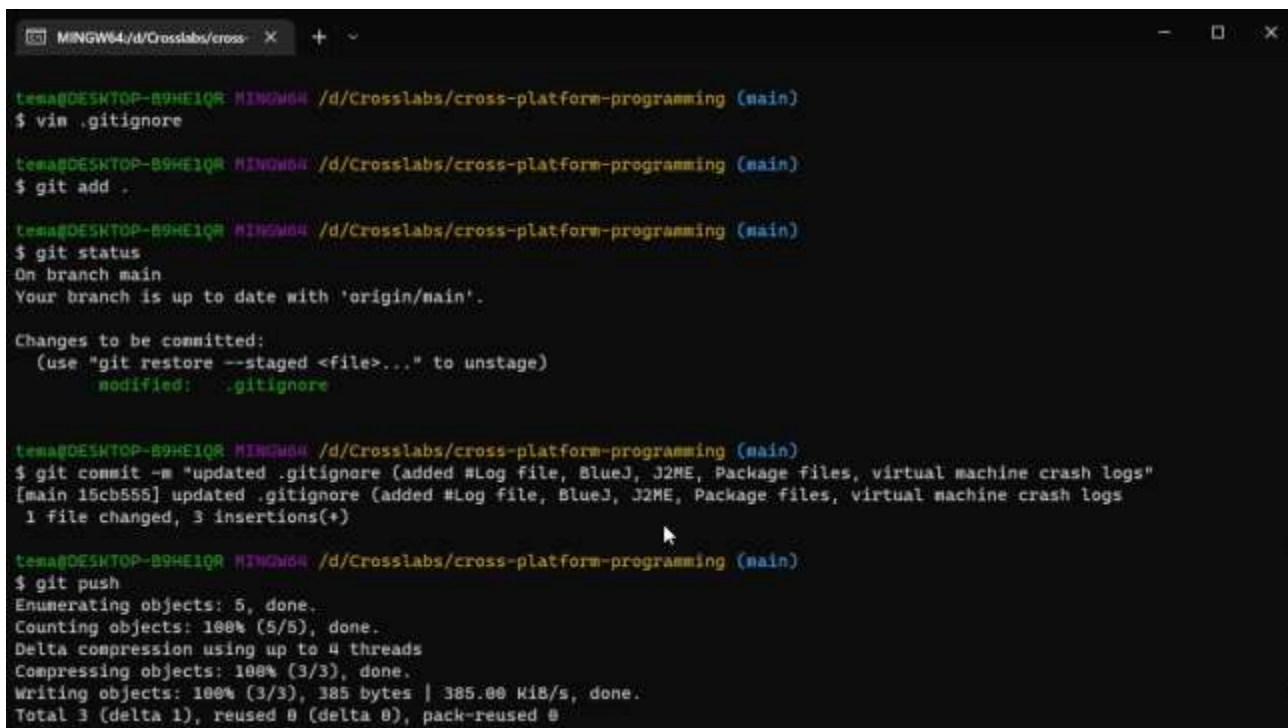
```
tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming (main)
$ |
```

Репозиторий успешно клонирован.

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



```
MINGW64/d/Crosslabs/cross- X + ~
tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming (main)
$ vim .gitignore

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

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.


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

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming (main)
$ git commit -m "updated .gitignore (added #Log file, BlueJ, J2ME, Package files, virtual machine crash logs"
[main 15cb555] updated .gitignore (added #Log file, BlueJ, J2ME, Package files, virtual machine crash logs
 1 file changed, 3 insertions(+)

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming (main)
$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 385 bytes | 385.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
```

Добавлены изменения в .gitignore.

4. Добавьте в файл README.md информацию о группе и ФИО студента, выполняющего Лабораторную работу.



```
add new strings
main
tema@DESKTOP-B9HE1QR committed 1 hour ago
Showing 1 changed file with 6 additions and 2 deletions.
README.md
1 - # cross-platform-programming
2 - Лабораторные работы по "Основы кроссплатформенного программирования"
3 + # cross-platform-programming
4 + Лабораторные работы по "Основы кроссплатформенного программирования"
5 + # 15cb555-22-1
6 + # 15cb555-22-1
7 + # 15cb555-22-1
8 + # 15cb555-22-1
9 + # 15cb555-22-1
10 + # 15cb555-22-1
11 + # 15cb555-22-1
12 + # 15cb555-22-1
13 + # 15cb555-22-1
14 + # 15cb555-22-1
15 + # 15cb555-22-1
16 + # 15cb555-22-1
17 + # 15cb555-22-1
18 + # 15cb555-22-1
19 + # 15cb555-22-1
20 + # 15cb555-22-1
21 + # 15cb555-22-1
22 + # 15cb555-22-1
23 + # 15cb555-22-1
24 + # 15cb555-22-1
25 + # 15cb555-22-1
26 + # 15cb555-22-1
27 + # 15cb555-22-1
28 + # 15cb555-22-1
29 + # 15cb555-22-1
30 + # 15cb555-22-1
31 + # 15cb555-22-1
32 + # 15cb555-22-1
33 + # 15cb555-22-1
34 + # 15cb555-22-1
35 + # 15cb555-22-1
36 + # 15cb555-22-1
37 + # 15cb555-22-1
38 + # 15cb555-22-1
39 + # 15cb555-22-1
40 + # 15cb555-22-1
41 + # 15cb555-22-1
42 + # 15cb555-22-1
43 + # 15cb555-22-1
44 + # 15cb555-22-1
45 + # 15cb555-22-1
46 + # 15cb555-22-1
47 + # 15cb555-22-1
48 + # 15cb555-22-1
49 + # 15cb555-22-1
50 + # 15cb555-22-1
51 + # 15cb555-22-1
52 + # 15cb555-22-1
53 + # 15cb555-22-1
54 + # 15cb555-22-1
55 + # 15cb555-22-1
56 + # 15cb555-22-1
57 + # 15cb555-22-1
58 + # 15cb555-22-1
59 + # 15cb555-22-1
60 + # 15cb555-22-1
61 + # 15cb555-22-1
62 + # 15cb555-22-1
63 + # 15cb555-22-1
64 + # 15cb555-22-1
65 + # 15cb555-22-1
66 + # 15cb555-22-1
67 + # 15cb555-22-1
68 + # 15cb555-22-1
69 + # 15cb555-22-1
70 + # 15cb555-22-1
71 + # 15cb555-22-1
72 + # 15cb555-22-1
73 + # 15cb555-22-1
74 + # 15cb555-22-1
75 + # 15cb555-22-1
76 + # 15cb555-22-1
77 + # 15cb555-22-1
78 + # 15cb555-22-1
79 + # 15cb555-22-1
80 + # 15cb555-22-1
81 + # 15cb555-22-1
82 + # 15cb555-22-1
83 + # 15cb555-22-1
84 + # 15cb555-22-1
85 + # 15cb555-22-1
86 + # 15cb555-22-1
87 + # 15cb555-22-1
88 + # 15cb555-22-1
89 + # 15cb555-22-1
90 + # 15cb555-22-1
91 + # 15cb555-22-1
92 + # 15cb555-22-1
93 + # 15cb555-22-1
94 + # 15cb555-22-1
95 + # 15cb555-22-1
96 + # 15cb555-22-1
97 + # 15cb555-22-1
98 + # 15cb555-22-1
99 + # 15cb555-22-1
100 + # 15cb555-22-1
```

Добавление ФИО студента, выполняющего Лабораторную работу.

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

```
commit 4cccce7fb3a9d305b0e98c4c718b95c78a389b2f6 (HEAD -> main, origin/main, origin/HEAD)
Author: Artem Bakanov <rtmbakanov@gmail.com>
Date: Mon Feb 27 16:07:46 2023 +0300

    added result output

commit d422ee40333e6cf03edea06c872f17f31997db9b
Author: Artem Bakanov <rtmbakanov@gmail.com>
Date: Mon Feb 27 16:05:13 2023 +0300

    added condition

commit 5d2ef5a45945b819fb37f667b406cf9b7aa11981
Author: Artem Bakanov <rtmbakanov@gmail.com>
Date: Mon Feb 27 16:02:31 2023 +0300

    added data input

commit 2d2a7c8c360e318ecc07fcc5074852309c808152
Author: Artem Bakanov <rtmbakanov@gmail.com>
Date: Mon Feb 27 16:01:11 2023 +0300

    installing Russian language

commit b9ff05d787120849b1886d72d7f984c1396e4cad
Author: Artem Bakanov <rtmbakanov@gmail.com>
Date: Mon Feb 27 15:59:21 2023 +0300

    Created second C++ file for the lab task/installing C++ libraries

commit b9be88a2913f5872838cc2ba77a045eff895c0fd
Author: Artem Bakanov <rtmbakanov@gmail.com>
Date: Mon Feb 27 15:53:56 2023 +0300
```

Коммиты программы на C++ (файл app2.cpp).

6. Добавьте файл README и зафиксируйте сделанные изменения.

```
tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming (main)
$ vim README.md

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

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

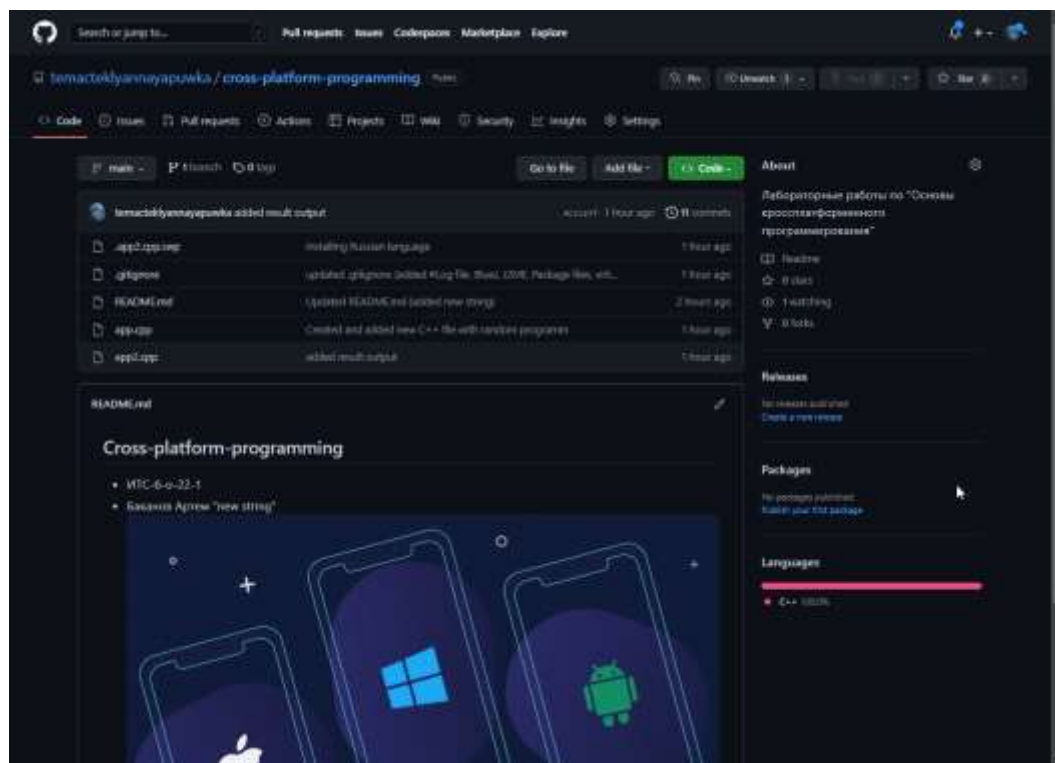
tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

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

tema@DESKTOP-B9HE1QR MINGW64 /d/Crosslabs/cross-platform-programming (main)
$ git commit -m "Updated README.md (added new string)"
[main 143abb7] Updated README.md (added new string)
1 file changed, 1 insertion(+), 1 deletion(-)
```

Обновление файла README.md.

**Итог лабораторной работы:**



Скриншот созданного репозитория.

## **Вопросы для защиты работы**

### **1. Что такое СКВ и каково ее назначение?**

Система, регистрирующая изменения в одном или нескольких файлах с тем, чтобы в дальнейшем была возможность вернуться к определённым старым версиям этих файлов.

### **2. В чем недостатки локальных и централизованных СКВ?**

Локальные: возможность потери данных вследствие возникновения физических поломок оборудования; отсутствие возможности совместной разработки.

Централизованные: отсутствие доступа к данным при сбое работы сервера; довольно низкая скорость работы (из-за возникновения сетевых задержек).

### **3. К какой СКВ относится Git?**

В Git каждая рабочая копия кода сама по себе является репозиторием.

#### **4. В чем концептуальное отличие Git от других СКВ?**

Бесплатный и open-source. Можно бесплатно скачать и вносить любые изменения в исходный код;

Небольшой и быстрый. Выполняет все операции локально, что увеличивает его скорость. Кроме того, Git локально сохраняет весь репозиторий в небольшой файл без потери качества данных;

Простое ветвление. В других системах контроля версий создание веток— утомительная и трудоёмкая задача, так как весь код копируется в новую ветку. В Git управление ветками реализовано гораздо проще и эффективнее.

#### **5. Как обеспечивается целостность хранимых данных в Git?**

Git обеспечивает целостность хранимых данных, используя контрольные суммы в качестве идентификаторов.

#### **6. В каких состояниях могут находиться файлы в Git? Как связаны эти состояния?**

Отслеживаемые файлы могут находиться в 3 состояниях: Не изменено (**Unmodified**), изменено (**Modified**), подготовленное (**Staged**).

#### **7. Что такое профиль пользователя в GitHub?**

У каждого пользователя есть публичный профиль, который помогает в поиске работы. Чтобы показать свой опыт потенциальному работодателю, нужно оставить в резюме ссылку на профиль. Когда рекрутер или другой специалист перейдут по ней, то увидят информацию о вас.

#### **8. Какие бывают репозитории в GitHub?**

Репозиторий Git бывает локальный и удалённый.

#### **9. Укажите основные этапы модели работы с GitHub.**

Установка Git; добавление имени, фамилии и адреса электронной почты; ввод определенных команд для Git; загрузка изменений в состояние (staged); добавление коммита; отправка в репозиторий на сервис GitHub.



## **10. Как осуществляется первоначальная настройка Git после установки?**

Добавление имени, фамилии и адреса электронной почты:

```
git config --global user.name <YOUR_NAME> - указывает ваше имя, фамилию.  
git config --global user.email <EMAIL> - указывает вашу электронную почту.  
git init - создает новый репозиторий Git.
```

## **11. Опишите этапы создания репозитория в GitHub.**

Ввод имени для репозитория, добавление описания проекта (выборочно), выбор приватности данного репозитория, добавление дополнительных файлов, как README.md и .gitignore.

## **12. Какие типы лицензий поддерживаются GitHub при создании репозитория?**

Academic Free License v3.0; Boost Software License 1.0; Creative Commons license family; Eclipse Public License 1.0; ISC; MIT и многие другие.

## **13. Как осуществляется клонирование репозитория GitHub? Зачем нужно клонировать репозиторий?**

С помощью команд `git clone`/`git push`. Чтобы упростить устранение конфликтов слияния, добавление или удаление файлов и отправку больших фиксаций.

## **14. Как проверить состояние локального репозитория Git?**

Используйте команду `git status`, чтобы проверить текущее состояние репозитория.

**15. Как изменяется состояние локального репозитория Git после выполнения следующих операций: добавления/изменения файла в локальный репозиторий Git; добавления нового/ измененного файла под версионный контроль с помощью команды `git add`; фиксации (коммита) изменений с помощью команды `git commit` и отправки изменений на сервер с помощью команды `git push`?**

Репозиторий будет находиться в начальном состоянии, то есть Git напишет о том, что вы находитесь на основной ветке и то что вам нечего коммитить, рабочая ветка пуста.

**16. У Вас имеется репозиторий на GitHub и два рабочих компьютера, с помощью которых Вы можете осуществлять работу над некоторым проектом с использованием этого репозитория. Опишите последовательность команд, с помощью которых оба локальных репозитория, связанных с репозиторием GitHub будут находиться в синхронизированном состоянии. Примечание: описание необходимо начать с команды `git clone`.**

`git clone; git remote -v; git pull;`

**16. GitHub является не единственным сервисом, работающим с Git. Какие сервисы еще Вам известны? Приведите сравнительный анализ одного из таких сервисов с GitHub.**

Bitbucket - веб-сервис для хостинга проектов и их совместной разработки, основанный на системах контроля версий Mercurial и Git. GitHub вращается вокруг общедоступного кода, тогда как Bitbucket предназначен в основном для частных проектов. Это основное различие между GitHub и Bitbucket. GitHub — ведущее сообщество разработчиков открытого исходного кода, тогда как Bitbucket в основном используется предприятиями и предприятиями.

**17. Интерфейс командной строки является не единственным и далеко не самым удобным способом работы с Git. Какие Вам известны программные средства с графическим интерфейсом пользователя для работы с Git? Приведите как реализуются описанные в лабораторной работе операции Git с помощью одного из таких программных средств.**

Можно работать через терминал windows.

