Git / GitHub Tutorial

1) Create Repository
    a. Create a folder for your git repositories (e.g. c:\git).
    b. Make a directory (mkdir TestGit) called TestGit and go into it (cd TestGit) using the Git Bash
    c. Convert the TestGit directory to a git Repository: **git init**
2) Commit Changes to the Repository
    a. Create a file called test.txt in the TestGit directory and write some text in it.
    b. See the unstaged files in the repository: **git status**
    c. Stage the file in preparation for adding it to the repository: **git add test.txt**
    d. See the staged files in the repository: **git status**
    e. Commit the file to the repository: **git commit –m "First File"**
    f. Change the file and do a **git add** and a **git commit** again.
3) Branches
    a. Create a branch to do a new task: **git branch task12**
    b. Display a list of all the branches: **git branch –a**
    c. Change to the branch: **git checkout task12**
    d. Create a new file in the branch called hello.txt and write some text in it.
    e. Use **git add** and **git commit** to commit the change to the branch (note that if you want to add all unstaged files use "." instead of the filename when you do the **git add**).
    f. Go back to the master (main) branch: **git checkout master**
    g. Look at the files in the folder and notice there is no hello.txt since it only exists in the branch
4) Rebase a Branch
    a. Change test.txt in master. Commit the change in master.
    b. Change back to the branch: **git checkout task12**
    c. Notice that test.txt does not have the update you made in the master branch.
    d. Rebase the branch so it contains the latest updates from the master: **git rebase master**
    e. Look at test.txt and you will see the changes from master.
5) Merge a Branch
    a. Go back to the master branch: **git checkout master**
    b. Merge the changes you made in the branch into master: **git merge task12**
    c. Notice the file created in the branch is now in master
    d. Delete the branch since it is not needed anymore: **git branch –d task12**

6) Create GitHub Repository
    a. Three ways to create a repository in Github:
        i. Create a project in Github and then clone it on your hard drive from the folder where you have your git repositories (e.g. c:\git): **git clone <Github URL>**
        NOTE: You can do this with the CS246_Class repository
        ii. Use IntelliJ to "Share Project in Github" (you will do this in assignment 4). This will create a Git repository wherever you have your project stored on your hard drive.
        iii. If you already have a git repository on your computer (which we have been doing in this tutorial), then you can first create a repository in Github and then type the following:
        git remote add origin <Github URL>
        git push -u origin master
        After doing this initial push, you can just do **git push** and **git pull**
7) Synchronize Changes with GitHub
    a. Make a change to test.txt and commit the change to your repository.
    b. Share the changes with Github:
        i. Always first get the lastest changes from Github: **git pull**
        ii. Send your changes to Github: **git push**
    c. Go into GitHub and see the change that was made
8) Merge Conflicts
    a. Simulate two changes happening by different people at the same time by going into Github and changing test.txt. Click on Commit.
    b. On your computer, change test.txt and commit your change (**git add** and **git commit**)
    c. Always do the pull first: **git pull**
    d. Notice the merge conflict. Edit the file and fix the conflict manually. Redo the **git add** and **git commit** for the updated file.
    e. Redo the pull and notice that it is "up to date": **git pull**
    f. You can now do the push: **git push**
    g. Go into GitHub and see the change that was made
9) Synchronize a Branch with Github
    a. Create a branch in your repository and commit a change. Push the branch to Github:
        i. **git branch task40** and **git checkout task40**
        ii. After modifying the file, **git add** and **git commit**
        iii. **git push –u origin task40**
    b. After the initial push of the branch, you can just do **git push** and **git pull**. Look in GitHub and see the branch. When you merge a branch, you can create a pull request in GitHub for it. This will allow for some reviews and approvals prior to merging the branch into the master.