GIT TUTORIAL

1) saving and sharing program listnings are common programmer problems. One should find git repository as great solution for publication.
2) sign up to GitHub or Bitbucket
3) install git program:
    #sudo apt-get install git
4) configure github account:
    #git config --global user.email „githubAccount@email”
    #git config –global user.name „githubUser”
5) add local repository for creating new project
    #git init
6) create new repository on your Github account (via internet explorer) and add locally:
    #git remote add origin https://fromGitHubAccountRepositoryAddress
    ####git locally-stored remote origin in separate directory (git files)
    #mkdir pathToLocalOriginDirectory && cd pathToLocalOriginDirectory
    #git init
    #cd repositoryPath
    #git remote add origin pathToLocalOriginDirectory
    #git checkout -b main #create new branch "main"
    ####provide changes to new repository for first commit
    #git status
    #git add .
    #git commit -m "first commit"
    #git push --set-upstream origin main #only first push must point to remote, and repo branch
7) write / add some code and add to repository:
    #git status
    #git add .
8) local repo save:
    #git commit -m „comments on your code publically avaiable”
9) first remote repo save:
    #git push –set-upstream origin master
10) standard remote repo save (P.S. for basic usage above mentioned steps are sufficient!):
    #git push

11) add new/delete branch (f. e. there are more than one programmer) to repo:
    #git branch branchName
    #git branch -d branchDeleted
12) branch programms view:
    #git checkout branchName
13) first save remotely branch:
    #git push –set-upstream origin branchName
14) standard remote repo save:
    #git push origin branchName
15) join (merge) few branches in one:
    #git merge origin/branchName
    #git push
16) joining conflicts – meld package:
    #sudo apt-get install meld
    #git mergetool
    #git commit -m „comments on branches joining”
    #git push
17) commits and local changes management:

#sudo apt-get install -y gitk git-gui
#gitk #commits
#git gui #local changes

18) add new files as single lines or pieces:
#git add -N .

19) pull remote changes as information only:
#git fetch --all

20) make patches to official code:
#git clone https://www.github.com/uset/repo.git
#git branch #list all branches in repository
*main
#git checkout -b SPECIALIZED_DEVELOPMENT #create new branch
#git checkout SPECIALIZED_DEVELOPMENT
#git branch
*SPECIALIZED_DEVELOPMENT
main
####apply changes to code
#git status
#git add .
#git commit -m "approved changes"
#git format-patch main #point comparison branch with all commit's patches, or:
#git format-patch main -o patches #all patches will be put into directory "patches", or:
#git log #list all commits with fingerprints
#git format patch -1 selected_single_fingerprint_from_git_log main destDirectory
#git checkout main

21) apply patches from **SPECIALIZED_DEVELOPMENT** branch to **main** branch
#git checkout main
#for p in `ls -1 patchesPath/*.patch`; do git am $p; done #apply all changes to main code
#git am 00010-approved-changes.patch
#git log

1) downloading some repository from Internet - firstly install git program:
#sudo apt-get install git

2) configure github account:
#git config --global user.email „githubAccount@email”
#git config –global user.name „githubUser”

3) download (clone) repo:
git clone https://repositoryAddress

1) download private repository with assymetric key pairs in Command Line Interface:
#sudo apt-get install putty-tools

2) Putty-based generated key pairs (public+private) are saved by default in Putty format. Often it is necessary to be converted to OpenSSH format
cp user_p*key ~/.ssh/
#puttygen ~/.ssh/user_private_putty_key.ppk -O private-openssh -o
~/.ssh/user_private_openssh_key
#puttygen ~/.ssh/user_public_putty_key -O public-openssh -o
~/.ssh/user_public_openssh_key

3) write corresponding git config file:
#vi ~/.ssh/config
host github.com

HostName github.com
Identityfile ~/.ssh/user_private_openssh_key
User PiotrLenarczyk