Source Control

Kendra Wannamaker and Jarrett Spiker
https://github.com/JarrettSpiker/GitProblems.git
If you Don’t know:

- What source control is
- How to create a repo
- How to clone a Repo
- How to push and pull

There is an introductory workshop
What is source control

A method of collaboration and recollection.

Assumption: you all have a basic understanding of the recollection part. How to push different version of your software and how to go back to these versions if you mess up. Today we will cover collaboration.
Who we are and Our Experience with git

Kendra Wannamaker

Jarrett Spiker
Understanding Git

- Git thinks of its data as a set of filesystem snapshots
- Local history means that most operations seem almost instantaneous.
- Git checksums every commit with a SHA-1 hash
- Takes the danger out of coding
Branching

Work on a project without affecting the main version.

Multiple developers make incremental changes without affecting one another

git branch testing

git checkout testing

git checkout -b testing

git branch -v (the last commit on every branch)
git branch -D
Branching Best Practices

Case insensitive

Naming conventions

Delete Complete Branches

Avoid adding certain file types
Warm Up!

https://github.com/JarrettSpiker/GitProblems.git

git checkout Names

git checkout -b Names_<your name>

Modify the function that corresponds to your initials!
Merging

Incorporate your work with that of others

`git fetch`

`git merge <branch>`

`git merge <branch1> <branch2>`
MERGING EXERCISE

Fetch everyone else’s branches, and merge them into yours!

Contest:

Who can get everyone’s names printing first?
Rebasings

Alternative to merging.
Add all the existing changes before yours/
fix your commits

ebase <branch>

Rebase -i HEAD~<n>
GOLDEN RULE OF REBASEING!!!!

Dont rebase public branches. If a branch contains multiple people’s work, or other people are working off of it DO NOT REBASE IT.
REBASING EXERCISE

https://open.kattis.com/problems/busyschedule

Checkout RebaseFestival
REBASING EXERCISE

Fix lines 18 and 20

Should be “AM” not “A.M.”
REBASENG EXERCISE

Fix lines 28 and 30

Should be “AM” not “A.M.”
REbasin Exercise

Rebase your changes onto Rebase_Jarrett
Cherry Picking

Given one or more existing commits, apply the change each one introduces, recording a new commit for each. This requires your working tree to be clean (no modifications from the HEAD commit).

Git cherry-pick [--edit] [-n] [-m parent-number] [-s] [-x] [--ff] [-S[<keyid>]] <commit>...

git cherry-pick --continue

git cherry-pick --quit

git cherry-pick --abort
Git Config

~/.gitconfig or ~/.config/git/config

$ git config --global user.name "John Doe" $ git config --global user.email johndoe@example.com

  git config --global core.editor emacs

  git config --list

$ git help $ git --help $ man git-
Git Ignore

Should ignore:

Pictures, Jars, Etc

automatically generated files

Example: .gitignore file:

doc/server/arch.txt

doc/

doc/**/*.{pdf}
Git Log

When you run git log in this project, you should get output that looks something like this: $ git log

```
git log --pretty=oneline

  git log --pretty=format:"%h - %an, %ar : %s"
```

Git Diff

git diff

git diff --staged

git diff a (a could be branch or hash)
**Merge Conflict Exercise**

checkout IncreasingNumbers

IncreasingNumbers_a and IncreasingNumbers_b have the fixes, but they conflict

Diff against the second to last commit in IncreasingNumbers for reference
**Reset**

Git reset moves where you are pointing in the tree:

```bash
git reset --hard <commit>:
```

Keeps all the work as modified files:

```bash
git reset --soft <commit>
```
Commit amend

Allows you to fix up your most recent commit instead of creating an entirely new snapshot

git commit --amend
Squash

This is a great way to group certain changes together before sharing them with others.

1. git rebase -i HEAD~x
2. This will open up an editor with a list of commits
3. Change the word “pick” to “squash”
**FIX the Tree Exercise!!!!**

You want to print “hello world”...but you are forbidden from writing code!

Checkout HelloWorld

The HelloWorld_ prefixed branches have all changes you’ll need
Helpful Links

https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow

