**git init .** – initializes directory for working with Git (make it first in your directory)

**git status** – has a look at the current status

**git add .** – adds all files for commit

**git add <name of the file> (example: git add file1.txt)** – adding file/files for committing

**git rm <name of the file>** – deleting file/files

**git commit –m”<Your comment>” (example: git commit –m”Version 1.0”)** – committing your last change/changes

**git log** – history of your commits

**git log -1** – your last commit (example: **git log -3** – third commit you made from the end)

**git log -1 -p** – last commit in details (**git log -3 -p** – third commit from the end in details)

**git checkout -- <name of the file>** – rollbacks in the file just before commit has been done only **(!DON’T FORGET SPACE AFTER DASHES!**), it doesn’t work in case commit is done

**git diff -staged** – shows the last editions or new files before doing commit

**.gitignore** – text file with exceptions that git will not pay attention on; it could be certain file, directory/folder, format-type (example: .log, file1.txt, logs/ etc.)

**git clone <HTTPS link>** – clones data from git-hub directory by the mentioned link

**git push origin** – sends local changes on git-hub (enter your user-name and password on request)

**git remote -v** – shows a link used from git-hub

**git remote set -url origin <HTTPS link>** - changes current link with git-hub by HTTP-link (example: git remote set -url origin <http://git.yoursite.com>)

**Setting SSH-key for synchronization/connection with git-hub:**

1) launch Git-Bash → enter “**ssh-keygen**” (without quotes), 2) x3 “Enter” button, 3) open with text-editor created key (C:\Users\[User]\[required file.pub]), 4) go to git-hub and open “Settings”, 5) “SSH and GPG keys”, 6) add new SSH-key.

**git branch** – has a look at existing branches, also shows branch you are working in (marked in green with asterisk)

**git branch <name of the file>** – creates branch with name <name of the file> (example: **git branch mybranch**)

**git checkout <name of the branch>** – goes to the branch <name of the branch> (example: **git checkout <name of the branch>)**

**Deleting branch:**

1) go to the master branch: enter **git checkout master**, 2) delete desired/undesired branch: enter **git branch -d <desired/undesired branch>**

**git checkout -b <name of the branch>** – creates and go to the created branch (example: **git checkout -b mybranch**)

**git merge <name of the branch>** – links branch to the master-branch (example: **git merge mybranch**)

**git branch -D <branch name>** – deletes committed branch (example: **git branch -D mybranch**)

**git checkout <hash of the commit>** – rollbacks to the required/desired commit (example: **git checkout 96dc5352f8f766ed3422e4bf74e66a4cead73f54**)

**Editing the last commit without adding commit about the file edition/changing:**

1) enter **git commit--amend**, 2) press “Insert” button, 3) enter the changes you’d like to do, 4) press “Esc” button, 5) enter “:wq!” (without quotes)

**git reset -hard HEAD ~ 2** – hard rollbacks (with deleting all the files) to the required number of commits (example: **git reset -hard HEAD ~4** – rollbacks to 4commits from the end)

**git reset -soft HEAD~3** – soft rollbacks (with no deleting all the files, just deleting the commits) to the required number of commits (example: **git reset -soft HEAD~1** – rollbacks to 1 commit from the end)