Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии
Департамент цифровых, робототехнических систем и электроники

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №2 дисциплины «Основы кроссплатформенного программирования

Выполнил:
Пугачев Кирилл Дмитриевич курс, группа ИТС-б-о-23-1, 11.03.02 «Инфокоммуникационные технологии и системы связи», очная форма обучения

_____
(подпись)

Проверил:
Воронкин Р. А.

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Тема: Исследование основных возможностей Git и GitHub

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

Ссылка на репозиторий: https://github.com/chillkirill/laba2

Порядок выполнения работы:

1. Использование команды git log на репозитории

```
C:\Users\user\laba2>git log
commit c0d1971913501a90ac0daa22c84c60608873a350 (HEAD -> main, origin/main, origin/HEAD)
Author: chillkirill <quwlast@gmail.com>
Date:   Thu Dec 26 17:19:42 2024 +0300

    Создал копию программы и внес некоторые изменения

commit 91f9e4f97b188f937d264025f4237dcb456f6a23
Author: chillkirill <quwlast@gmail.com>
Date:   Thu Dec 26 17:15:53 2024 +0300

    Создал и добавил программу на языке Python

commit f6537c90fe43555e1f6cd01ae889cd0ac0ef4e9e
Author: chillkirill <quwlast@gmail.com>
Date:   Thu Dec 26 17:14:46 2024 +0300

    Внес изменения в своей программе

commit f1ae0be4e5afeb2cb47e04605452c374c783cd5b
Author: chillkirill <quwlast@gmail.com>
Date:   Thu Dec 26 17:11:28 2024 +0300

    Написал программу на языке Python

commit 9c07aaba3a30bc9d2492cff548314c7074eb1260
Author: chillkirill <quwlast@gmail.com>
Date:   Thu Dec 26 17:02:47 2024 +0300
```

Рис.1 Результат выполнения команды git log

2. Использование команды git log с аргументом –p и -2

```
C:\Users\user>cd laba2

C:\Users\user\laba2>git log -p -2
commit c0d1971913501a90ac0daa22c84c60608873a350 (HEAD -> main, origin/main, origin/HEAD)
Author: chillkirill <quwlast@gmail.com>
Date:   Thu Dec 26 17:19:42 2024 +0300

    Создал копию программы и внес некоторые изменения

diff --git a/python2.py.txt b/python2.py.txt
new file mode 100644
index 0000000..413a4a2
--- /dev/null
+++ b/python2.py.txt
@@ -0,0 +1,9 @@
+def is_palindrome(s):
+    s = ''.join(filter(str.isalnum, s)).lower()
+    return s == s[::-1]
+
+def factorial(n):
+  if n == 0:
+    return 2
+  else:
+    return n * factorial(n-1)

commit 91f9e4f97b188f937d264025f4237dcb456f6a23
Author: chillkirill <quwlast@gmail.com>
Date:   Thu Dec 26 17:15:53 2024 +0300
```

Рис. 2 Аргумент –p -2 команды git log

3. Использование опции --stat команды git log

```
C:\Users\user\laba2>git log --stat
commit c0d1971913501a90ac0daa22c84c60608873a350 (HEAD -> main, origin/main, origin/HEAD)
Author: chillkirill <quwlast@gmail.com>
Date:   Thu Dec 26 17:19:42 2024 +0300

    Создал копию программы и внес некоторые изменения

 python2.py.txt | 9 +++++++++
 1 file changed, 9 insertions(+)

commit 91f9e4f97b188f937d264025f4237dcb456f6a23
Author: chillkirill <quwlast@gmail.com>
Date:   Thu Dec 26 17:15:53 2024 +0300

    Создал и добавил программу на языке Python

 python.py.txt | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)

commit f6537c90fe43555e1f6cd01ae889cd0ac0ef4e9e
Author: chillkirill <quwlast@gmail.com>
Date:   Thu Dec 26 17:14:46 2024 +0300

    Внес изменения в своей программе

 python.py.txt | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
```

Рис. 3 Опция git log --stat

4. Изменение формата вывода с помощью опции -pretty=oneline

```
C:\Users\user\laba2>git log --pretty=oneline
c0d1971913501a90ac0daa22c84c60608873a350 (HEAD -> main, origin/main, origin/HEAD) Создал копию программы и внес некоторы
е изменения
91f9e4f97b188f937d264025f4237dcb456f6a23 Создал и добавил программу на языке Python
f6537c90fe43555e1f6cd01ae889cd0ac0ef4e9e Внес изменения в своей программе
f1ae0be4e5afeb2cb47e04605452c374c783cd5b Написал программу на языке Python
9c07aaba3a30bc9d2492cff548314c7074eb1260 Добавил новый файл для будущей программы
641db387743a45aa8090455c42ac66f8b843524f Изменил файл README
0ee4a973c836b732279da5565d387be5f1340294 Initial commit

C:\Users\user\laba2>
```

Рис. 4 Вывод с опцией --pretty=oneline

5. Использование форматного вывода опции --pretty

```
C:\Users\user\laba2>git log --pretty=format:"%h - %an, %ar : %s"
c0d1971 - chillkirill, 27 minutes ago : Создал копию программы и внес некоторые изменения
91f9e4f - chillkirill, 30 minutes ago : Создал и добавил программу на языке Python
f6537c9 - chillkirill, 31 minutes ago : Внес изменения в своей программе
f1ae0be - chillkirill, 35 minutes ago : Написал программу на языке Python
9c07aab - chillkirill, 43 minutes ago : Добавил новый файл для будущей программы
641db38 - chillkirill, 47 minutes ago : Изменил файл README
0ee4a97 - Kirill, 64 minutes ago : Initial commit

C:\Users\user\laba2>
```

Рис. 5 Форматный вывод

6. Использование опции -pretty с опцией --graph

```
C:\Users\user\laba2>git log --pretty=format:"%h %s" --graph
* c0d1971 Создал копию программы и внес некоторые изменения
* 91f9e4f Создал и добавил программу на языке Python
* f6537c9 Внес изменения в своей программе
* f1ae0be Написал программу на языке Python
* 9c07aab Добавил новый файл для будущей программы
* 641db38 Изменил файл README
* 0ee4a97 Initial commit
```

Рис. 6 Вывод с опцией --graph

7. Создание нового репозитория и внесение 7 коммитов

```
C:\Users\user\laba2>git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

C:\Users\user\laba2>
```

Рис. 7 Внесение коммитов

8. Создание тега

```
C:\Users\user\laba2>git tag -a v1.0 -m "Версия 1.0"
```

Рис. 8 Создание тега

9. Просмотр журнала хранилища

```
C:\Users\user\laba2>git log --graph --pretty=oneline --abbrev-commit
* c0d1971 (HEAD -> main, tag: v1.0, origin/main, origin/HEAD) Создал копию программы и внес некоторые изменения
* 91f9e4f Создал и добавил программу на языке Python
* f6537c9 Внес изменения в своей программе
* f1ae0be Написал программу на языке Python
* 9c07aab Добавил новый файл для будущей программы
* 641db38 Изменил файл README
* 0ee4a97 Initial commit
```

Рис. 9 Журнал хранилища

10. Просмотр содержимого последнего коммита

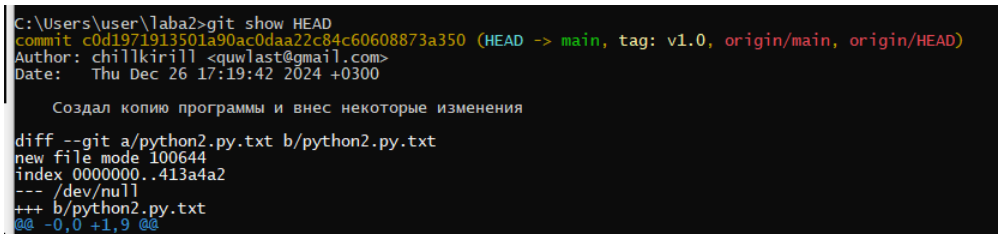

Рис. 10 Содержимое последнего коммита

11. Просмотр содержимого предпоследнего коммита

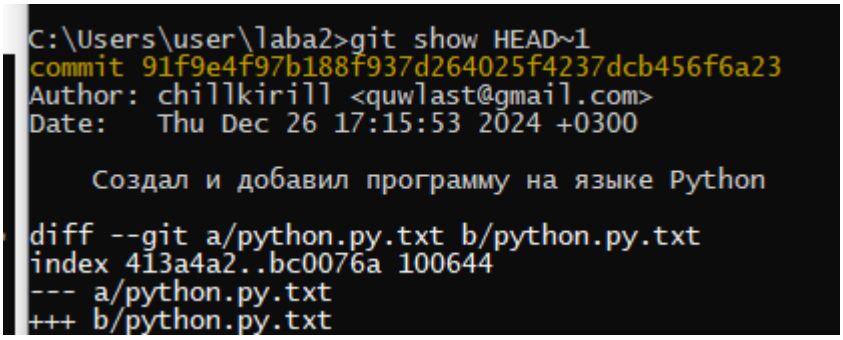

Рисунок 11. Содержимое предпоследнего коммита

12. Просмотр содержимого коммита с указанным хэшем

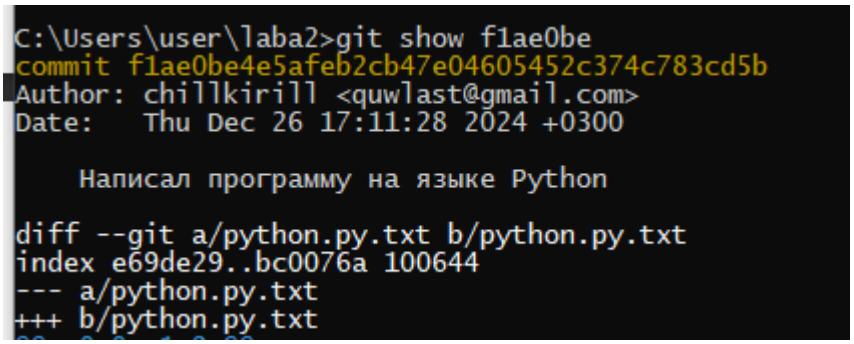

Рисунок 12. Содержимое коммита ec17348

13. Удаление кода программы и возврат изменений командой git checkout

```
C:\Users\user\laba2>git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        deleted:    python.py.txt

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

C:\Users\user\laba2>git checkout -- python.py.txt

C:\Users\user\laba2>
```

Рисунок 13. Возврат изменений

14. Создание коммита и откат к предыдущей версии командой git reset

```
C:\Users\user\laba2>git commit -m "удаление файла"
[main c770d52] удаление файла
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 sqwrt.txt

C:\Users\user\laba2>git reset --hard HEAD~1
HEAD is now at c0d1971 Создал копию программы и внес некоторые изменения
```

Рисунок 14. Откат коммита

Контрольные вопросы:

1. Что такое СКВ и каково ее назначение?

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.

2. В чем недостатки локальных и централизованных СКВ?

Локальные СКВ (системы контроля версий) менее удобны для совместной работы, поскольку требуют постоянной синхронизации изменений между участниками проекта. Централизованные СКВ, в свою очередь, могут быть уязвимы к отказам сервера, а также к потере данных, если главный репозиторий поврежден.

3. К какой СКВ относится Git?

Git относится к децентрализованным системам контроля версий.

4. В чем концептуальное отличие Git от других СКВ?

Git отличается от других СКВ тем, что он децентрализован: каждый разработчик имеет полную копию репозитория, а не только доступ к нему. Это позволяет работать автономно, делать коммиты локально и синхронизировать изменения с другими разработчиками.

5. Как обеспечивается целостность хранимых данных в Git?

Git использует криптографическую хеш-функцию SHA-1 для вычисления уникального хэша для каждого файла и коммита. Изменения в файле изменяют его хэш, что позволяет Git отслеживать изменения и обеспечивать целостность данных.

6. В каких состояниях могут находится файлы в Git? Как связаны эти состояния?

Файлы в Git могут находиться в 3 состояниях:

– Untracked: файл не отслеживается Git, т.е. не включен в репозиторий.

– Staged: файл был изменен и подготовлен к коммиту, т.е. включен в следующий коммит.

– Committed: файл был зафиксирован в истории репозитория.

– Связь между состояниями:

1. Untracked -> Staged: файл переходит в состояние Staged после команды git add.

2. Staged -> Committed: файл переходит в состояние Committed после команды git commit.

7. Что такое профиль пользователя в GitHub?

Профиль пользователя на GitHub — это публичная страница, которая представляет вас как разработчика. На ней отображаются ваши репозитории, активности, подписки, а также информация о вас: имя, местоположение, сайт, био, аватар.

8. Какие бывают репозитории в GitHub?

– Публичные: доступны всем пользователям GitHub. Их код можно просматривать, скачивать и использовать бесплатно.

– Приватные: доступны только владельцу репозитория и приглашенным сотрудникам. Для приватных репозиториев нужна платная подписка GitHub.

– Также существуют fork-репозитории: это копии публичных репозиториев, которые можно модифицировать и использовать для собственных проектов. Fork-репозитории могут быть как публичными, так и приватными.

9. Укажите основные этапы модели работы с GitHub.

Основные этапы работы с GitHub:

1. Создание репозитория: Создается новый репозиторий на GitHub, где будет храниться код проекта.

2. Клонирование репозитория: Локальная копия репозитория скачивается на компьютер с помощью команды git clone.

3. Работа с кодом: В локальной копии вносятся изменения в код проекта.

4. Добавление изменений в индекс: Измененные файлы добавляются в индекс с помощью команды git add.

5. Коммит: Изменения в индексе сохраняются в локальной истории с помощью команды git commit.

6. Пуш: Изменения из локального репозитория отправляются на сервер GitHub с помощью команды git push.

7. Pull: Изменения из репозитория GitHub скачиваются в локальную копию с помощью команды git pull.

10.Как осуществляется первоначальная настройка Git после установки?

git config - это команда, которая позволяет вам настроить Git под свой стиль работы и предпочтения. Она используется для установки глобальных, локальных и системных параметров Git.

Глобальные параметры: применяются ко всем репозиториям на вашем компьютере. Их установка осуществляется с флагом --global.

Локальные параметры: применяются только к текущему репозиторию. Их установка осуществляется без флага --global.

Системные параметры: применяются ко всем пользователям системы. Их установка осуществляется с флагом --system.

11.Опишите этапы создания репозитория в GitHub.

1. Авторизоваться на сайте.
2. Нажать кнопку "New" и ввести название, описание и тип доступа для нового репозитория.
3. Нажать кнопку "Create repository".

12. Какие типы лицензий поддерживаются GitHub при создании репозитория?

– MIT: Позволяет свободное использование, модификацию и распространение, включая коммерческие цели.

– Apache 2.0: Позволяет свободное использование, модификацию и распространение, включая коммерческие цели, с условием сохранения исходной лицензии.

– GPL 3.0: Позволяет свободное использование, модификацию и распространение, включая коммерческие цели, но с обязательством публикации исходного кода под той же лицензией.

– BSD 3-Clause: Позволяет свободное использование, модификацию и распространение, включая коммерческие цели, с условием сохранения имени автора и отказа от ответственности за использование программного обеспечения.

– Public Domain: Отказывается от всех прав на программное обеспечение, позволяя свободное использование, модификацию и распространение, без каких-либо ограничений.

– Также GitHub предоставляет возможность выбрать "None" (без лицензии), что означает, что вы не предоставляете никаких прав на использование вашего кода.

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

Клонирование репозитория GitHub осуществляется с помощью команды git clone: git clone <URL_репозитория>

Клонирование репозитория создает локальную копию репозитория на вашем компьютере, что позволяет вам:

Работать с кодом локально: Вносить изменения, создавать ветки, использовать Git без подключения к Интернету.

Вносить вклад в проект: Вносить изменения в код проекта и отправлять их на GitHub.

Сохранять историю изменений: Локальная копия сохраняет полную историю всех изменений в репозитории.

Совместная работа: Клонирование позволяет нескольким разработчикам работать над проектом одновременно и синхронизировать свои изменения.

14. Как проверить состояние локального репозитория Git?

Чтобы проверить состояние локального репозитория Git, используется команда git status.

Она покажет:

- Измененные файлы: Файлы, которые были изменены, но еще не добавлены в индекс.
- Файлы, добавленные в индекс: Файлы, которые были изменены и добавлены в индекс для следующего коммита.
- Неотслеживаемые файлы: Файлы, которые не отслеживаются Git (не входят в репозиторий).
- Текущая ветка: Ветка, в которой вы находитесь.
- Состояние ветки: Информирует о том, соответствует ли локальная копия ветки удаленной копии на GitHub.

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 ?

После добавления или изменения файла в локальном репозитории Git он переходит в состояние "Untracked" (не отслеживается). Git еще не знает об этом файле и не включает его в историю изменений.

Команда git add <имя_файла> добавляет файл в индекс Git. Файл переходит в состояние "Staged" (подготовлен к коммиту). Теперь Git знает об этом файле и будет включать его в следующий коммит.

Команда git commit -m "Сообщение о коммите" создает новый коммит в локальной истории Git. Файл переходит в состояние "Committed" (зафиксирован). Изменения, которые были в индексе, теперь стали частью истории репозитория.

Команда git push origin main отправляет изменения из локальной истории на сервер GitHub. Локальные коммиты теперь доступны в удаленном репозитории.

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.

1. Клонировать репозиторий на оба компьютера:

git clone <URL_репозитория>.

2. Работать над проектом на первом компьютере, фиксировать изменения (git add, git commit).

3. Отправить изменения на GitHub: git push origin main.

4. Обновить локальный репозиторий на втором компьютере: git pull origin main.

5. Повторить шаги 2-4 для второго компьютера.

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

Bitbucket: Разработан Atlassian, известен своей интеграцией с другими продуктами Atlassian, такими как Jira и Confluence. Предлагает как бесплатные, так и платные планы с поддержкой приватных репозиториев.

- GitLab: Открытый сервис с возможностью развертывания на собственных серверах. Предлагает множество функций для управления кодом, CI/CD, и безопасности.
- Azure DevOps: Облачный сервис от Microsoft, включающий в себя систему контроля версий Git, CI/CD, управление проектами и другие инструменты.

GitHub и Bitbucket - популярные платформы для хостинга Gitрепозиториев, но у них есть некоторые различия. GitHub предлагает бесплатные публичные репозитории и платные приватные, в то время как Bitbucket предоставляет бесплатные приватные репозитории для небольших команд и платные планы с расширенными функциями.

GitHub интегрируется с различными сервисами, такими как Travis CI и CircleCI, в то время как Bitbucket отлично интегрируется с продуктами Atlassian, такими как Jira и Confluence. GitHub предлагает более широкий набор функций для управления кодом, сотрудничества и развертывания, в то время как Bitbucket предлагает набор функций, достаточный для большинства проектов. Сообщество разработчиков на GitHub значительно больше, чем на Bitbucket, но Bitbucket обеспечивает хорошую поддержку
Atlassian.

18. 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.

GitHub Desktop: Разработан самой компанией GitHub, интегрирован с веб-платформой GitHub и предлагает простой и интуитивно понятный интерфейс.

- GitKraken: Известен своим стильным дизайном, большим количеством функций и поддержкой многих систем контроля версий, включая Git, Mercurial и SVN.
- Sourcetree: Разработан Atlassian, производителем Jira и Bitbucket, имеет хорошую интеграцию с этими сервисами и предлагает широкий набор функций для управления репозиториями.
- TortoiseGit: Бесплатная программа с открытым исходным кодом для Windows, известна своей интеграцией в контекстное меню Проводника Windows и простой использованием.

Как реализуются операции Git в GitHub Desktop

В GitHub Desktop необходимо нажать кнопку "Clone a repository" и ввести URL репозитория. После чего выбрать папку для клонирования.

Вывод: в ходе лабораторной работы были исследованы базовые возможности системы контроля версий Git и веб-сервиса для хостинга ITпроектов GitHub.