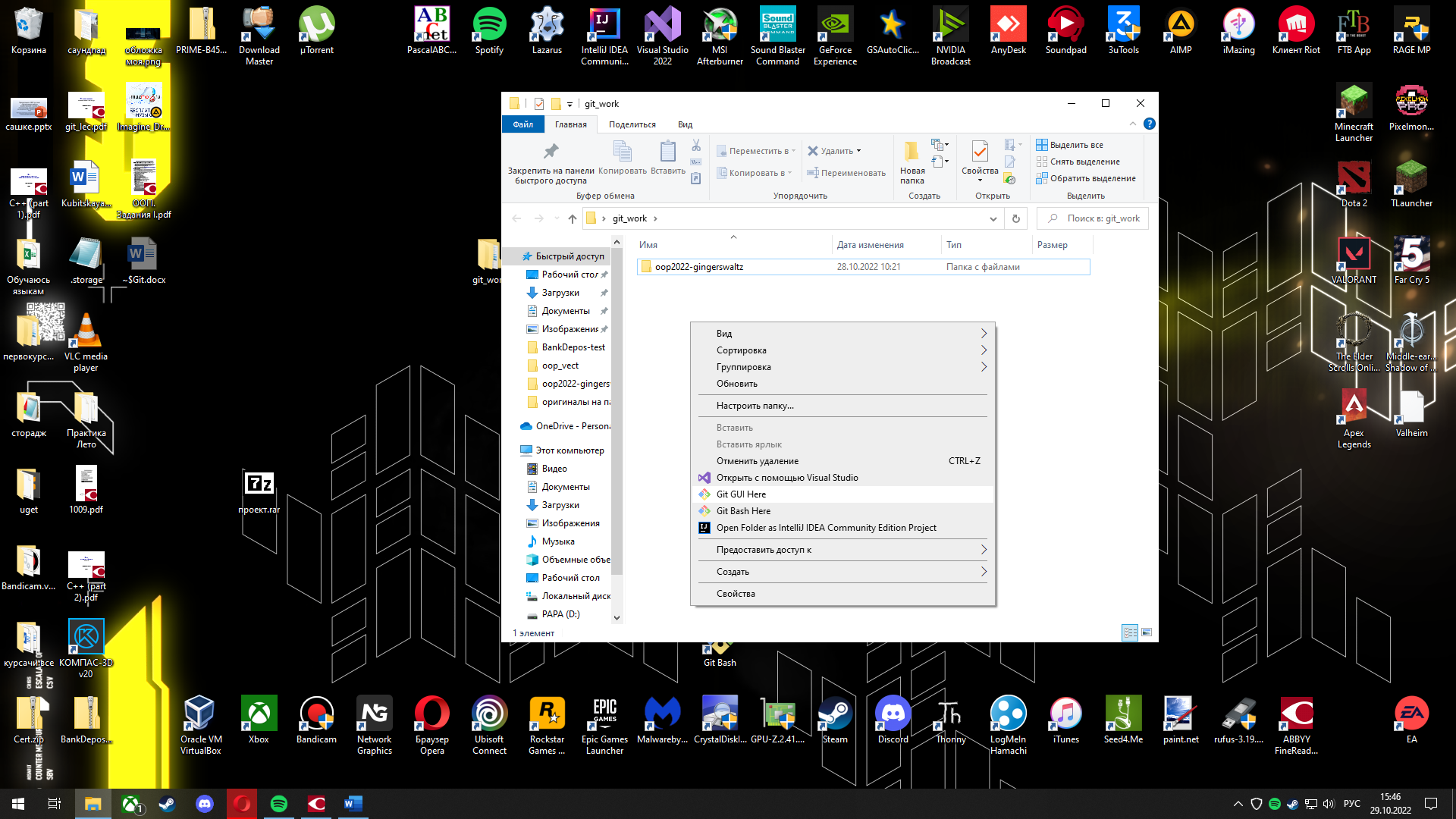
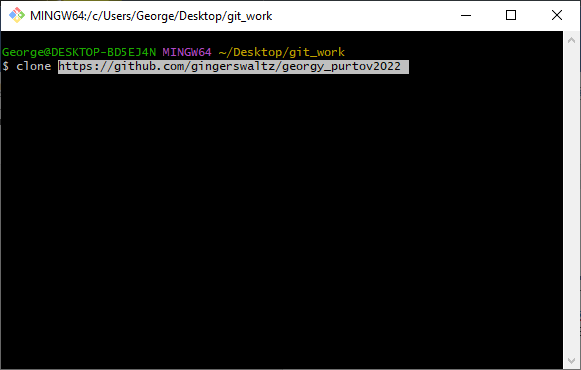
# Работа с Git Bash

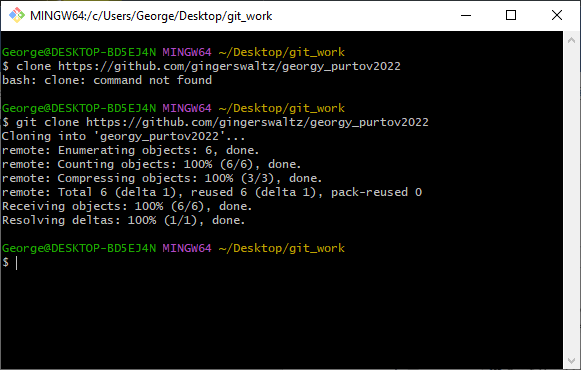
## Клонирование существующего репозитория

Скопировать ссылку на существующий удаленный репозиторий  
 ( <https://github.com/gingerswaltz/georgy_purtov2022> ) , открыть в предварительно созданной папке Git Bash консоль

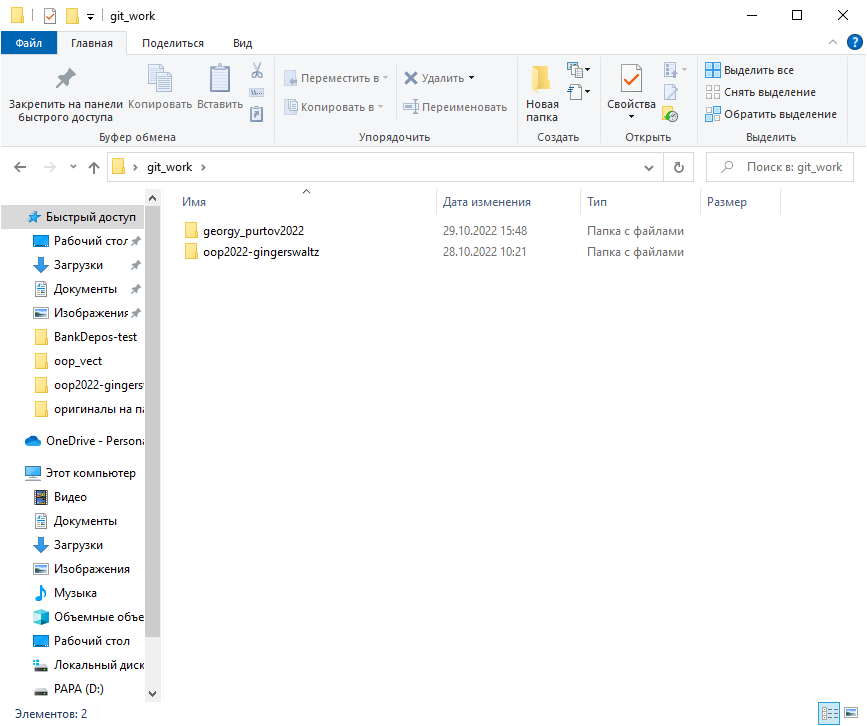


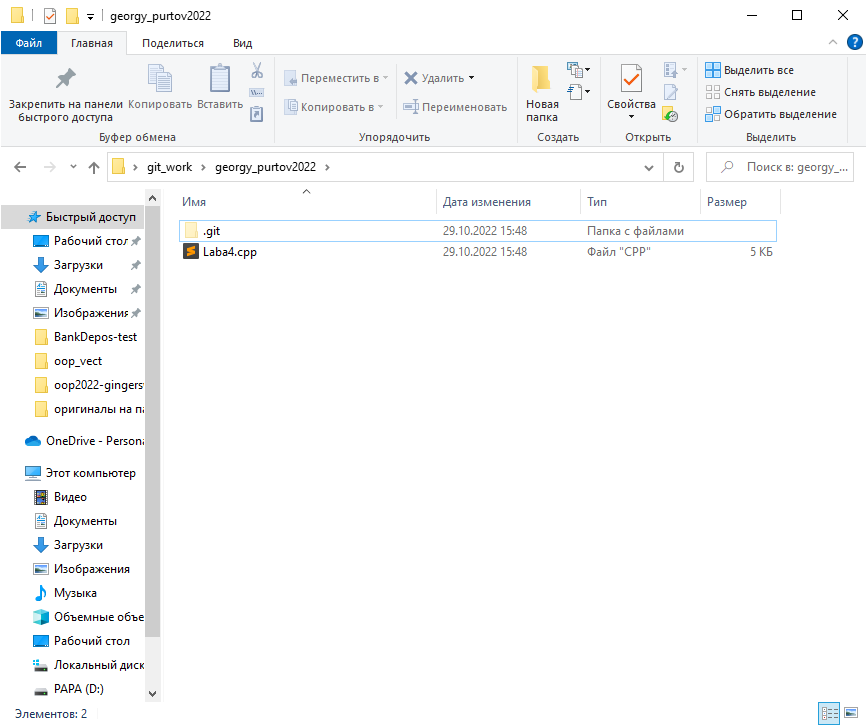
Написать команду **git clone «ссылка на удаленный репозиторий»,** нажать Enter





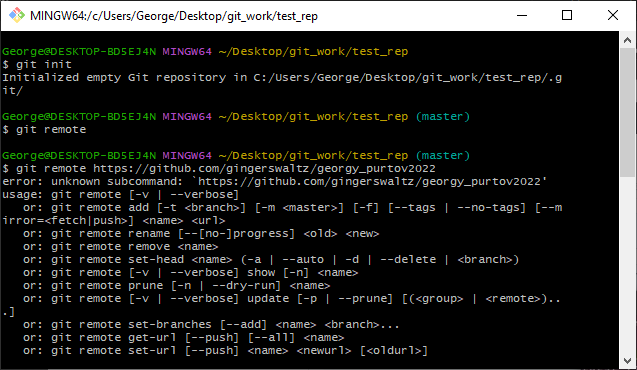
Результат на скринах ниже.



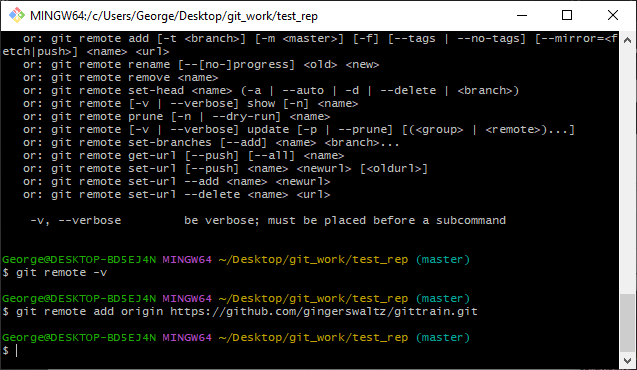


Инициализация репозитория произведена (я уже пользовался им, поэтому загрузился cpp файлик)

## Через git init

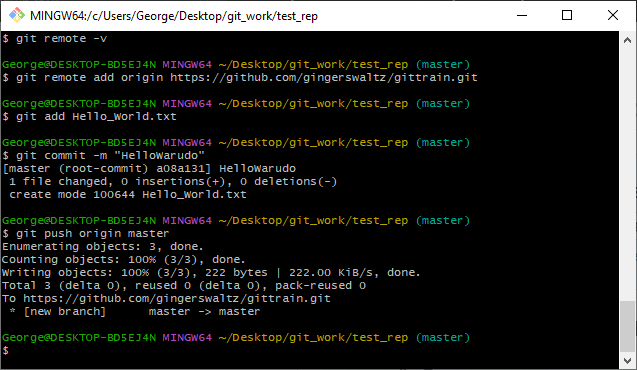
Создать папку, в ней открыть git bash, прописать команду **git init**

Затем написать команду **git remote -add «алиас» «ссылка на ваш репозиторий»**



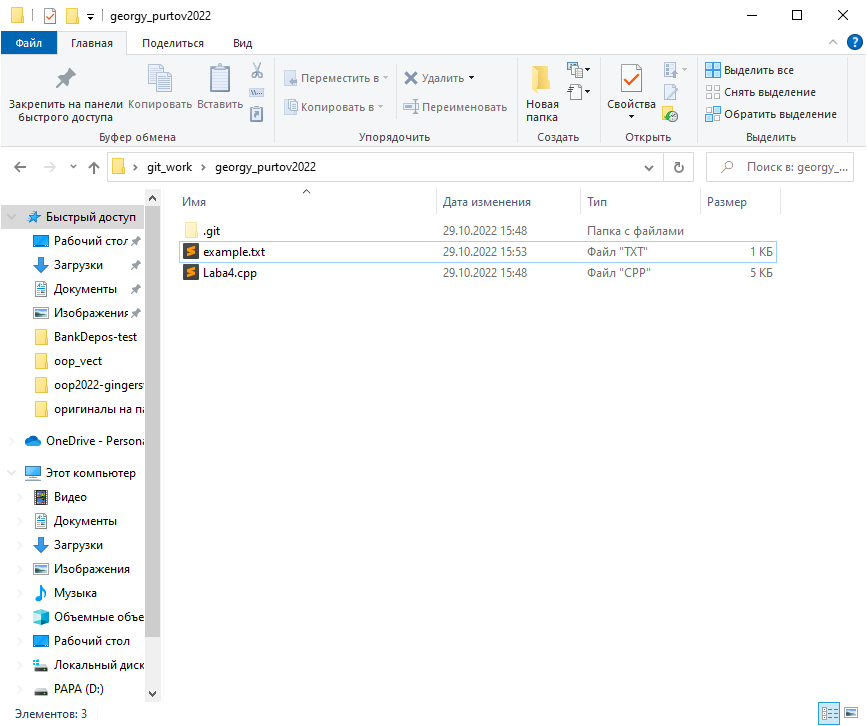
Затем можно работать с удаленным репозиторием через команду

**git push «нужная ветка в УДАЛЕННОМ репозитории» «нужная ветка в ЛОКАЛЬНОМ текущем репозитории»**



## Добавление файлов к отслеживанию

Создадим в вышеуказанной папке файл example.txt и добавим в него содержимое text 1;

Откроем git bash в вышеуказанной папке.

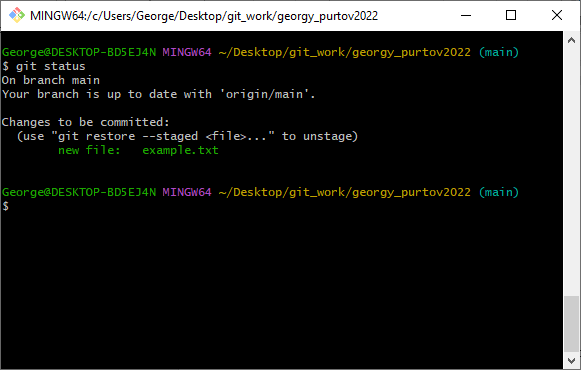
В открытом git bash пропишем команду **git add “ваш файл(-ы)”** и нажмем Enter



Консоль никак не отреагирует.

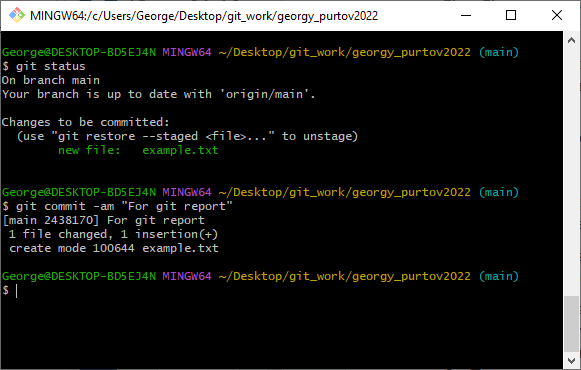
## Просмотр состояния репозитория

В текущем окне git bash введите команду git status



## Сделать коммит (зафиксировать изменения).

Ввести **команду git commit -am “сообщение об изменении**” и Enter

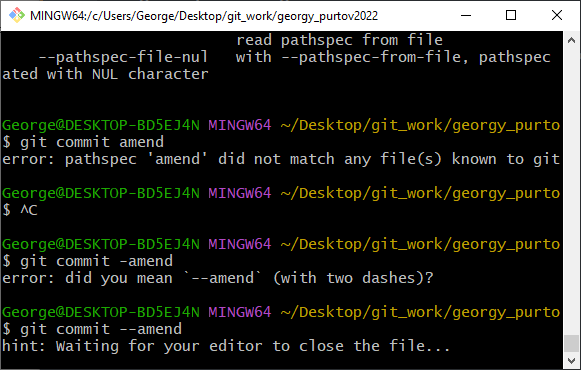


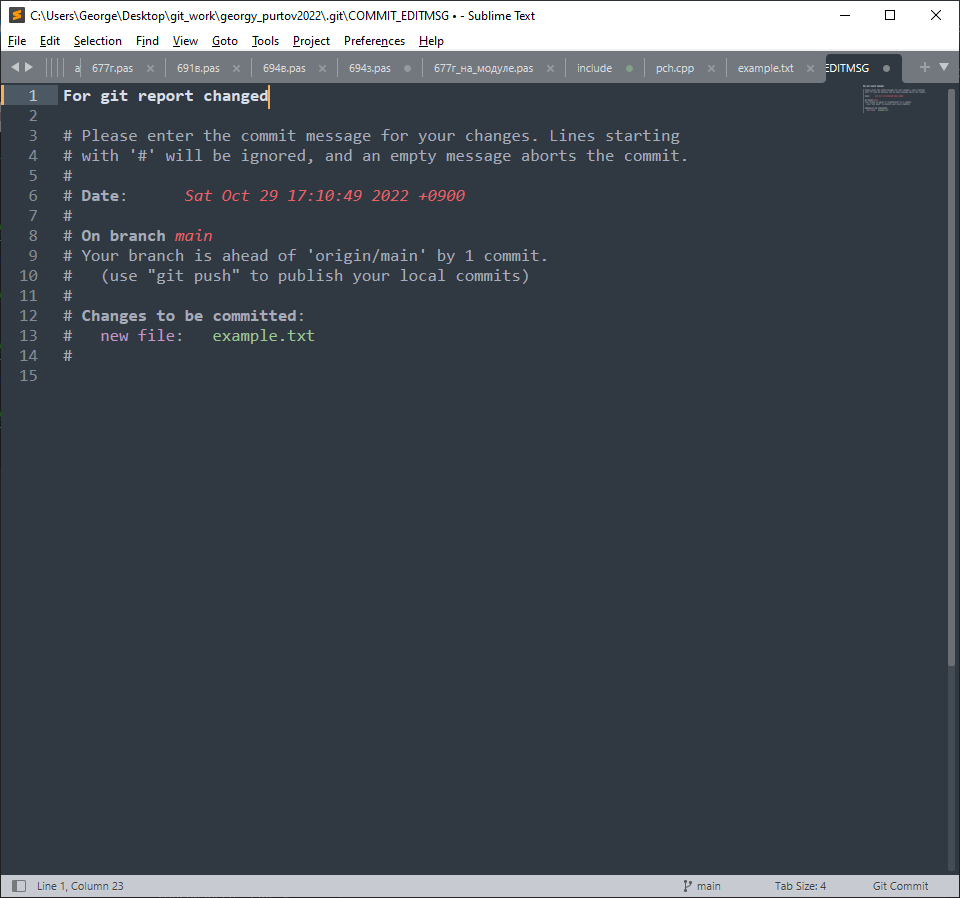
## Исправить сообщение предыдущего коммита.

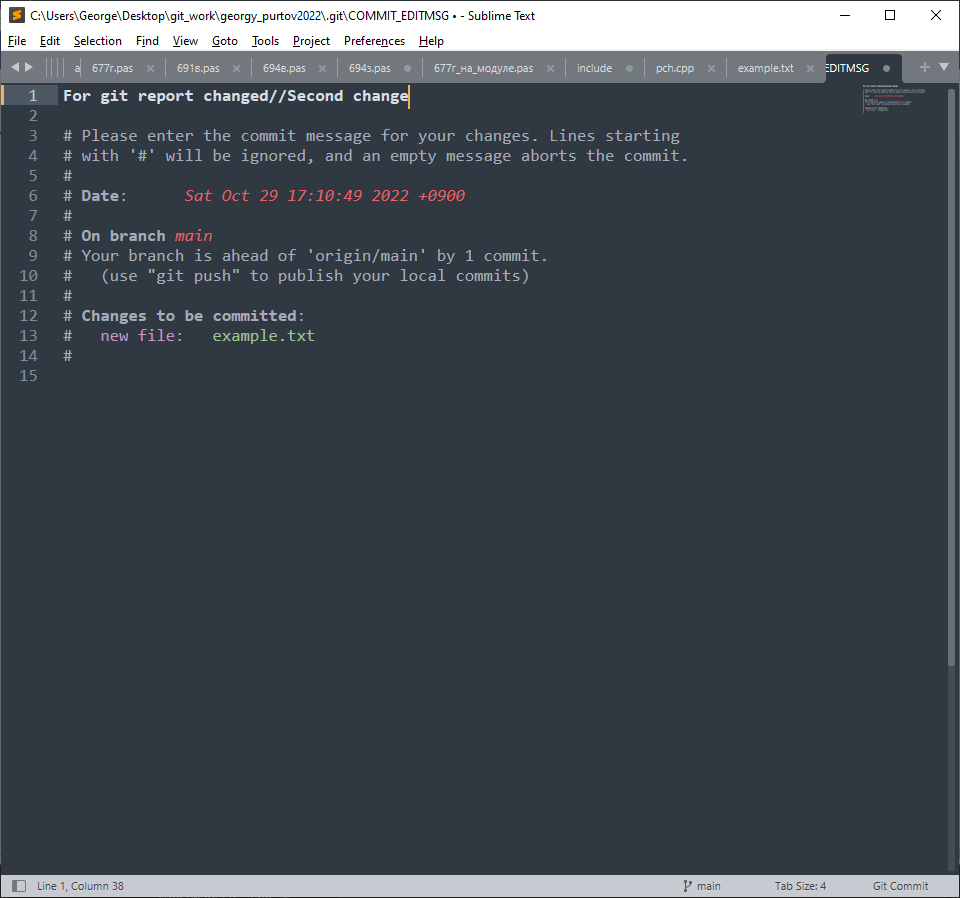
Если вы хотите изменить только сообщение вашего последнего коммита, это очень просто:

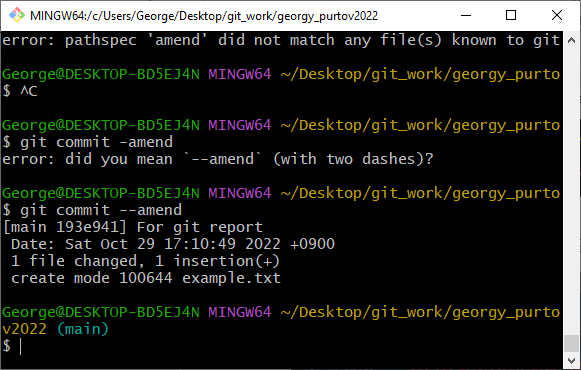
**git commit --amend**

Откроется текстовый редактор, в нем изменяем сообщение и закрываем его









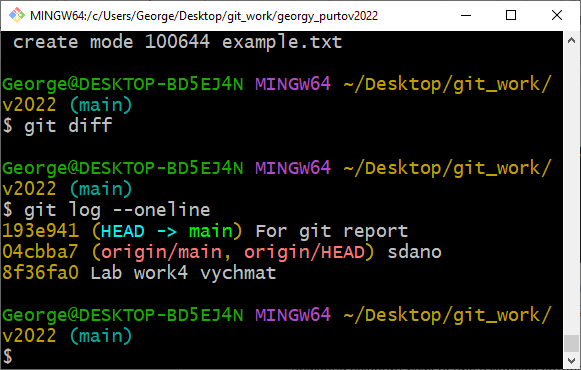
## Сделать коммит. Просмотреть разницу между коммитами.

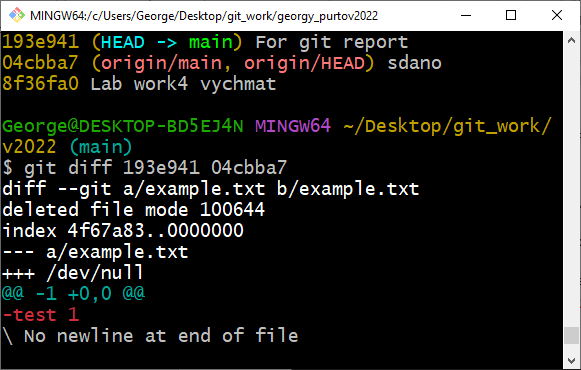
Коммит сделан на изображении выше; Посмотрим разницу между коммитами 1 и 2

**git diff**

Команда выводит изменения в файлах, которые еще не были добавлены в индекс. Сравнение происходит с последним коммитом.

Сравнение коммитов  
Команда git diff позволяет сравнивать два различных коммита. Сначала нужно определить хеш (ID) коммитов, которые требуется сравнивать. Можно воспользоваться командой git log, чтобы вывести список коммитов и их идентификаторы:  
**git log –oneline**Теперь сравним два коммита. Для этого в качестве первого аргумента команде git diff указывается хеш первого коммита, а вторым аргументом хеш второго коммита, например:  
**git diff 4612297 5e356cf**



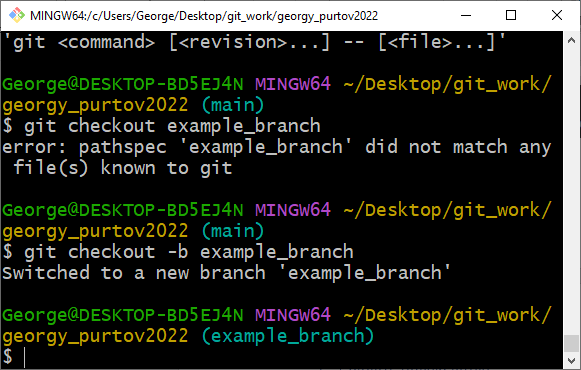


## Создать новую ветку + 8. Переключиться на неё

Создать рабочую ветку и переключиться

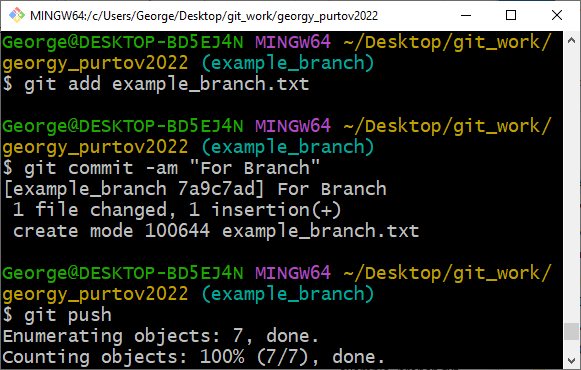
**git branch “название ветки”** затем **git checkout “имя ветки”**

**или git checkout -b «название ветки»** (создать и сразу переключиться)



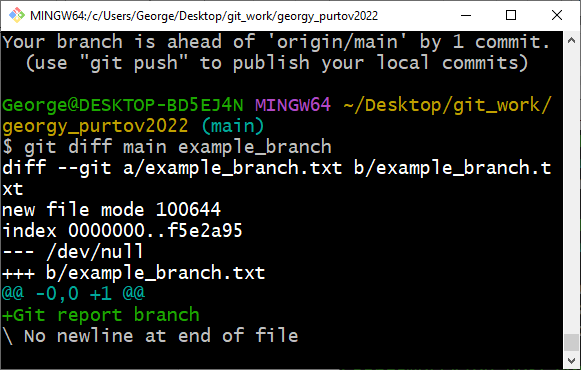
## Внести изменения. Посмотреть разницу (diff) между ветками+ 10. Зафиксировать изменения.

Вносим изменения в ветку согласно вышенаписанным пунктам , используем новый файл example\_branch.txt



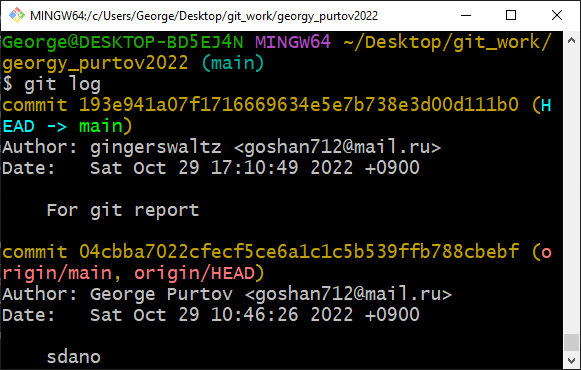
Чтобы сравнить конкретные файлы между двумя ветками используется команда:

**git diff branch1 branch2 ./myfile.cpp**

Вместо branch1, branch2 нужно указать название веток, а вместо myfile.cpp путь до сравниваемого файла. Через пробел можно добавить еще файлы для сравнения. 

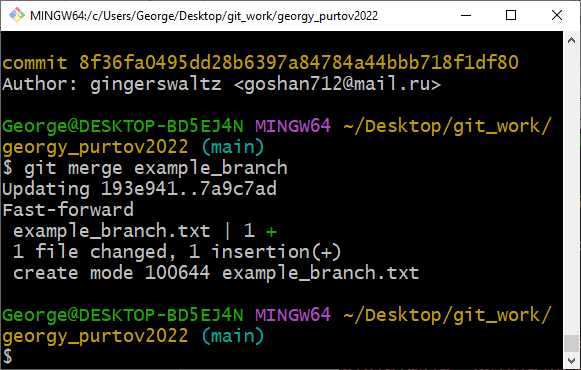
## 11. Посмотреть что содержится в файлах на ветке master (main).

**Команда git log**



## 12. Объединить ветки.

Команда **git merge “ветка которую обьединим с основной»**



## 14. Отправить изменения в удалённый репозитарий, забрать изменения из удалённого репозитария.

Описано в пункте 1 и 2

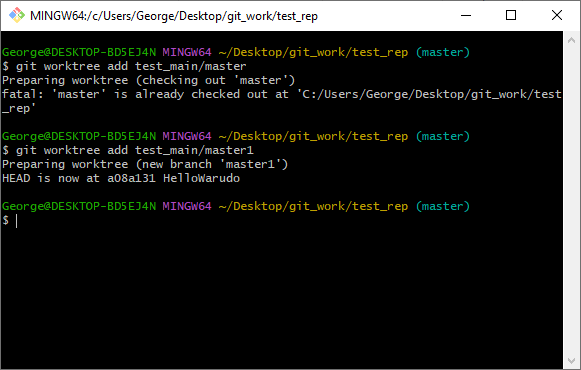
## 15. Worktree

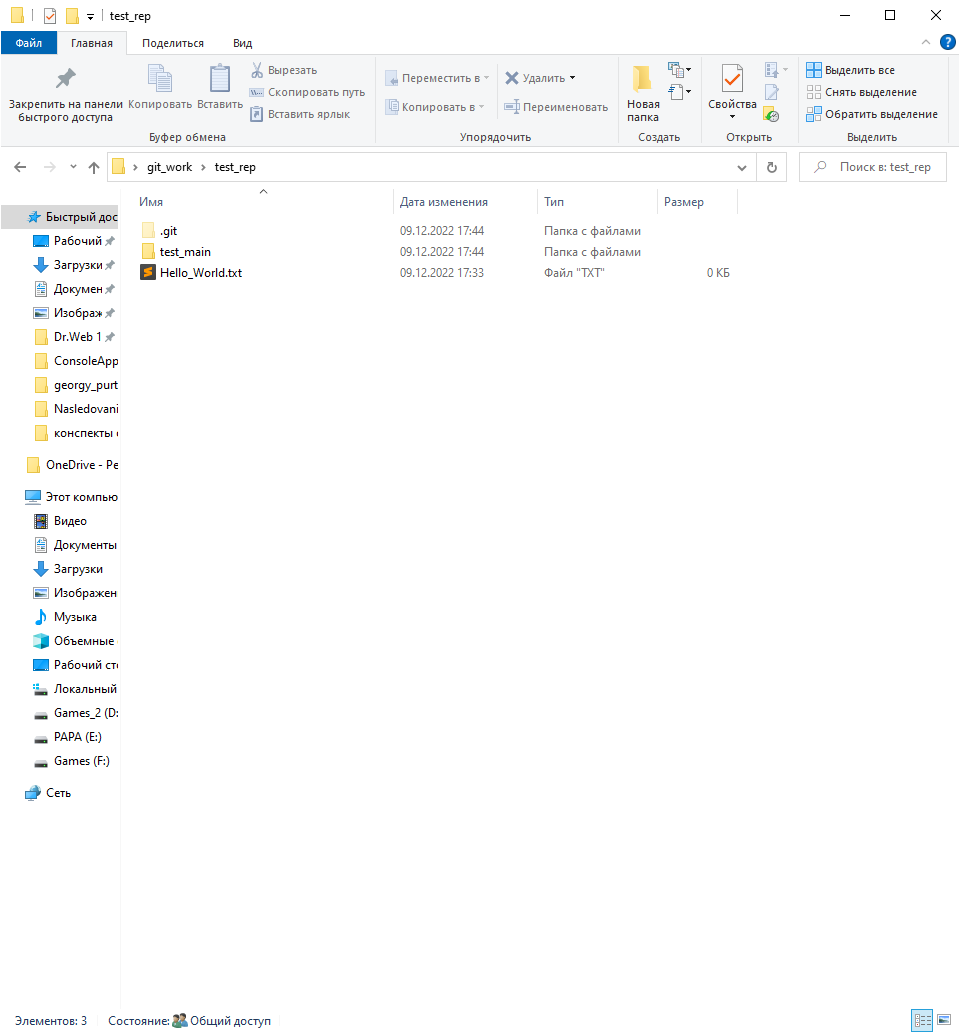
Worktree позволяет выгружать в отдельную папку единичные ветки и работать с ними параллельно работе с текущей рабочей веткой.

Рабочие деревья позволяют держать запущенными две версии вашего софта, что может быть полезно для сравнения поведения после внесённых правок.

Добавить рабочую ветку

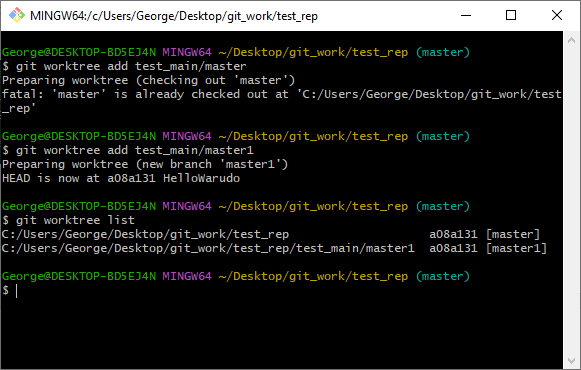
**git worktree add ../folder name-of-branch**





Посмотреть список рабочих деревьев

**git worktree list**



Когда все изменения сделаны и отправлены на сервер можно просто удалить эту папку, а затем выполнить команду которая всё подчистит.

**git worktree prune**

## Источники

1) <https://raw.githubusercontent.com/VetrovSV/Programming/master/git_lec.pdf>

2) [https://git-scm.com](https://git-scm.com/book/ru/v2/%D0%92%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%B8%D0%B5-%D0%9F%D0%B5%D1%80%D0%B2%D0%BE%D0%BD%D0%B0%D1%87%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F-%D0%BD%D0%B0%D1%81%D1%82%D1%80%D0%BE%D0%B9%D0%BA%D0%B0-Git)

3) <https://gist.github.com/rdnvndr/cb21a06c5a71fd71213aed1619380b8e>

4) <https://inoyakaigor.ru/blog/85>