

Git (система управления версиями). Основы работы

Детальная информация по работе с git: <https://git-scm.com/book/ru/v2>

Примечание: в данном случае не рассматриваются вопросы настройки удалённого репозитория, связывания удалённого и локального репозитория, настройки прав доступа к репозиториям, генерации ключей для работы по защищённому протоколу и т.д. Предполагается, что вся предварительная работа по настройке уже проведена. Также не рассматриваются часто используемые команды скачивания из удалённого репозитория и просмотра истории и структуры изменений.

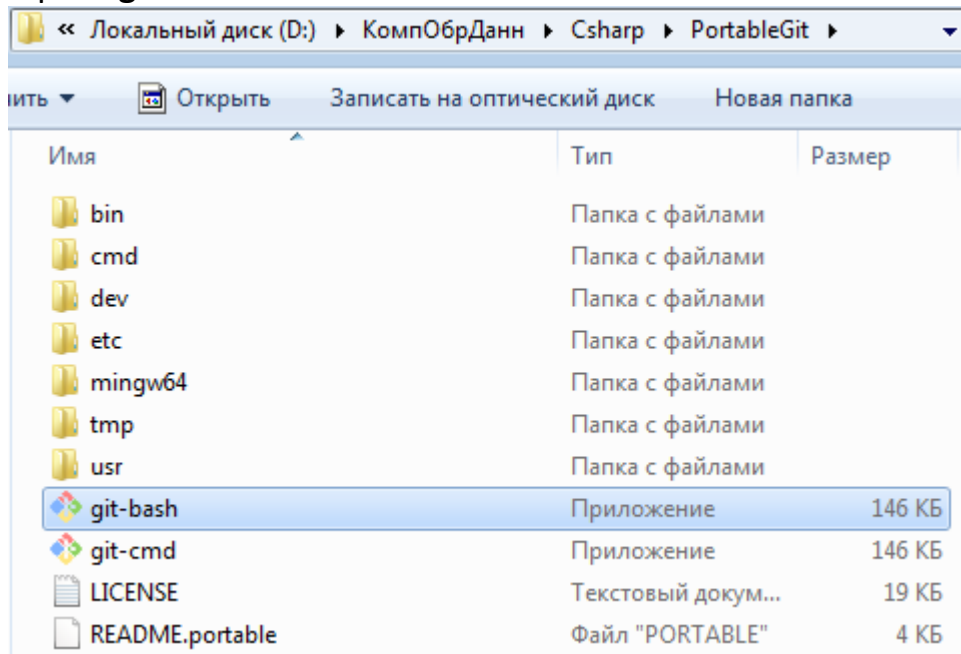
Краткое описание алгоритма действий

(*более развёрнутое описание этих же команд с пояснениями находится далее по тексту)

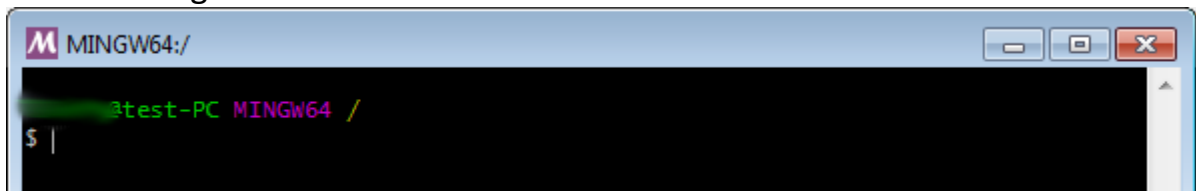
1. Перейти в Проводнике в папку **D:\КомпОбрДанн\Csharp\PortableGit**
2. Запустить файл **git-bash.exe**, появится окно git-консоли для ввода всех нижеследующих команд:
3. **cd "d:\КомпОбрДанн\Csharp"** – перейти в папку
4. **git clone git@github.com:student-programming/Lab1_Calculator.git** - клонировать проект из удалённого репозитория на локальный компьютер
5. **cd "d:\КомпОбрДанн\Csharp\Lab1_Calculator"** – перейти в папку проекта
6. **git checkout -b EiE2807_IvanovAB** – создать новую ветку и переключиться на неё. Каждый студент должен создать свою собственную ветку. Имя ветки должно иметь в названии номер группы студента и его ФИО. Имя ветки должно содержать только цифры, некоторые знаки и английские буквы.
7. **git push -u origin EiE2807_IvanovAB** – отправить созданную ветку на удалённый репозиторий, связав тем самым локальный и удалённый репозитории
8. **git status** – проверка наличия изменений в проекте
9. внести изменения в проект
10. **git status** – проверка наличия изменений в проекте
11. **git add *** – подготовить (добавить) все изменения для создания коммита (снимка промежуточного состояния проекта)
12. **git status** – проверка наличия изменений в проекте
13. **git commit -m "Text comment"** – создать коммит (снимок промежуточного состояния проекта) с комментарием
14. **git push** – отправить коммит в удалённый репозиторий
15. повторить пункты 9 – 14 необходимое количество раз

Описание алгоритма действий с пояснениями

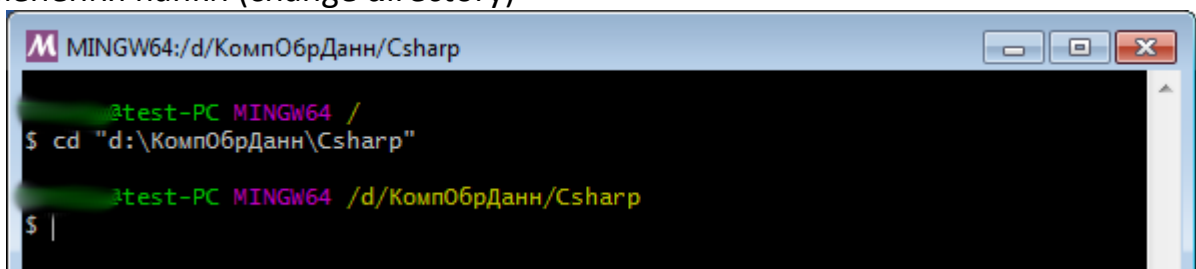
1. Перейти в Проводнике в папку **D:\КомпОбрДанн\Csharp\PortableGit**
2. Запустить файл **git-bash.exe**



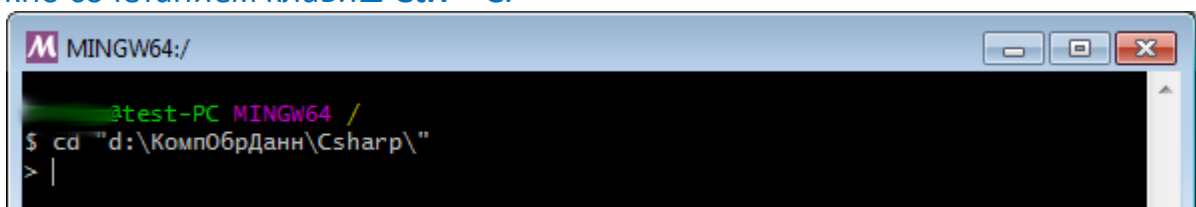
3. Появится окно git-консоли



4. В окне консоли ввести команду **cd "d:\КомпОбрДанн\Csharp"** (перед последними двойными кавычками не ставить обратный слеш!). Команда **cd** – это команда изменения папки (**change directory**)



Если всё-таки ввели перед последними двойными кавычками обратный слеш нажали **Enter**, и получилась ситуация как на рисунке ниже, то отменить команду можно сочетанием клавиш **Ctrl + C**.

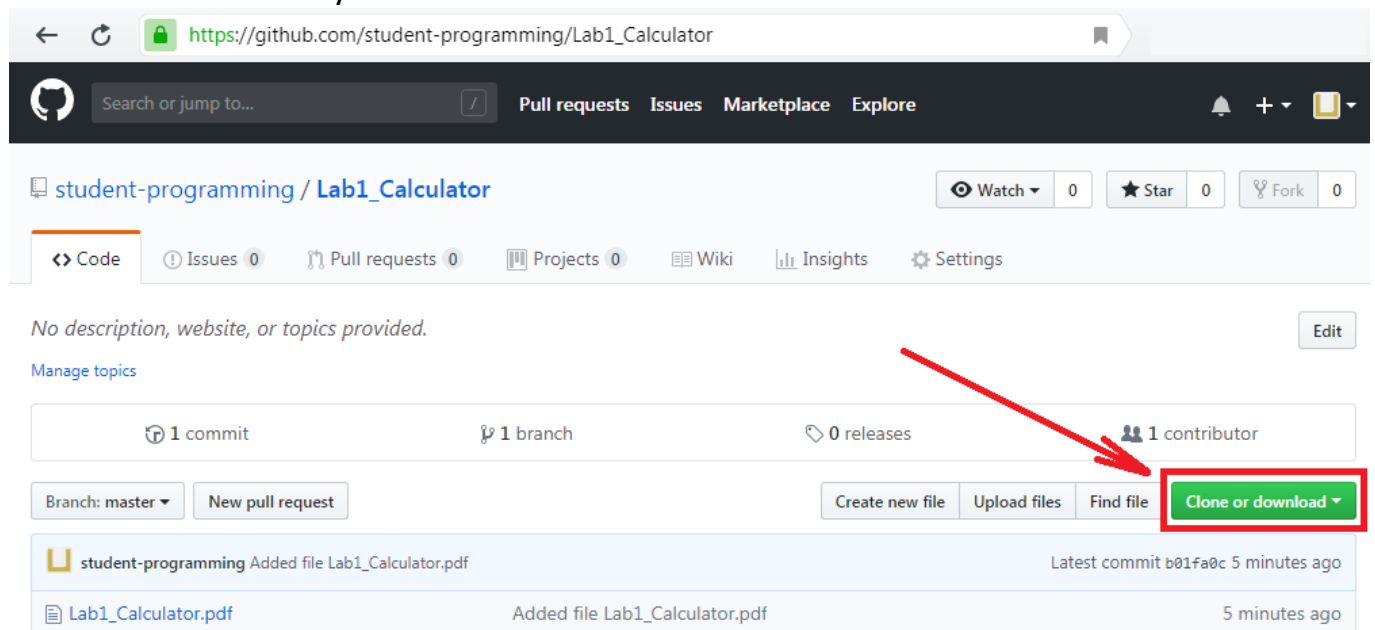


5. В браузере перейти по ссылке

https://github.com/student-programming/Lab1_Calculator

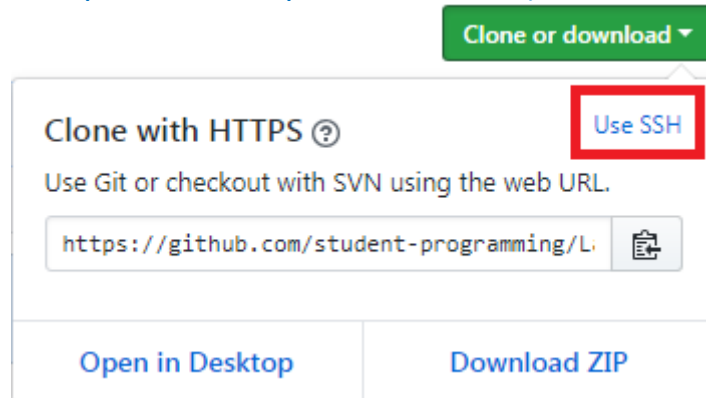
Ссылка имеет следующий формат: **домен/учётная запись/репозиторий (проект)**.

6. Нажать на кнопку **Clone or Download**

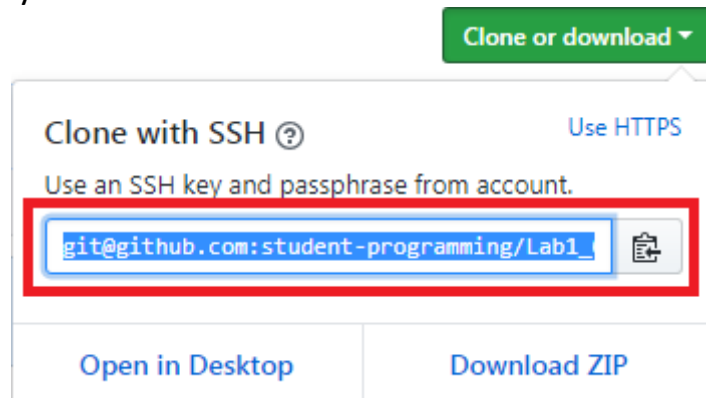


7. В появившемся окне нажать на **Use SSH**

([https](#) и [ssh](#) – сетевые протоколы передачи данных)

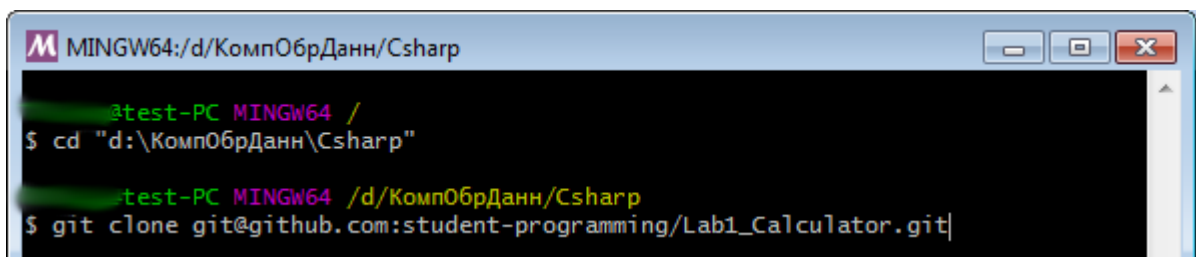
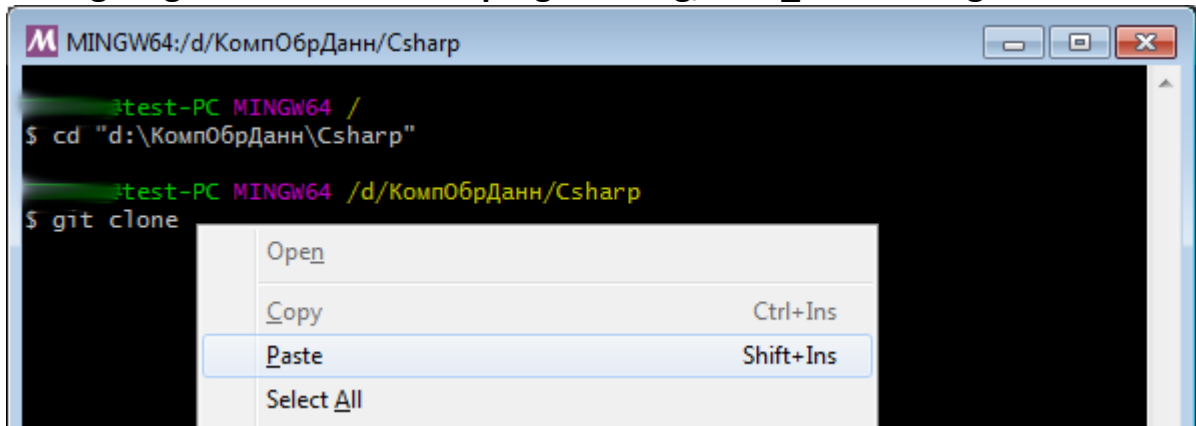


8. Скопировать ссылку

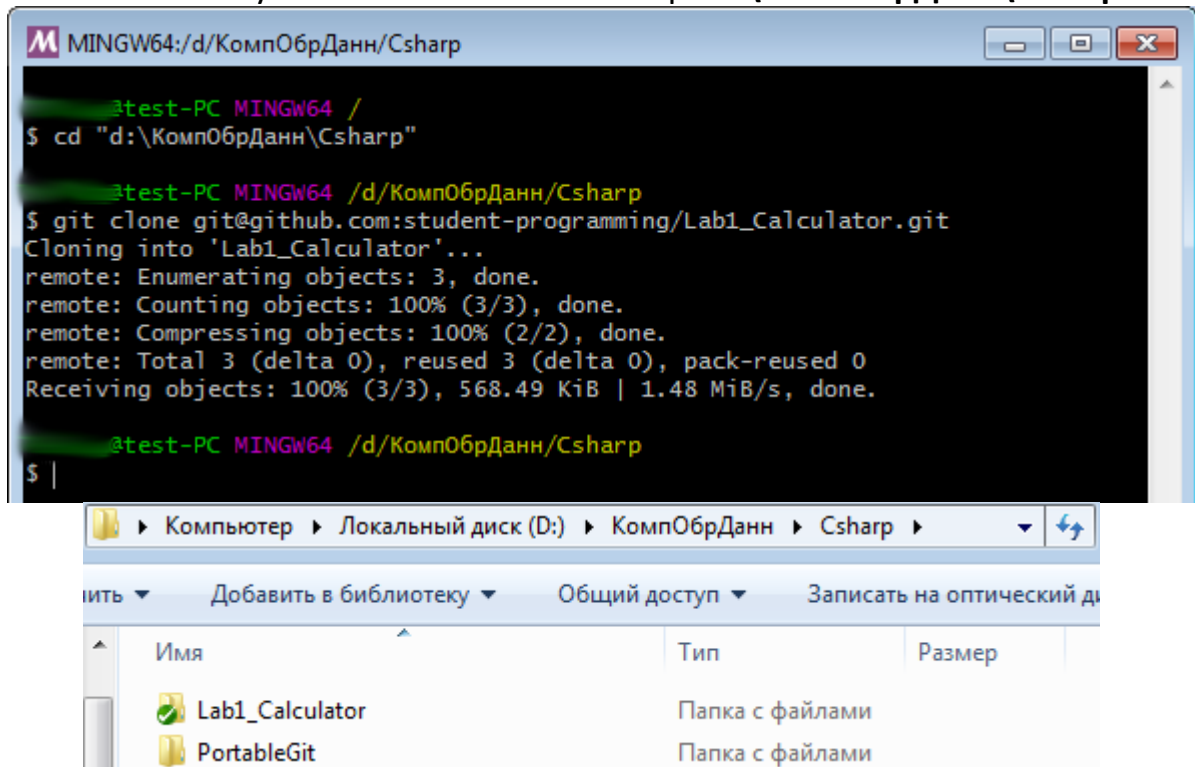


9. Вернуться в окно git-консоли, набрать команду **git clone** и не нажимая Enter, нажать сочетание клавиш **Shift+Ins**, либо нажать правой кнопкой мыши и в контекстном меню выбрать пункт **Paste**. Таким образом, полная команда будет иметь вид:

git clone git@github.com:student-programming/Lab1_Calculator.git

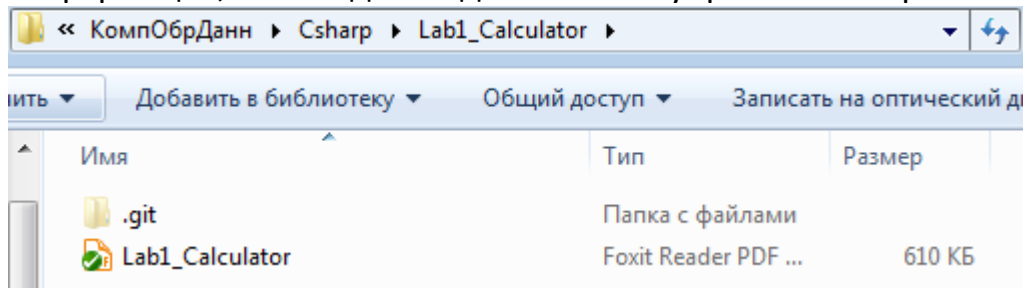


10. В результате выполнения данной команды произойдёт скачивание проекта **Lab1_Calculator** из удалённого интернет-репозитория (хранилища) домена **github.com** в папку на локальном компьютере **d:\КомпОбрДанн\Csharp**.



Репозиторием (хранилищем) проекта может быть любая папка: на локальном компьютере, в локальной сети, в сети Интернет.

11. В папке **d:\КомпОбрДанн\Csharp\Lab1_Calculator** кроме файлов проекта (в данном случае это один rfd-файл), также создаётся скрытая папка **.git**, в которой хранится информация, необходимая для системы управления версиями.



12. В git-консоли перейти в папку **Lab1_Calculator** командой:
cd "d:\КомпОбрДанн\Csharp\Lab1_Calculator"

```

MINGW64:/d/КомпОбрДанн/Csharp/Lab1_Calculator
$ cd "d:\КомпОбрДанн\Csharp"

MINGW64:/d/КомпОбрДанн/Csharp
$ git clone git@github.com:student-programming/Lab1_Calculator.git
Cloning into 'Lab1_Calculator'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), 568.49 KiB | 1.48 MiB/s, done.

MINGW64:/d/КомпОбрДанн/Csharp
$ cd "d:\КомпОбрДанн\Csharp\Lab1_Calculator"

MINGW64:/d/КомпОбрДанн/Csharp/Lab1_Calculator (master)
$ |

```

13. Текущее состояние проекта находится на ветке **master**. Имя текущей ветки выделено голубым цветом в круглых скобках в конце строки.

Git (система управления версиями) основана на ветвлении и снимках промежуточного состояния (коммитах). Веток может быть произвольное количество. На каждой ветке может быть произвольное число коммитов.

Как правило, на ветках располагается какой-либо отдельный функционал проекта, а в коммитах зафиксированы промежуточные шаги по достижению данного функционала.

От каждой ветки можно сделать новое ответвление, ветки можно сливать вместе (разрешая при этом коллизии) или удалять, можно переключаться на любой коммит любой ветки в любое время работы над проектом, вносить и отменять изменения, сделанные в проекте.

14. До внесения любых изменений в скачанный исходный проект (до изменения существующих или создания новых файлов в папке **d:\КомпОбрДанн\Csharp\Lab1_Calculator**) необходимо, **во-первых**, создать новую ветку проекта и переключиться на неё.

15. Создание ветки и переключение на неё делается или двумя последовательными командами:

git branch branchname (создать новую ветку с именем **branchname**)

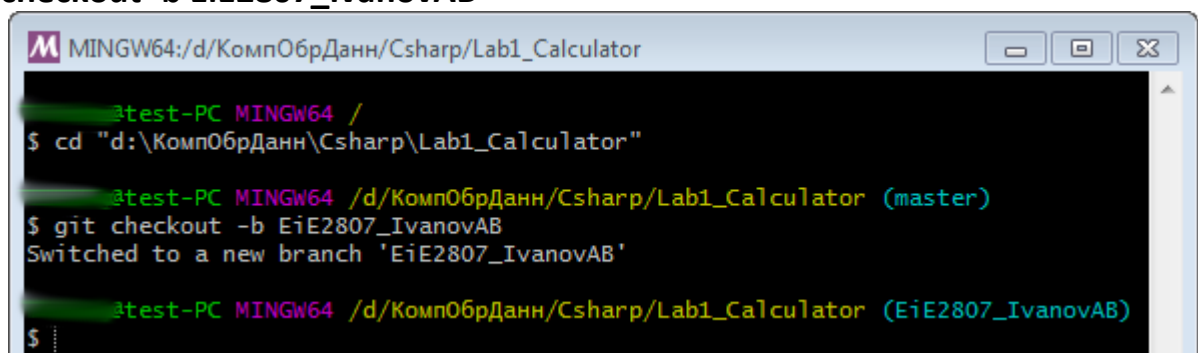
git checkout branchname (переключиться на ветку с именем **branchname**)

или же это можно сделать одной командой:

git checkout -b branchname

Каждый студент должен создать свою собственную ветку. Имя ветки не должно содержать кириллические символы (русские буквы), допустимы только латинские символы (английские буквы), цифры и некоторые знаки. Имя ветки должно иметь в названии номер группы студента и его ФИО, например, **EiE2807_IvanovAB**, тогда команда может выглядеть следующим образом:

git checkout -b EiE2807_IvanovAB



```

MINGW64:/d/КомпОбрДанн/Csharp/Lab1_Calculator
$ cd "d:\КомпОбрДанн\Csharp\Lab1_Calculator"

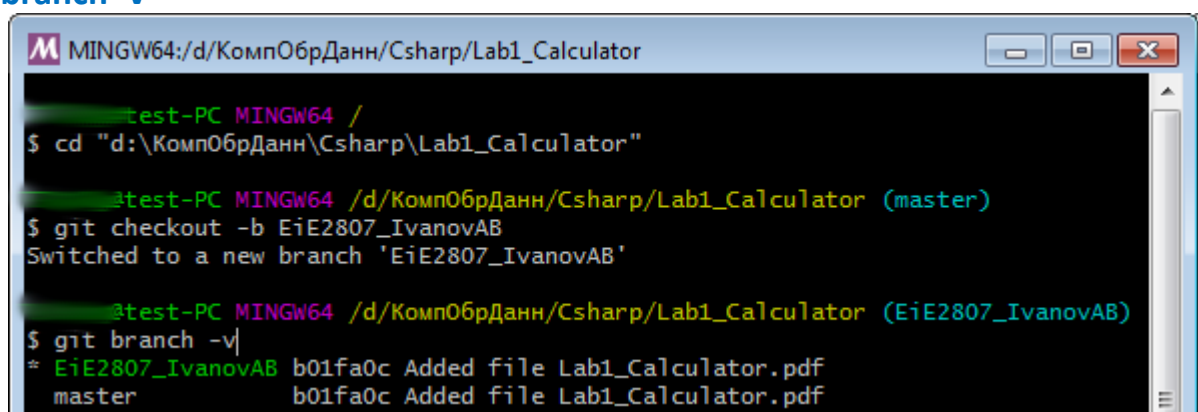
MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (master)
$ git checkout -b EiE2807_IvanovAB
Switched to a new branch 'EiE2807_IvanovAB'

MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$
  
```

Данная ветка создаётся в локальном репозитории на локальном компьютере (в папке **d:\КомпОбрДанн\Csharp\Lab1_Calculator**), но в удалённом репозитории (репозитории домена **github.com** в интернете) пока что никаких изменений нет.

Посмотреть список веток в локальном репозитории можно командой

git branch -v



```

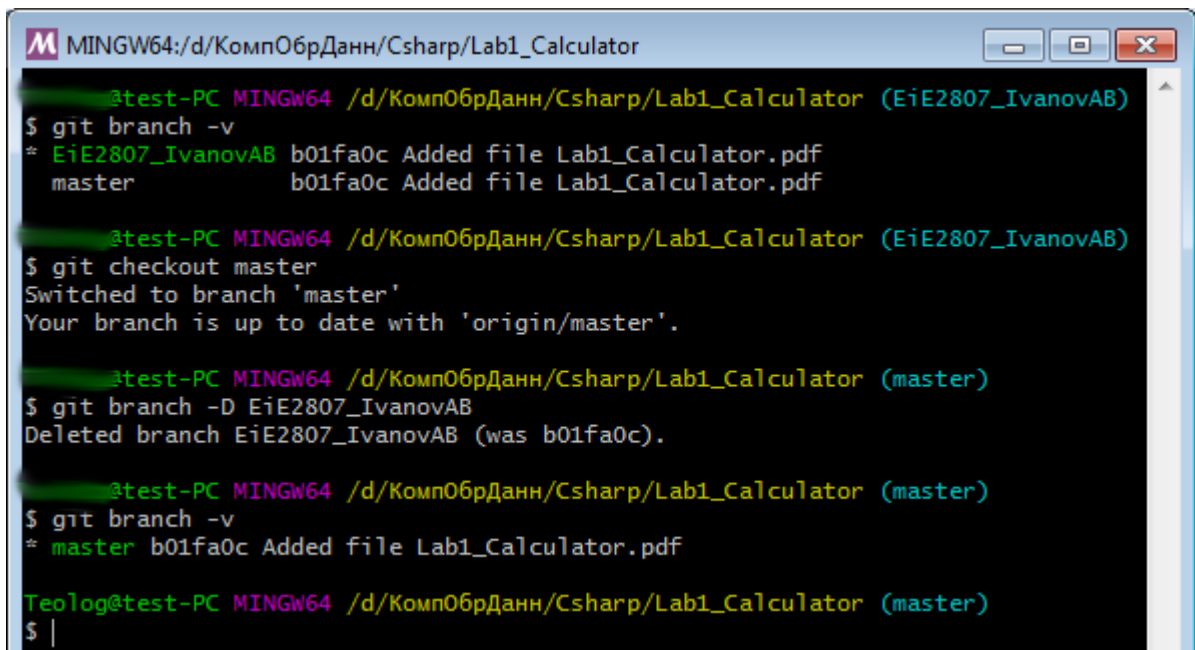
MINGW64:/d/КомпОбрДанн/Csharp/Lab1_Calculator
$ cd "d:\КомпОбрДанн\Csharp\Lab1_Calculator"

MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (master)
$ git checkout -b EiE2807_IvanovAB
Switched to a new branch 'EiE2807_IvanovAB'

MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$ git branch -v
* EiE2807_IvanovAB b01fa0c Added file Lab1_Calculator.pdf
  master          b01fa0c Added file Lab1_Calculator.pdf
  
```

В данном случае существуют две ветки: основная **master** (созданная автоматически в локальном репозитории при клонировании проекта с удалённого репозитория), и только что созданная **EiE2807_IvanovAB**.

В случае, если при создании ветки в её имени была допущена ошибка, то эту ветку можно удалить. Для этого необходимо сначала переключиться обратно на ветку **master** командой, **git checkout master**, а затем удалить ветку **EiE2807_IvanovAB** командой **git branch -D EiE2807_IvanovAB**



```

MINGW64:/d/КомпОбрДанн/Csharp/Lab1_Calculator
@test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$ git branch -v
* EiE2807_IvanovAB b01fa0c Added file Lab1_Calculator.pdf
  master          b01fa0c Added file Lab1_Calculator.pdf

@test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.

@test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (master)
$ git branch -D EiE2807_IvanovAB
Deleted branch EiE2807_IvanovAB (was b01fa0c).

@test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (master)
$ git branch -v
* master b01fa0c Added file Lab1_Calculator.pdf

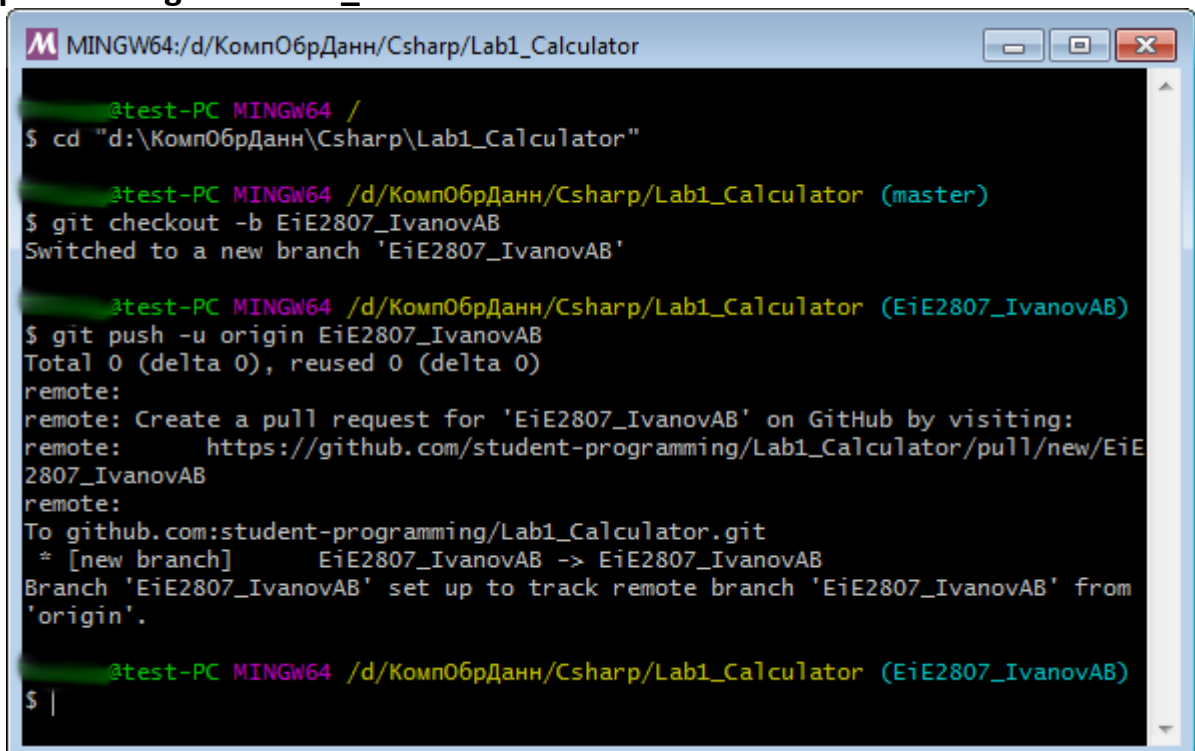
Teolog@test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (master)
$ |

```

16. До внесения любых изменений в скачанный исходный проект (до изменения существующих или создания новых файлов в папке **d:\КомпОбрДанн\Csharp\Lab1_Calculator**) необходимо, **во-вторых**, отправить информацию о созданной ветке на удалённый репозиторий.

17. Отправка информации о вновь созданной ветке в удалённый репозиторий осуществляется один раз командой:

git push -u origin EiE2807_IvanovAB



```

MINGW64:/d/КомпОбрДанн/Csharp/Lab1_Calculator
@test-PC MINGW64 /
$ cd "d:\КомпОбрДанн\Csharp\Lab1_Calculator"

@test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (master)
$ git checkout -b EiE2807_IvanovAB
Switched to a new branch 'EiE2807_IvanovAB'

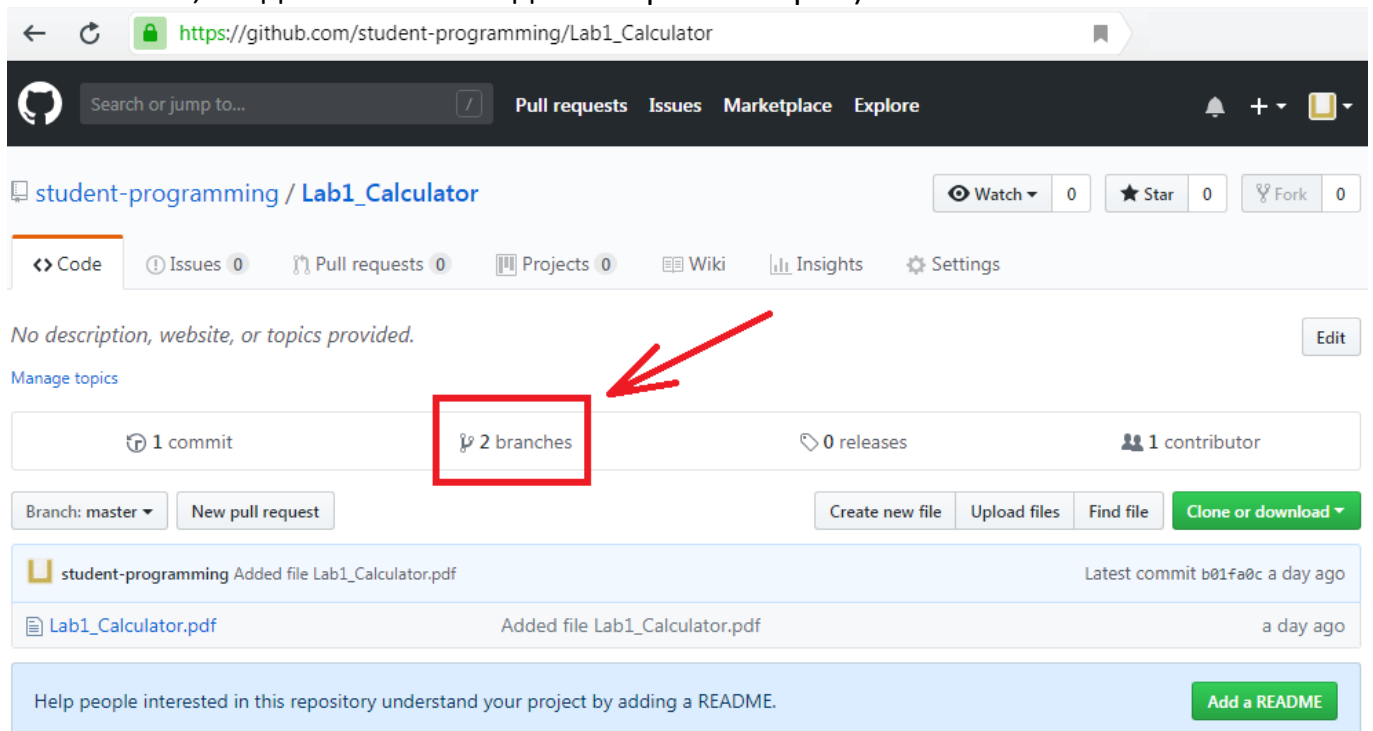
@test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$ git push -u origin EiE2807_IvanovAB
Total 0 (delta 0), reused 0 (delta 0)
remote:
remote: Create a pull request for 'EiE2807_IvanovAB' on GitHub by visiting:
remote:   https://github.com/student-programming/Lab1_Calculator/pull/new/EiE2807_IvanovAB
remote:
To github.com:student-programming/Lab1_Calculator.git
 * [new branch]      EiE2807_IvanovAB -> EiE2807_IvanovAB
Branch 'EiE2807_IvanovAB' set up to track remote branch 'EiE2807_IvanovAB' from 'origin'.

@test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$ |

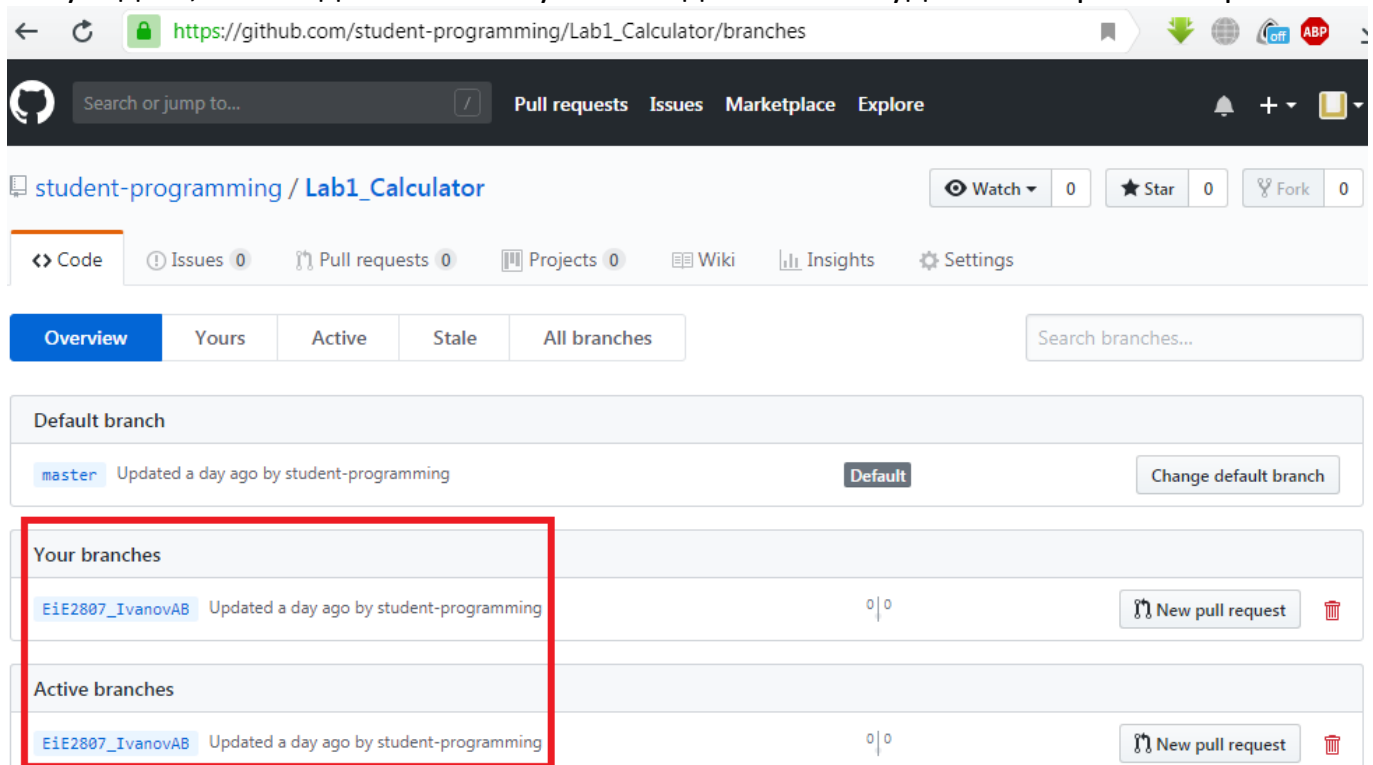
```

После этой команды локальная ветка **EiE2807_IvanovAB** и ветка **EiE2807_IvanovAB** в удалённом репозитории будут связаны между собой. После такой команды (которая выполняется единожды), в дальнейшем можно будет отправлять изменения на удалённый репозиторий лишь выполняя команду **git push**.

18. После этого, если в браузере обновить открытую страницу (по ссылке https://github.com/student-programming/Lab1_Calculator), то можно увидеть, что в репозитории изменилось (увеличилось) количество веток (зависит от количества человек, создавших ветки в данном репозитории).



19. При переходе по выделенной на рисунке выше красным ссылке с количеством веток (https://github.com/student-programming/Lab1_Calculator/branches) можно увидеть, что созданная ветка успешно добавлена в удалённый репозиторий.



20. Проверить статус изменений файлов/папок локального репозитория командой:
git status

```

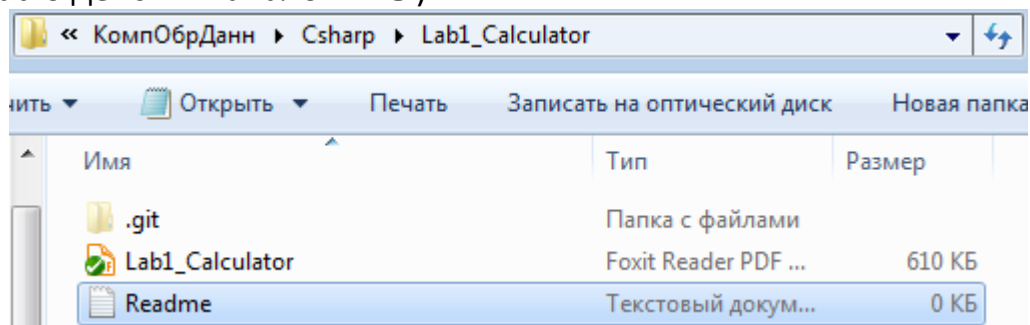
MINGW64:/d/КомпОбрДанн/Csharp/Lab1_Calculator
3test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$ git status
On branch EiE2807_IvanovAB
Your branch is up to date with 'origin/EiE2807_IvanovAB'.

nothing to commit, working tree clean
3test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$ |

```

На данный момент никаких изменений в исходный проект не внесено.

21. После того, как локальная и удалённая ветки связаны, можно вносить изменения в локальный репозиторий в папку **d:\КомпОбрДанн\Csharp\Lab1_Calculator** (создавать / редактировать / удалять файлы и папки).
Для примера создадим в этой папке текстовый файл **Readme.txt** (для C#-проекта Visual Studio действия аналогичны).



22. Теперь команда **git status** вернёт другой результат. Красным цветом подсвечены файлы / папки в которых произошли изменения.

```

MINGW64:/d/КомпОбрДанн/Csharp/Lab1_Calculator
3test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$ git status
On branch EiE2807_IvanovAB
Your branch is up to date with 'origin/EiE2807_IvanovAB'.

nothing to commit, working tree clean

3test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$ git status
On branch EiE2807_IvanovAB
Your branch is up to date with 'origin/EiE2807_IvanovAB'.

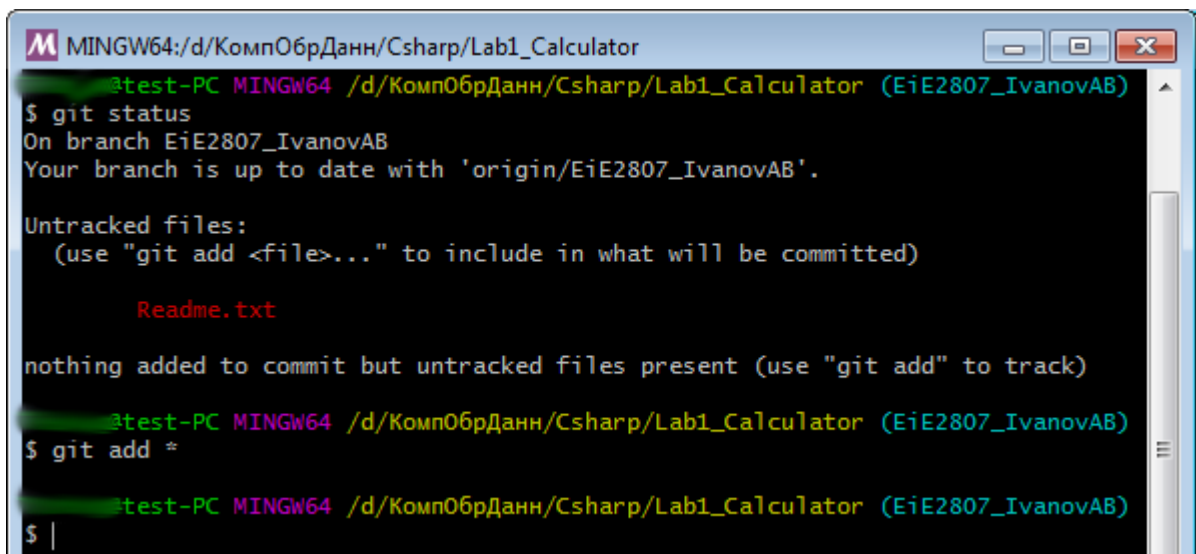
Untracked files:
  (use "git add <file>..." to include in what will be committed)

  Readme.txt

nothing added to commit but untracked files present (use "git add" to track)
3test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$ |

```

23. Далее надо подготовить файлы для создания коммита (снимка промежуточного состояния локальной версии проекта). Для этого используется команда **git add ***



```

MINGW64:/d/КомпОбрДанн/Csharp/Lab1_Calculator
@test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$ git status
On branch EiE2807_IvanovAB
Your branch is up to date with 'origin/EiE2807_IvanovAB'.

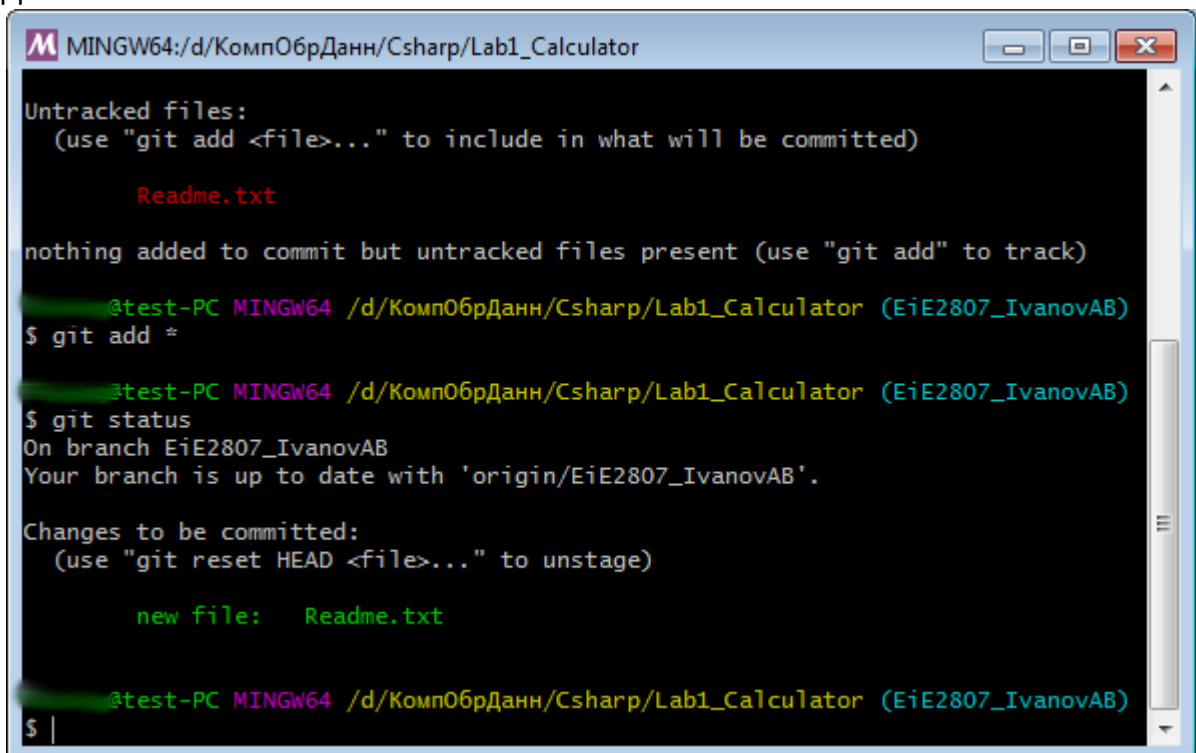
Untracked files:
  (use "git add <file>..." to include in what will be committed)

    Readme.txt

nothing added to commit but untracked files present (use "git add" to track)
@test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$ git add *
@test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$ |

```

24. Проверить состояние подготовленных для коммита файлов снова командой **git status**. Зелёным цветом подсвечены файлы / папки подготовленные для создания коммита.



```

MINGW64:/d/КомпОбрДанн/Csharp/Lab1_Calculator
Untracked files:
  (use "git add <file>..." to include in what will be committed)

    Readme.txt

nothing added to commit but untracked files present (use "git add" to track)
@test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$ git add *
@test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$ git status
On branch EiE2807_IvanovAB
Your branch is up to date with 'origin/EiE2807_IvanovAB'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file:   Readme.txt

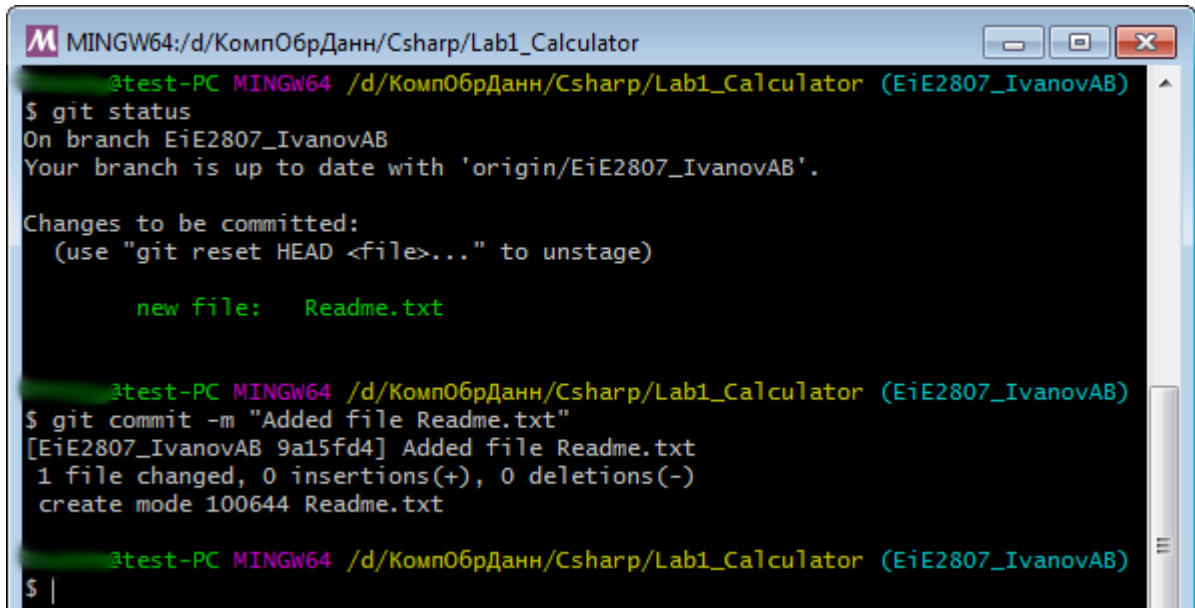
@test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$ |

```

25. Создание коммита (снимка промежуточного состояния файлов / папок проекта), т.е. фиксация изменений в проекте в некоторый момент времени на некоторой стадии проекта делается командой:
git commit -m "Text" (где **Text** – это произвольная строка текста, допускаются только цифры, некоторые знаки и латинские символы (английские буквы))

Для примера можно выполнить следующую команду:

git commit -m "Added file README.txt"



```

MINGW64:/d/КомпОбрДанн/Csharp/Lab1_Calculator
@test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$ git status
On branch EiE2807_IvanovAB
Your branch is up to date with 'origin/EiE2807_IvanovAB'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

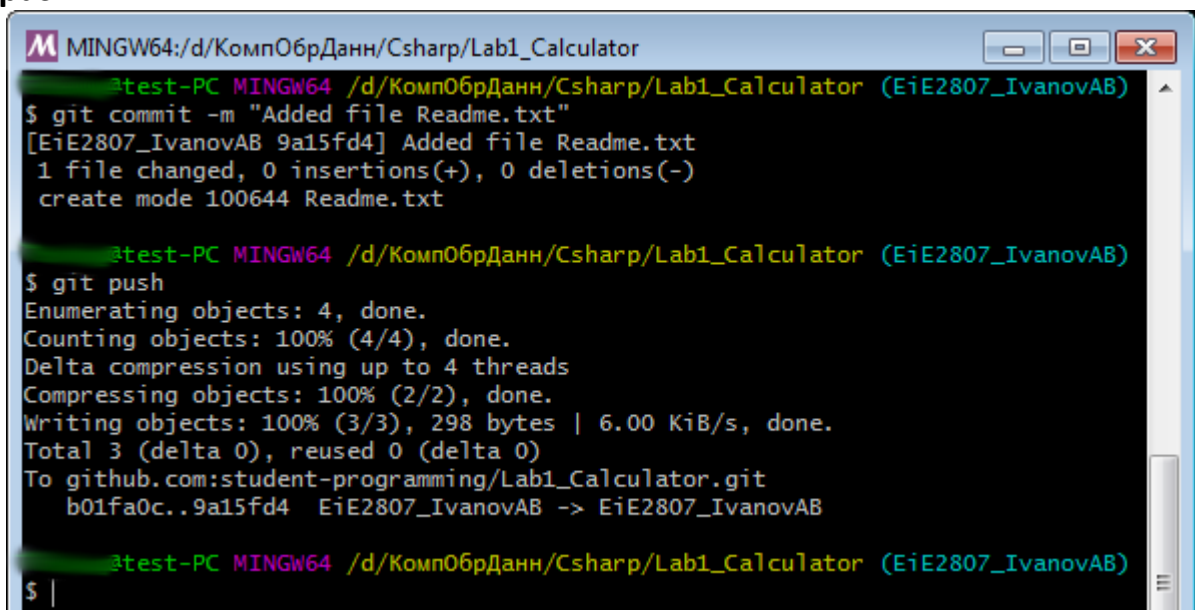
    new file:   README.txt

@test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$ git commit -m "Added file README.txt"
[EiE2807_IvanovAB 9a15fd4] Added file README.txt
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 README.txt

@test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$
  
```

26. Наконец созданный коммит (снимок промежуточного состояния проекта) необходимо отправить на удалённый репозиторий командой:

git push



```

MINGW64:/d/КомпОбрДанн/Csharp/Lab1_Calculator
@test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$ git commit -m "Added file README.txt"
[EiE2807_IvanovAB 9a15fd4] Added file README.txt
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 README.txt

@test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$ git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 298 bytes | 6.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:student-programming/Lab1_Calculator.git
   b01fa0c..9a15fd4  EiE2807_IvanovAB -> EiE2807_IvanovAB

@test-PC MINGW64 /d/КомпОбрДанн/Csharp/Lab1_Calculator (EiE2807_IvanovAB)
$
  
```

27. Изменения в удалённом репозитории можно посмотреть по ссылке (https://github.com/student-programming/Lab1_Calculator) обновив страницу браузера.

student-programming / Lab1_Calculator

No description, website, or topics provided.

Manage topics

1 commit 2 branches 0 releases 1 contributor

Your recently pushed branches:

EiE2807_IvanovAB (4 minutes ago) Compare & pull request

Branch: master New pull request Create new file Upload files Find file Clone or download

student-programming Added file Lab1_Calculator.pdf Latest commit b01fa0c a day ago

Lab1_Calculator.pdf Added file Lab1_Calculator.pdf a day ago

Help people interested in this repository understand your project by adding a README. Add a README

По ссылке на коммиты можно посмотреть содержание коммитов:

https://github.com/student-programming/Lab1_Calculator/commits/master - ветка master

Code Issues 0 Pull requests 0

Branch: master

Commits on Jan 30, 2019

Added file Lab1_Calculator.pdf

student-programming committed a day ago

Switch branches/tags

Find or create a branch...

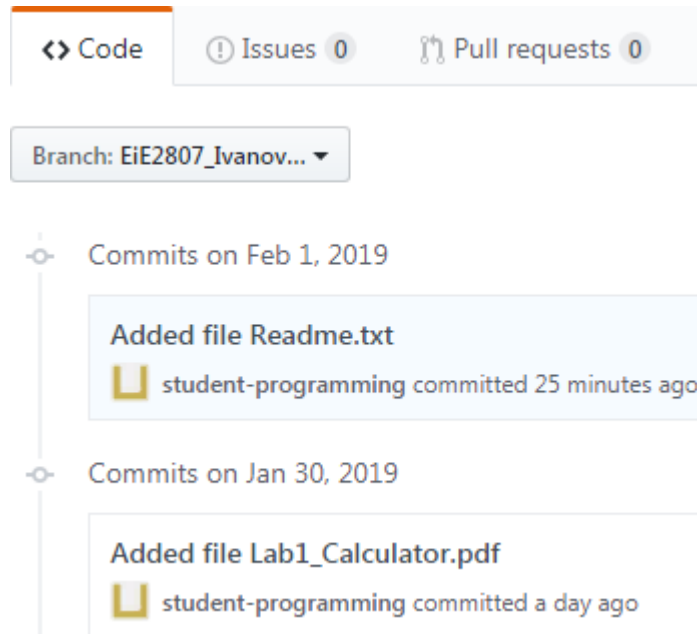
Branches Tags

EiE2807_IvanovAB

✓ master

Или переключиться на ветку **EiE2807_IvanovAB**

https://github.com/student-programming/Lab1_Calculator/commits/EiE2807_IvanovAB



Нажав на зелёную кнопку **Compare & pull request** можно выполнить сравнение веток и создать запрос на их слияние.

28. Для создания нового коммита повторить пункты 22 – 26.
29. Остальные часто используемые команды работы с git (например, **git fetch** для скачивания всех изменений из удалённого репозитория со всеми ветками и коммитами, **git pull** для скачивания изменений с одной текущей удалённой ветки, **git merge** для слияния веток, **git log** для просмотра истории коммитов и структуры веток) могут быть рассмотрены в дальнейшем по мере необходимости.