Computer Systems Organization Recitation [Spring 2019]

CSCI-UA 0201-002

R1: Introduction & environment setup

Some slides based on Chien-Chin Huang's Spring 2018 CSO recitation

Know your staffs

- Instructor: Prof. Jinyang Li
- Recitation Instructor and grader: Lingfan Yu
- TA: Jingyu Liu
  - Office hour: Tue & Thu, 1-3 pm
  - Starting from today

Where we release course materials

- Course Website
  - https://nyu-cso.github.io/index.html
  - Recitation slides also on the course schedule page
- Piazza
  - All course announcements are released here
  - It's your responsibility to read Instructor's Note on Piazza
  - You are encouraged to ask questions on Piazza
- GitHub
  - all labs and recitations are released and should be submitted on GitHub

How to contact us

- If you have general questions about course contents or labs / recitations
  - ask on Piazza
  - come to office hour
- If you want to send us a private message
  - email cso-staff mailing list at cso-staff@cs.nyu.edu
  - Tell us who your name, your GitHub username, and your NYU NetID

What is this recitation for?

- Review course contents
- Exercises to help you understand course contents better.
- Tutorials of labs
- Go over samples of previous years before quizzes
- Explain answers after quizzes.

How are we going to proceed?

- For the first two weeks, we will focus on environment setups, usage of basic tools, etc.
  - Today we will cover environment related setups
  - Next recitation will cover programming tools like gcc, gdb, and make
- Problems driven starting from the third week
  - We will go through recitation exercises together
- Exercises will be due on Wednesday **11:00 pm**.
  - **Attention**: deadline is **NOT** 12:00 AM
  - Our script automatically pulls your submission. Don’t be late
  - Submit when you finish! Don’t wait until the deadline!!!

Is everyone required to attend recitation?

- Not necessarily
  - If you are very confident that you don’t need any guides on the exercises
- Whether or not you decide to attend recitation, you **MUST** finish each week’s recitation exercise
  - That’s 15% of your course grade

Most Importantly: Academic Integrity

- Do not use or even look at code from any other places, e.g., Internet, your friends.

- Websites like Chegg are **strictly prohibited**
  - Your paying someone to do your assignments does not mean it's not plagiarism

- We will use tools to check your code. And if the alarm sounds, we will inspect manually

- You will fail the class because of plagiarism.
  - Although we emphasized integrity over and over again, someone still broke the code and failed…

- Just don't do it!

Grace day policy

- No grace day for recitation. You must hand in recitation exercise by the deadline

- 5 grace days in total for labs

- The granularity is half day. In other words, you can request grace days in multiple of half day.
  - for example, you can request 1.5 grace days for lab1, 3.5 grace days for lab2. Then for lab 3 to 5, you have no grace days left

- How to request grace day will be posted on Piazza later (before lab1 deadline)

Setup Github

- create a GitHub account if you don’t have one
- enroll yourself in the GitHub classroom
  - Create your recitation repository by clicking the link below
    - https://classroom.github.com/a/f5qD2YPK
  - Select your Net Id
    - Don’t skip this step, or your submission will not be graded
    - Be careful! Don’t select someone else’s netid!
- If you newly get enrolled and cannot find your netid, let me know!
- Now create your labs repository by clicking the link below
  - https://classroom.github.com/a/tgJky7zi

Let’s do some statistics

- How many of you have used Unix-like systems?
- How many of you have used command line?
- How many of you have programmed in C or C++?
- How many of you have used version control softwares like Git?

Today's Topic

- Setup your virtual box
- Setup your git repository for recitation and lab
- Basic Unix commands
- Program development
  - Editor (Sublime)
  - Version control (Git)

Today's Goal

- By the end of today's recitation, you should
  - have our class virtual machine installed
  - have your recitation and lab git repository initialized
- Homework/Exercise today
  - Read through and sign the CSO Cheat Sheet

Setup Class Virtual Machine

- Why do we need a virtual machine?
  - C code will be compiled to machine code, which is highly dependent on the environment
    - Again, why? You will get an answer in CSO class
  - To make sure everyone has the same environment, we ask everyone to use virtual machine
  - No matter whether you are using Windows, Linux, or Mac, you will get a virtual machine with a system called Ubuntu 16.04

Alternative choices

- If you are already a Ubuntu 16.04 user, you don’t have to install virtual machine.
  - But it’s still recommended since it provides you with a clean environment
- If you are Windows user, and for some weird or unknown reasons your machine just can’t install VirtualBox…
  - You can try Linux subsystem
  - Google “linux subsystem” and follow MicroSoft’s tutorial

Basic virtual machine setup

- Follow instructions on https://nyu-cso.github.io/labs/ to
  - download VirtualBox 5.2.24
  - class virtual machine image (please use the image we provide)
  - Launch Virtualbox and import class VM image
  - Launch Lubuntu Linux and install necessary packages
  - username “lab”, password “lab12345”

Common questions

- When I type in my password, nothing happens
  - This is a security feature
  - Just continue typing, and hit Enter

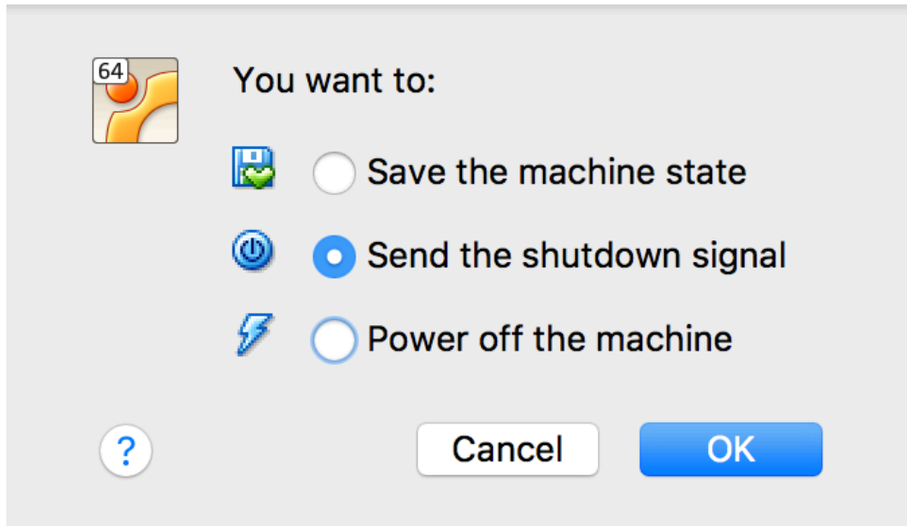
- VM fails to launch and halts with either black screen or distorted image
  - Try allocating more video memory to your VM. Shutdown your VM first. Then open "Settings", click "Display", and change "Video Memory" to a larger value

- More on https://nyu-cso.github.io/labs/ “FAQs on Virtual Box Setup” section

Advanced VM setup

- After finishing the basic setup, you are good to go
- But what if I want to
  - resize VM window to full screen
  - share clipboard between VM and host machine
  - copy files between VM and host machine
  - mount a folder of host machine in class VM
- check out Lab instruction pages for advanced setup
  - https://nyu-cso.github.io/labs/

Caveat



- NEVER “power off” virtual machine
  - You should shutdown it properly
  - The “Power off” option is equivalent to pulling the power plug on a real computer without shutting it down
  - “Power off” can potentially cause data loss or an inconsistent state, meaning you may lose your work
- It’s better to type the command “sync” in the virtual machine every time you need a break
  - This explicitly synchronizes disks and make sure your data are safely stored

Attention: You MUST test your code in your class virtual machine

- Throughout the semester, all your labs and recitations will be graded inside the class virtual machine we provide to you

- You may do your assignments in your native OS, but you must test them inside the virtual machine

- If you fail to do so, as long as the result is wrong when graded inside the virtual machine, you will get ZERO for this lab/recitation. No excuse accepted!!!

Command line

- A command line interface (CLI for short) is somewhere issue commands to programs
  - You are interacting with a program known as “shell”
  - You issue commands to shell
  - Shell executes commands for you

Open up a terminal (command line)

- Click start icon (the bottom left icon)
- go to “system tools” tab, which will then expand
- click “LXterminal”

- The keyboard short cut to open up terminal is
  - ctrl + alt + t

- To copy paste in a terminal, you need to use
  - ctrl + shift + c, ctrl + shift + v

Basic Commands

- try the following commands in a terminal:
  - man
  - ls, cd, pwd, mkdir
  - cp, mv, rm
  - echo, cat
  - wc
  - grep
  - ctrl-c, ctrl-d, ctrl-z, fg, bg
  - |, >, <, >>
  - apt install/search
  - history, ctrl-r

Basic Commands

- Whenever you want to find out how to do something using command line, ask google first

- Here is a link contains useful command, for both beginners and experienced users:
  - https://github.com/jlevy/the-art-of-command-line

Editor and IDE

- A **text editor** is a type of program used for editing plain text files.

- An **integrated development environment** (**IDE**) is a software application that packs up editor, compiler, runtime

- A text edit is not an IDE but an IDE must contain a text edit.

Editor

- You need a good editor to code with for productivity
- Popular editors used by programmers:
  - vim
  - emacs
  - **sublime**
- We recommend you use Sublime Text
  - install sublime by “sudo apt install sublime-text”

Version Control

- What is version control?
  - Manages changes to documents, source files and other collections of information
- Why is version control indispensable?
  - track code changes
  - roll back to older version
  - collaborate with others
- We are going to use the popular “Git” as our version control system

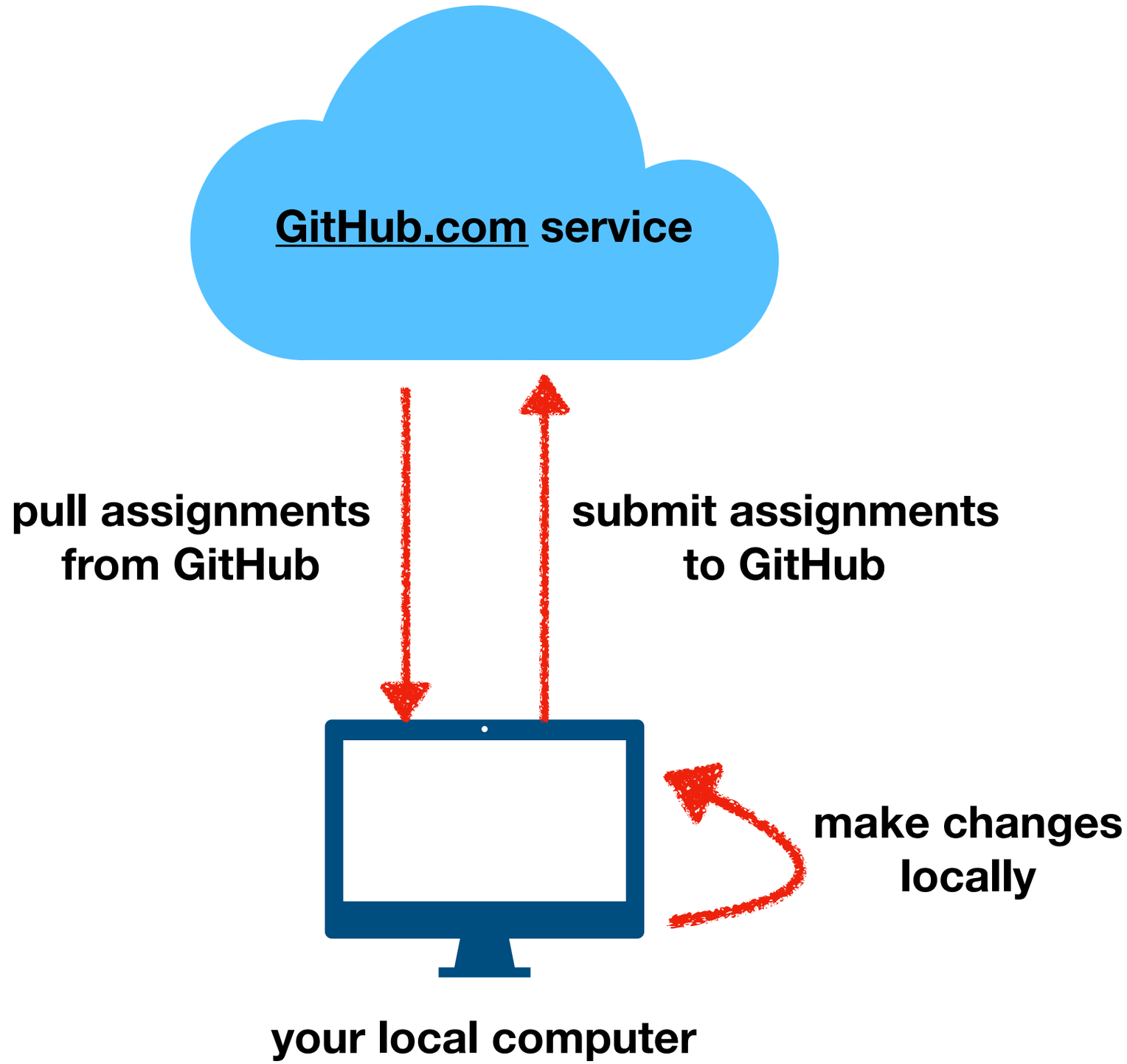
A list of git commands you need

- git clone
- git status
- git remote
- git add <file name>
- git commit -m <commit messages>
- git push origin master
- git pull upstream master

You need to config git first!

- **git config --global user.email “<Your Email>”**
- **git config --global user.name "<Your Name>”**
- **Please do above two commands ONLY ONCE.**
- **You can issue “git config --list” to check your configuration**

Git Overview

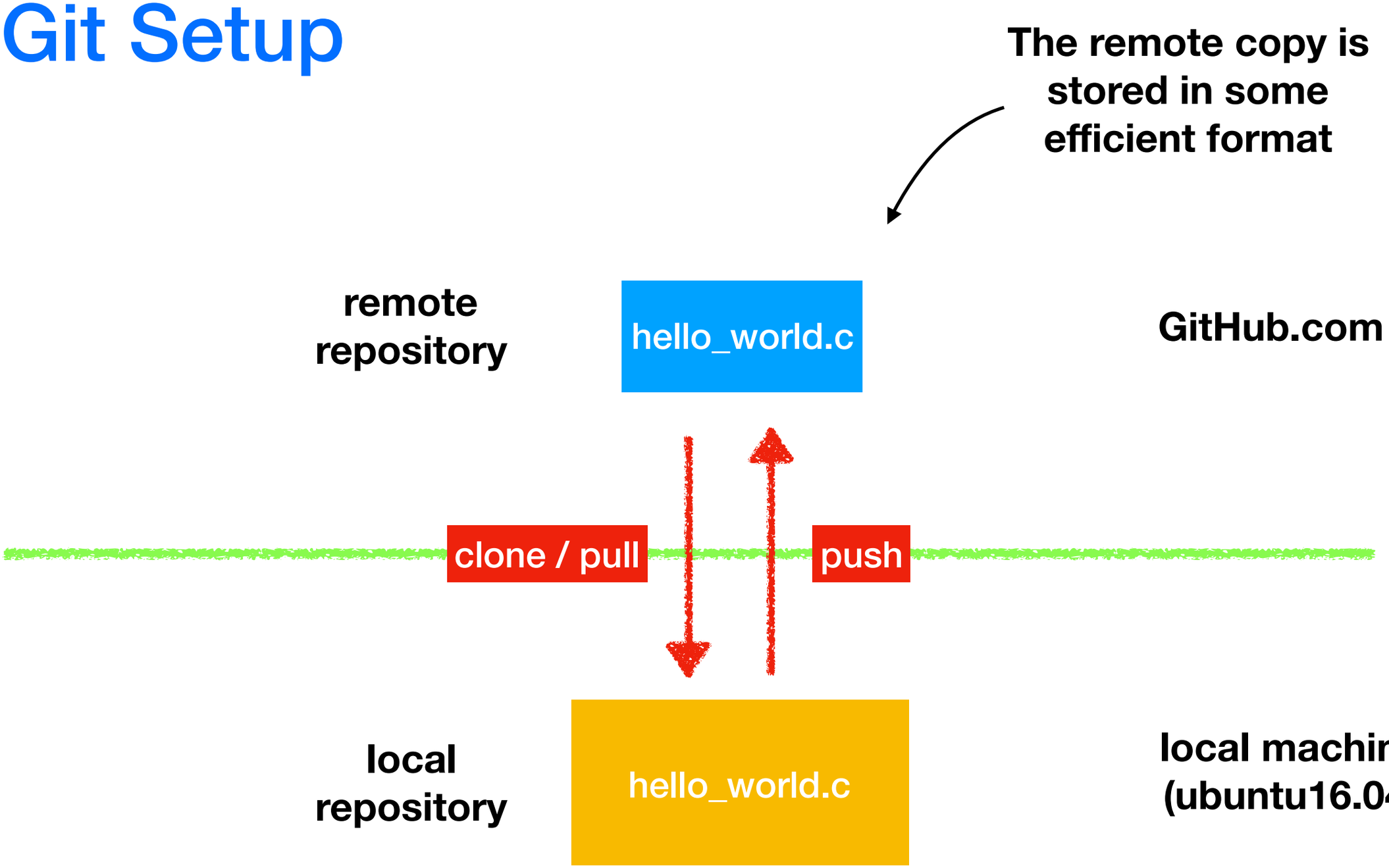


Setting up your recitation repo

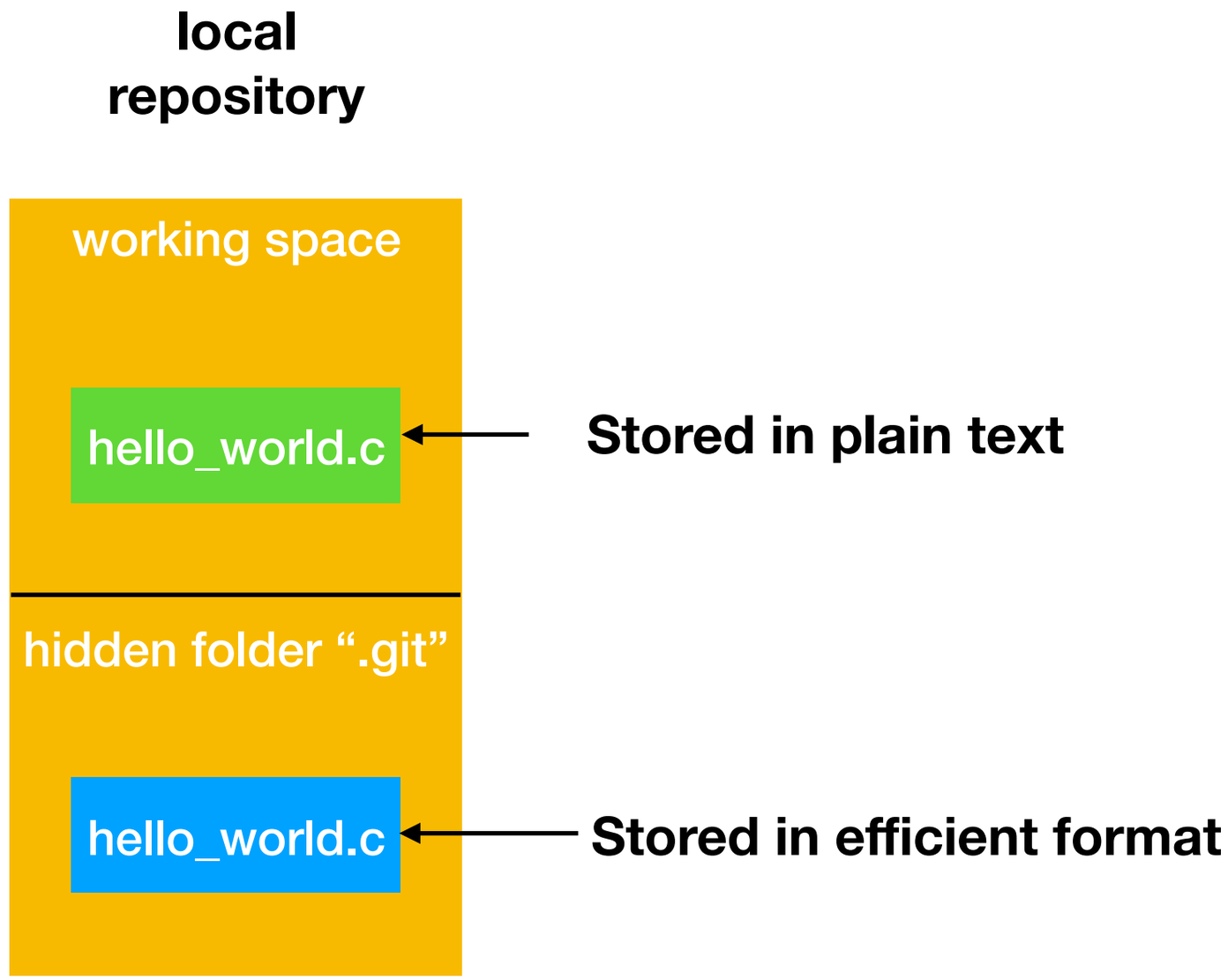
- In command line, type:
  - git clone https://github.com/nyu-cso-fa18/recitation-“Your GitHub Username”
    - If you copy the above command to command line, don’t let the line break
    - Replace “Your GitHub Username” (including the quote marks) with your GitHub username.
  - cd recitation-“Your GitHub Username”

- You only need to “git clone” once

Git Setup

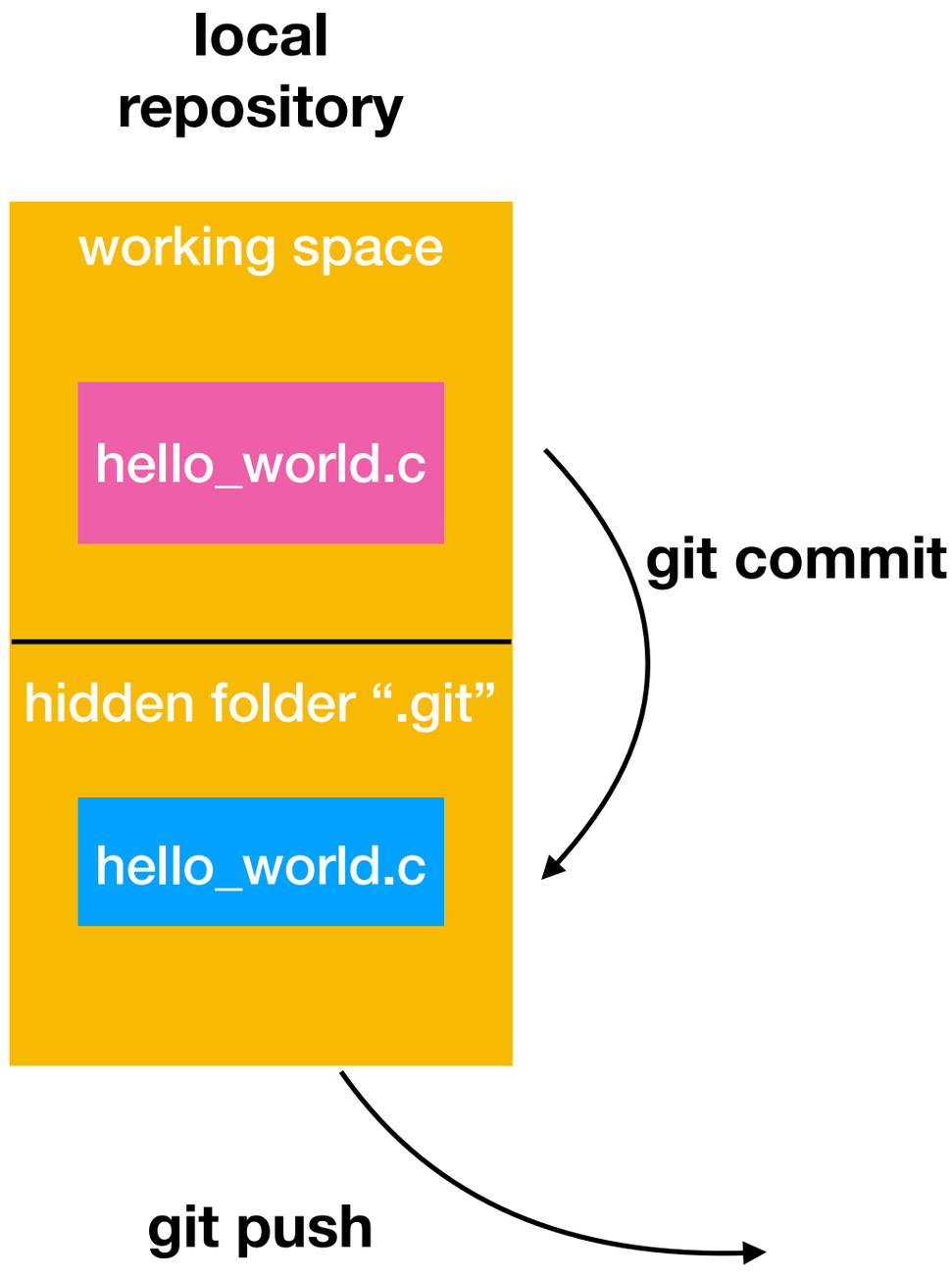

The remote copy is stored in some efficient format
remote repository
hello_world.c
GitHub.com
clone / pull
push
local repository
hello_world.c
local machine (ubuntu16.04)

A closer look at your local repository



How to interact with Git

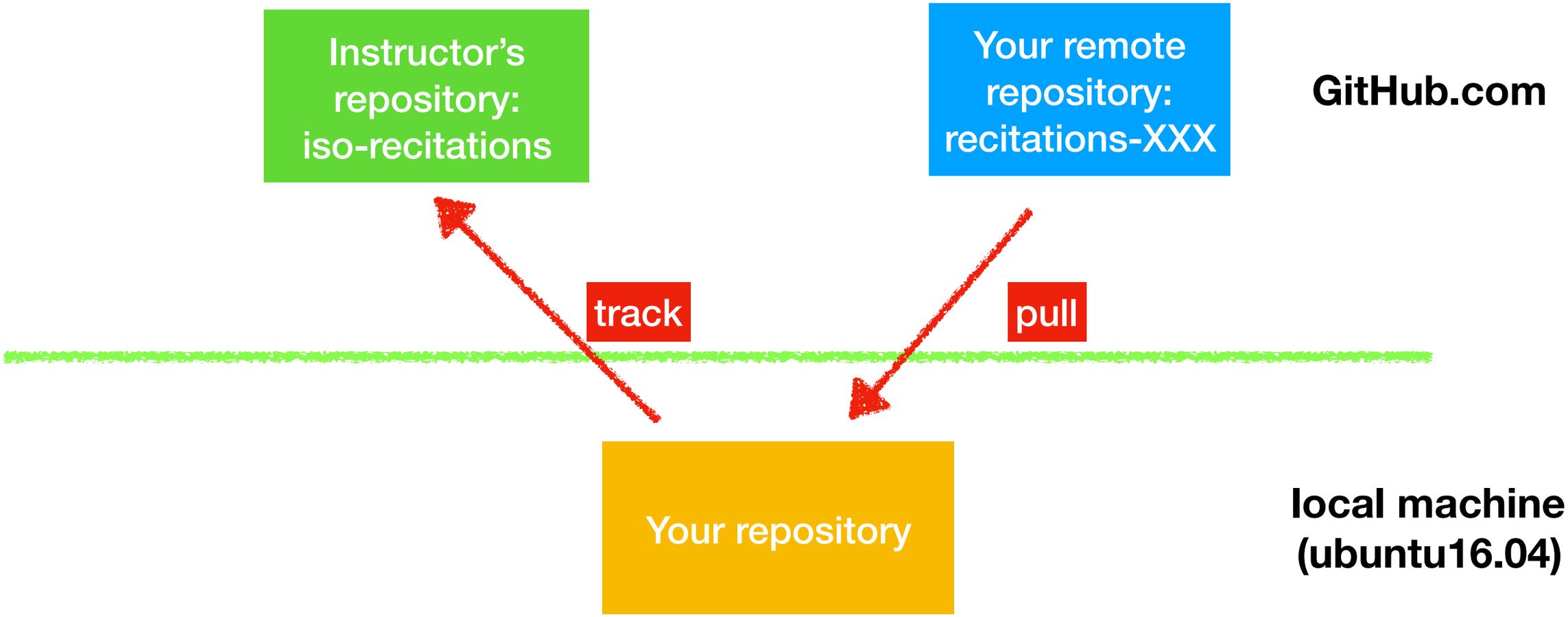
- git add hello_world.c
  - tell git to track changes to hello_world.c
- git commit
  - store tracked file to .git
- git push
  - submit commits to your remote repository



It’s not the end of story

- What if instructor post a new lab/recitation
- Where can I find it?
- How can I download it into my repo?

Lab/Recitation Setup



Track your instructor's repository

- In command line, type:
  - git remote add upstream https://github.com/nyu-cso-sp19/cso-recitations
    - Don't let the line break for above command when copy paste to terminal
- You only need to do this once

Work flow for our recitation and labs

Instructor's repository: iso-recitations

r01

r02

Your remote repository: recitations-XXX

r01

GitHub

pull

clone

git remote add

push

Your local repository

r01

r02

local machine (ubuntu16.04)

For each lab and recitation

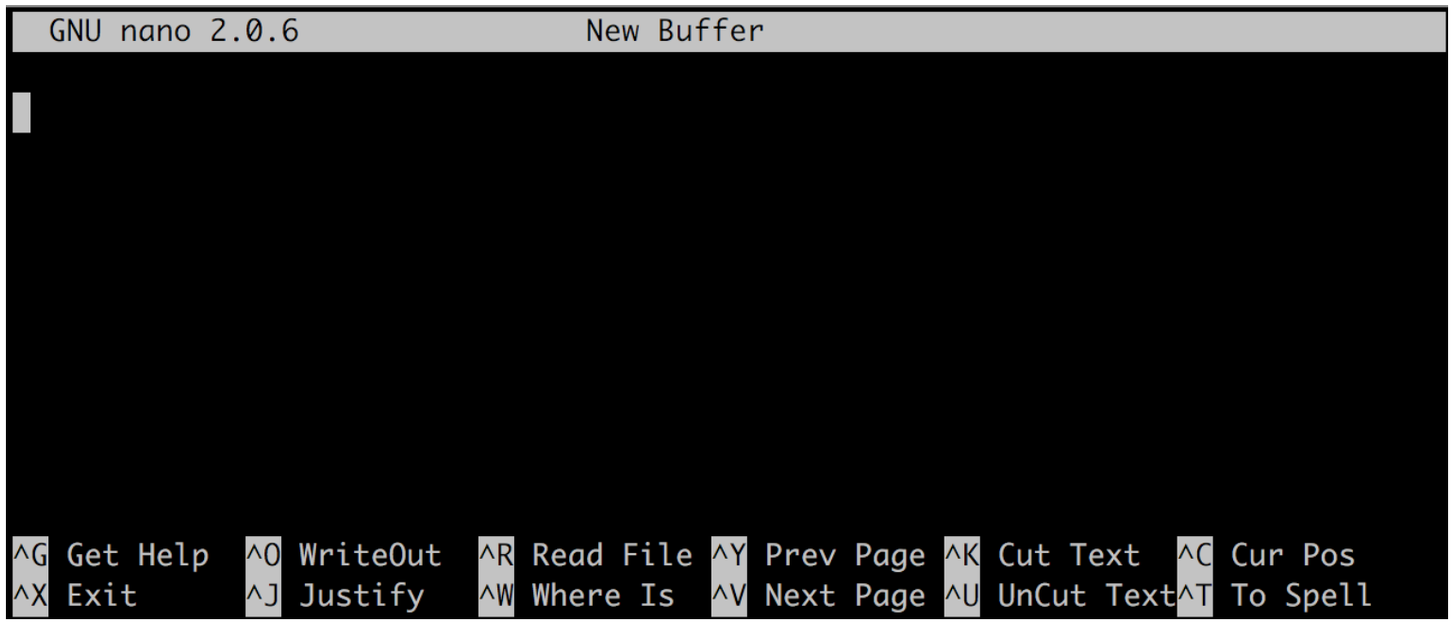
- pull latest lab and recitation materials
  - git pull upstream master
- then make changes locally on you computer
- tell git to track changes
  - git add “file name”
- commit changes
  - git commit -m “commit messages”
- submit to your remote repository
  - git push origin master

Git commit

- If you added some file before, you can skip “git add” by
  - git commit -am “message”
- When you issue “git commit”, you need to provide a message which is a short description of changes you made
- You can use “-m” option to provide commit message
  - git commit -m “my first commit”
- If you don’t use “-m” option, an command line editor will pop up for you to edit the commit message

How to get out of Nano Editor

- The default editor is called Nano.



- To exit, you need to
  - first type in some commit message
  - hit ctrl + o to save your commit message (^ means ctrl)
  - hit ctrl + x to exit

Double check with “git status”

- Sometimes, you might forget to do some (or all) of
  - git add, git commit, git push
- It’s always good to check the status of your repository
- git status tells you
  - what files are going to commit
  - what files are not tracked
  - whether you forget to push commits to remote

Double check with “git status”

```
➜  r01 git:(master) ✗ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

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

        modified:   CSO_CHEAT_SHEET.md

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   README.md

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

        r01-backup.key
```

Triple check with GitHub

- Still not sure/confident about whether assignment was submitted properly?

- Go to github.com, then go to your repo

- Manually check if every file contains the up-to-date information

Git is much more powerful than that

- Our git introduction only covers a small part of Git
- Git tutorial:
  - https://www.atlassian.com/git/tutorials/what-is-version-control
  - https://try.github.io/levels/1/challenges/1

All the git commands you need for CSO

- For beginners, it’s super easy to mess up Git
- After setting recitation and lab repository, you ONLY need to use the following git commands:
  - git add filenames
  - git commit -m “commit message”
  - git push origin master
  - git pull upstream master
  - git status

Warning: unless you know what you are doing, do not use any other git commands or git command flags

What if I got trouble with Git?

- A lot of knowledge about Git wasn't covered
  - branching
  - conflicts
  - etc.

- If you encounter any problem
  - Go through git instructions in r01/CSO_CHEAT_SHEET.md first
  - Search online for help

What if something goes really wrong

- If your repository is messed up beyond repair, you want a fresh start. do the following:
  - Do manual backup first
    - cp -r recitations-xxx recitations-xxx-backup
  - clone a fresh copy your repository
    - git clone <remote repo address> new_repo
  - manually copy your changes to new_repo (but don’t copy .git folder)
    - cp -r recitations-xxx/r01 new_repo
  - remove old repository
    - rm -r recitations-xxx
    - or you can save it somewhere else, but don’t use it any more

Ask the staff for help

- If you really cannot fix conflicts or other git problems, you should ask course staff for help
  - You need to email the staff to make an appointment
  - You should start your lab earlier
- Don’t randomly issue commands to further mess things up

Things you should Never do

- What CSO cheat sheet forbids
  - You need to sign it, it's today's homework.
- never use git add *, git add .
  - instead, you should always specify the file names you want to commit
- never modify any file using GitHub website
  - instead, you should always make changes locally on our laptop and then push commits to GitHub
  - otherwise, there will be conflicts

CSO cheat sheet

- Under r01 directory, there is a file called CSO_CHEAT_SHEET.md
  - This cheat sheet is summarized by previous CSO recitation leaders

- This files contains important course logistics, what you must or must not do, and instructions to resolve git conflicts and corruptions.

- In the future, whenever you get problems listed in this CSO_CHEAT_SHEET.md, you must follow instructions to resolve them.
  - failure in following instructions in the cheat sheet may lead to at least 20% of penalty.

Exercise to submit for today

- Read and sign the git cheat sheet following instructions in r01/README.md
  - change all XXX in CSO_CHEAT_SHEET.md to your GitHub username
  - don’t forget to change those in URLs or in Unix commands
- submit your cheat sheet to GitHub by
  - git add CSO_CHEAT_SHEET.md
  - git commit -m “sign cheat sheet”
  - git push origin master
- Due: Feb 6th, 2019 11PM ET

If you want to get a good grade for CSO

- I hope Professor has convinced you that CSO is pretty hard
  - You will find the grade of the class to be pretty uniformly distributed

- Expect to devote at least 15 hours to CSO after class every week
  - Or you might easily get a D…

- If you think you are lagging behind, speak to the Professor for help.

- Don’t wait until last second to ask for help or to submit your assignments

Summary

- Read the following A HUNDRED times to yourself
  - Shutdown VM properly, never use “power off” option
  - Always test your code in the VM before submitting
  - Follow the cheat sheet instructions
  - To submit any lab or recitation exercise, you need git add, git commit, AND git push, don’t forget any of them!!!
- Enjoy your CSO
- Office hour starts at 1 p.m. today at 60 Fifth Ave room 406