COMPUTER SCIENCE

UNIT F WEEK 2, TUESDAY MAR 27TH + THURSDAY MAR 29TH

WWW.CITA.UTORONTO.CA/~WOODFORD

WOODFORD@CITA.UTORONTO.CA

THIS WEEK IN CS AND STEM

- An Aurora named STEVE
  - https://futurism.com/nasa-new-aurora-steve/
- A not-so-classic twin case study
  - https://futurism.com/nasa-twin-study-kelly-preliminary/

- Robot clones! Made by Walmart™
  - https://futurism.com/robot-bees-drones-walmart/
- The future is now – flying cars in New Zealand
  - https://futurism.com/flying-taxi-new-zealand/

FINAL PROJECTS

- 3 different project descriptions, you need to:
  - Complete the coding, using all of the skills we've learned
  - Write user documentation
  - Write a report
  - Give a 10 minute presentation showcasing your work (last day of classes, June 5th)
- Due dates:
  - Pick your project (1 of the 3 given): April 3rd – tell me in person or in email!
  - Submit Report, code, supporting documents: June 4th at midnight
  - Give presentation: June 5th in class

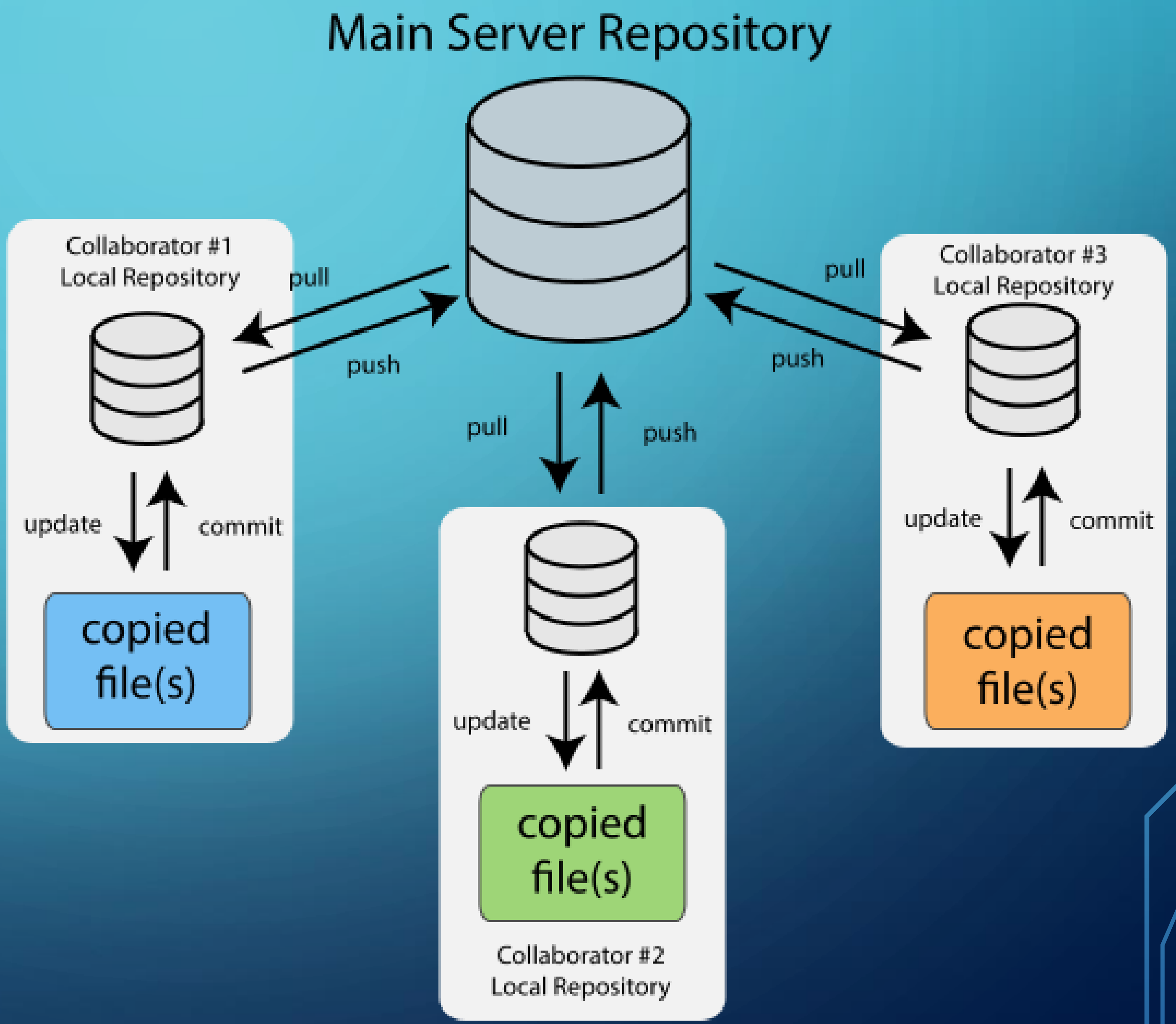
VERSION CONTROL REVIEW

- What is version control? Why is it important?
- We typically need to the following when working on shared or publically available codes:
  - Backup and Restore.
  - Synchronization.
  - Short-term undo.
  - Long-term undo.
  - Track Changes.
  - Track Ownership.
  - Branching and merging.

Key Terms:

- Repository
- Server
- Client
- Working Copy
- Main
- Check out/Pull
- Check in/Push
- Check in Message/ Commit
- Log/History
- Revert
- Branch
- Merge

Distributed Version Control



GIT : COMMANDS

- When you have a repository that uses git and have cloned it onto your machine, the most common commands are:
  - git push
  - git pull
  - git add [ ]
  - git commit –m [ ]
  - git log
  - git status

GIT CLONE

- First we need to get access to a repo.

- https://github.com/cwoodford/Abelard_CS
  - Make your own github account – it will allow you access to public repos on github like the one I've initialized above
  - You won't be able to clone yet, but let's look at the repo and figure out what we need to do to access it!

SSH AND TUNNELING

- In order to access the repo from your local machine, you need to git clone. There two ways to do this:
  - SSH
  - HTTPS
- You NEVER want to use HTTPS. So, we need to lean how to SSH.

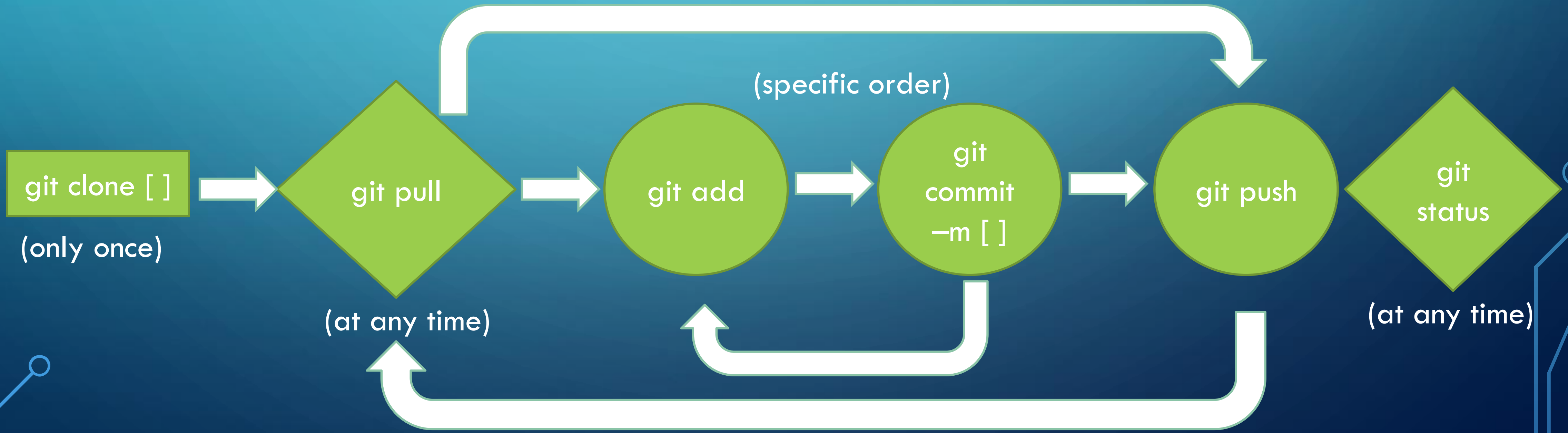
BACK TO OUR REPO…

- How to generate keys and put them into GitHub: https://help.github.com/enterprise/2.12/user/articles/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent/

- Let’s try this together!

HOW TO USE GIT – PRACTISE RUN

- Once you've cloned the repo onto your client (ie, your laptop), you can add and modify files. Once you've done a significant amount of work (up to you how much or little that is), you then add (git add) the file, and make a commit with a short message (git commit). When you're ready, you then push (git push) back to the main repo.



FORKING AND PULL REQUESTS

- My repo is open for anyone to edit. However, most of the time you don't want that to happen.
- Typically code will be available for pulling, but NOT for pushing changes. This is to protect the main repo from malicious or otherwise unsavoury changes.
- So, what to do?

- Forking: most common and best logistical way to have multiple people work on code but keep the original unchanged until the new changes are checked and verified by an admin to be OK.
- Fork (or branch) your own copy to work on. This is different than copying to your local repo – this creates a completely identical but separate main repo for you to then clone and work on. You can commit and push as you please to this repo.
- When you’re ready to try and merge the forks together (ie. The main and your copy), you make a pull request. The owner of the code will then verify that your version is an improvement without bugs and integrate it – or leave comments on what you should change before trying another pull request should there be something you missed.

TRY IT OUT!

- https://github.com/cwoodford/Abelard_CS_private

- You won’t be able to push to this branch (ie. The main branch). As it says in the ReadMe, you’ll have to fork, make your changes, commit and push to your fork, and then when you’re ready you can make a pull request.

COMMON COMMANDS

- We've already discussed git commands that are used constantly, but there are a few more that will also come in handy:
  - git init (for a repo that isn't being tracked by git, you can start tracking it using this)
  - git rm [ ] (remove a file, files, or whole directories. This removes them from the local repo.)
  - git reset [--hard, --soft] (undo recent commits in the local repo, can be the most recent or specify up to a specific commit)
  - **git config --global core.editor ["emacs –nw", vim] (automatically open an editor to write your message when using "git commit")**
- Note that all of the git commands we've seen and used have many options and flags. Use the "-h" flag to see all the options and their meaning, or search on the git website for further descriptions and examples!

ASSIGNMENT 13

- Due Monday April 2nd at midnight
- Using whichever server you prefer (github, bitbucket, etc), make your own git repo to house material for your final project. Note that you can keep everything and anything in the repo: code, notebooks, pdf, images, txt, Microsoft/Libre office documents, etc. Name the repo appropriately! It should have a ReadMe and at least 1 other file (any kind of file). Subsequently, there should be at least 2 commits (initial commit and one for adding the file).
- Send me the url to your repo by email - note that I should be able to pull but NOT push as this is your final project repo!
- Grading: K (40%) I (10%) C (10%) A (40%)

REFERENCES

- https://betterexplained.com/articles/a-visual-guide-to-version-control/
- https://www.visualstudio.com/learn/what-is-version-control/
- https://git-scm.com/
- https://git-scm.com/book/en/v2/Getting-Started-Installing-Git
- https://www.ssh.com/ssh/protocol/
- https://sethrobertson.github.io/GitFixUm/fixup.html
- https://guides.github.com/activities/forking/
- Video on github forks: https://www.youtube.com/watch?v=D2j0zebizdw